Efficient Time Integration Methods for Linear Parabolic Partial Differential Equations with Applications

Von der Fakultät 1 – MINT – Mathematik, Informatik, Physik,
Elektro- und Informationstechnik
der Brandenburgischen Technischen Universität Cottbus-Senftenberg
genehmigte Dissertation
zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von

M. Sc. Martin Bähr
geboren am 13.05.1988 in Cottbus, Deutschland

Vorsitzender: Prof. Dr. rer. nat. habil. Ralf Wunderlich
Gutachter: Prof. Dr. rer. nat. habil. Michael Breuß
Gutachter: Prof. Dr. rer. nat. habil. Andreas Meister
Gutachter: Prof. Dr.-Ing. Michael Oevermann

Tag der mündlichen Prüfung: 10.12.2021

DOI: https://doi.org/10.26127/BTUOpen-5753

Danksagung

An dieser Stelle möchte ich die Gelegenheit nutzen, all jenen zu danken, die mich bei der Anfertigung meiner Dissertation unterstützt haben.

Mein besonderer Dank gilt Prof. Michael Breuß, der mir die Möglichkeit gegeben hat, als "Optimierer" in die Welt der Numerik einzutauchen und meine Dissertationsschrift unter seiner Leitung anfertigen zu dürfen, für seine langjährige Betreuung und für die zahlreichen fachlichen Diskussionen und Hinweise, die zum Entstehen dieser Arbeit maßgeblich beigetragen haben.

Weiterhin möchte ich mich in besonderem Maße bei Prof. Andreas Meister und Prof. Michael Oevermann für die Begutachtung meiner doch recht umfangreichen Dissertation bedanken.

Für den Austausch, die interessanten Gespräche und das angenehme Arbeitsklima danke ich allen Kollegen und Kolleginnen des Lehrstuhls Angewandte Mathematik - insbesondere Robert Dachsel & Georg Radow für die fruchtbaren mathematischen Diskussionen; Silke Büttner für den regelmäßigen technischen Support; Annette Kallweit & Ashkan Yarahmadi für die moralische Unterstützung - sowie Angie Burtchen für ihre Ratschläge, Ermutigungen und wertvolle Unterstützung auf meinem langen Weg zur Fertigstellung dieser Arbeit.

Mein größter Dank gilt jedoch meiner Familie und meinen Freunden, die mich insbesondere während der Erstellung dieser Arbeit uneingeschränkt und unermüdlich auf emotionaler Ebene unterstützt haben und sich zu ihren eigenen Leidwesen sehr häufig zurücknahmen, um mir damit den Raum zu schaffen, diese Phase meines Lebens erfolgreich zu beschreiten.

Abstract

In this thesis we study efficient time integration methods for linear parabolic partial differential equations (PDEs) to solve practical problems that arise in a variety of real-world applications. The classical construction of numerical methods for solving PDEs is based on the method of lines, which leads to a large sparse semi-discretised system of ordinary differential equations (ODEs) to which any numerical method for initial value ODE problems can be applied. The standard method for solving such ODE systems and computing an approximate solution to the PDE uses numerical time integration. When dealing with parabolic-type problems, the underlying ODE systems are known to be stiff. Therefore, in the context of linear parabolic problems, the use of implicit schemes is usually considered to be the best choice in practice.

However, this statement is not entirely correct for some relevant real-world applications in image processing and computer vision or engineering. In particular, implicit schemes can cause high computational costs for various practical problems that are equipped with certain model conditions. Three examples of practical importance that we will focus on in this thesis are of such a type. The model problems considered here are coupled with various settings, ranging from many different initial conditions over long-term simulation with relatively frequent model updates, to dealing with very large-scale problems for which the matrix size can exceed several millions. For this reason, we are interested in sophisticated and computationally efficient numerical methods that bring the aspects of approximation accuracy as well as computational and storage complexity into balance.

Although several numerical solvers are available marking the state-of-the-art in diverse scientific fields, even nowadays it is still a challenging task to devise a numerical method that combines high accuracy, robustness and computational efficiency for the model problem to be solved. Therefore, the main objective is to find an easy and efficient ODE integration scheme for each individual model problem that is dealt with in this thesis. On this basis, we first give a comprehensive overview and introduction to the state-of-the-art methods that are often used for practical purposes. In this framework, we will investigate very detailed the theoretical and numerical foundations of two popular techniques that are widely used in their respective scientific fields, namely the fast explicit methods and the model order reduction techniques. This is primarily important in order to fully understand the numerical methods, and also absolutely essential in finding the best numerical method that is specifically suitable for the intended purpose.

Our second goal is then to efficiently solve the relevant practical problems that arise in connection with shape correspondence, geothermal energy storage and image osmosis filtering. For each application we specify a complete setup, and in order to provide an efficient and accurate numerical approximation, we give a thorough discussion of the various numerical solvers along with many technical details and own adaptations. We validate our numerical findings through many experiments using synthetic and real-world data. In this way we show that we can obtain fast and accurate numerical methods for solving the problems in this thesis. In addition, the thesis provides a complete and detailed description of the powerful methods that can be very useful for tackling similar problems that are the subject of interest in many applications.

Zusammenfassung

In dieser Arbeit untersuchen wir effiziente Zeitintegrationsmethoden für lineare parabolische partielle Differentialgleichungen (PDEs), um praktische Probleme zu lösen, die in einer Vielzahl von realen Anwendungen auftreten. Die klassische Konstruktion von numerischen Methoden zur Lösung von PDEs basiert auf der Linienmethode, die zu einem großen, dünnbesetzten und halbdiskretisierten System gewöhnlicher Differentialgleichungen (ODEs) führt, auf das anschließend jede numerische Methode für die Lösung eines Anfangswertproblems angewendet werden kann. Die Standardmethode zum Lösen solcher ODE-Systeme und zum Berechnen einer Näherungslösung für die PDE verwendet die numerische Zeitintegration. Bei parabolischen Problemen ist bekannt, dass die zugrunde liegenden ODE-Systeme steif sind. Daher wird im Zusammenhang mit linearen parabolischen Problemen die Verwendung impliziter Methoden in der Praxis normalerweise als die beste Wahl angesehen.

Diese Aussage ist jedoch für einige relevante Anwendungen in der Bildverarbeitung und Computer Vision oder im Ingenieurwesen nicht ganz richtig. Insbesondere können implizite Methoden hohe Rechenkosten für verschiedene praktische Probleme verursachen, die mit bestimmten Modellbeschränkungen verbunden sind. Drei Beispiele von praktischer Relevanz, auf die wir uns in dieser Arbeit konzentrieren werden, sind von einem solchen Typ. Die hier betrachteten Modellprobleme sind mit verschiedenen Einstellungen verbunden, die von vielen unterschiedlichen Anfangsbedingungen über die Langzeitsimulation mit relativ häufigen Modellupdates bis hin zur Behandlung sehr großer Probleme reichen, bei denen die Matrixgröße mehrere Millionen überschreiten kann. Deswegen sind wir an ausgefeilten und rechnerisch effizienten numerischen Methoden interessiert, die die Aspekte der Approximationsgenauigkeit sowie der Rechen- und Speicherkomplexität in Einklang bringen.

Obwohl viele numerische Löser verfügbar sind, die den Stand der Technik in diversen Forschungsbereichen markieren, ist es auch heute noch eine herausfordernde Aufgabe, eine numerische Methode zu konzipieren, die hohe Genauigkeit, Robustheit und Recheneffizienz für das zu lösende Modellproblem kombiniert. Das Hauptziel besteht darin, ein simples und effizientes ODE-Integrationsschema für jedes Modellproblem zu finden, das hierin behandelt wird. Zunächst geben wir einen umfassenden Überblick und eine Einführung in die modernsten Methoden, die oft für praktische Zwecke eingesetzt werden. In diesem Rahmen werden wir die theoretischen und numerischen Grundlagen zweier populärer Techniken, die in ihren jeweiligen wissenschaftlichen Bereichen weit verbreitet sind, sehr detailliert untersuchen, nämlich die schnellen expliziten Methoden und die Modellordnungsreduktionstechniken. Das ist erstens wichtig, um die Verfahren vollständig zu verstehen, und ist unbedingt erforderlich, um die beste numerische Methode zu finden, die speziell für den beabsichtigten Zweck geeignet ist.

Das zweite Ziel ist dann, die relevanten praktischen Probleme im Zusammenhang mit Formkorrespondenz, geothermischer Energiespeicherung und osmosebasierte Bildverarbeitung effizient zu lösen. Für jede Anwendung geben wir einen vollständigen Aufbau an. Um eine effiziente genaue numerische Approximation bereitzustellen, werden die numerischen Löser zusammen mit vielen technischen Details und eigenen Anpassungen ausführlich erläutert. Wir validieren unsere numerischen Resultate durch Experimente mittels synthetischer und realer Daten. Hierdurch zeigen wir, dass wir schnelle und genaue numerische Methoden zur Lösung der Probleme in dieser Arbeit erhalten können. Außerdem bietet die Arbeit eine vollständige und detaillierte Beschreibung der leistungsstarken Methoden, die sehr nützlich sein können, um ähnliche Probleme anzugehen, die in vielen Anwendungen von Interesse sind.

Contents

Acronyms

ADI	alternating direction implicit
AMOS	additive-multiplicative operator splitting
AOS	additive operator splitting
BiCGSTAB	biconjugate gradient stabilized
BT	balanced truncation
CG	conjugate gradient
CN	Crank-Nicolson
CPU	central processing unit
EE	explicit Euler
FED	fast explicit diffusion
FEDRK	fast explicit diffusion Runge-Kutta
GEP	generalised eigenvalue problem
GES	geothermal energy storage
GPU	graphics processing unit
HKS	heat kernel signature
IC	incomplete Cholesky
IE	implicit Euler
KSMOR	Krylov subspace model order reduction
LU	lower-upper
MCR	modal coordinate reduction
MG	multigrid
MIC	modified incomplete Cholesky
MICO	multi-input-complete-output
MIMO	multi-input-multi-output
MOL	method of lines
MOR	model order reduction
MOS	multiplicative operator splitting
MSE	mean squared error
ODE	ordinary differential equation

PCG preconditioned conjugate gradient
PDE partial differential equation
POD proper orthogonal decomposition
PR Peaceman-Rachford
PSC pointwise shape correspondence
RK Runge-Kutta
RKC Runge-Kutta-Chebyshev
RKL Runge-Kutta-Legendre
SEP symmetric eigenvalue problem
SICO single-input-complete-output
SISO single-input-single-output
STS super time stepping
SVD singular value decomposition
WD weighted-Douglas
WKS wave kernel signature
ZISO zero-input-single-output

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

There is a very large number of scientific areas such as natural, engineering and medical sciences, in which many important problems are modelled with linear or nonlinear *partial differential equations (PDEs)* in several dimensions. Second order parabolic equations are one of the most common classes of PDEs. Many complex problems that arise e.g. in astrophysics, biology, combustion, fluid mechanics, medicine, imaging and vision, mathematical finance, chemistry are formulated by nonlinear parabolic-based models, for some examples see [4, 120, 138]. However, there are plenty of situations in which physical, mechanical and mechatronic, micro-electromechanical and mathematical imaging and visual processes in problems related to *heat transfer* can be mathematically described with linear parabolic-type PDEs.

In general, heat transfer is mainly characterised by various mechanisms such as heat conduction, heat radiation or heat convection. In many practical applications the most important transport mechanism is *heat conduction* which describes the heat flow within and through some material itself. The associated PDE, called the *heat equation* or the *diffusion equation*, is a classic and well-studied differential equation. In the case of simplified but still realistic model problems, the underlying medium can be considered as homogeneous and isotropic, so that the Laplace differential operator involved is of linear form. In image processing, the digital image intensities correspond to concentrations, so that physical processes such as diffusion can be applied to manipulate the given image. If then diffusion does not depend on the evolving image over time, linear diffusion is obtained. The basic model equation in the form of a linear heat equation can be written as

$$\rho\, c\, \partial_t u(\boldsymbol{x}, t) = \lambda \Delta u(\boldsymbol{x}, t) + f(\boldsymbol{x}, t), \quad (\boldsymbol{x}, t) \in \Omega \times [0, t_F] \tag{1.1}$$

with constant parameters known as thermal conductivity λ, density ρ, specific heat capacity c and where f represents various heat sources and sinks. Here, ∂_t denotes the partial derivative with respect to time t, i.e. $\partial_t u(\boldsymbol{x}, t) := \frac{\partial u(\boldsymbol{x}, t)}{\partial t}$, and Δ declares the spatial *Laplace operator* or *Laplacian* in the continuous setting. More precisely, the Laplacian of a twice-differentiable real-valued function g in n-dimensional Euclidean space is defined as the sum of all the unmixed second partial derivatives in the Cartesian coordinates x_k:

$$\Delta g = \sum_{k=1}^{n} \frac{\partial^2 g}{\partial x_k^2} \tag{1.2}$$

A complete setup that covers a realistic scenario also includes various initial and boundary conditions, in which the latter is usually based on the Dirichlet-, Neumann- and Robin boundary conditions.

Once the physical model has been generated in mathematical terms as PDEs based on spatial and temporal derivatives from the model-dependent technical process, methods for the solution of such PDEs are required. Although the model problems considered here are linear, for which closed-form solutions may be available by the ansatz of the separation of variables in simple situations, their explicit calculation on a grid is computationally intensive, since the analytical solution is represented by an infinite series. In order to investigate the predictions of the PDE models, it is therefore necessary to numerically approximate their solution. The most popular technique for solving time-dependent PDEs is the *method of lines (MOL)*, see e.g. [138, 161, 189, 247], in which all but one partial derivatives are discretised. This proceeding leads directly to a *semi-discretised* system of *ordinary differential equations (ODEs)* with just one independent variable to which an appropriate numerical method for initial value ordinary equations can be applied. In other words, the focus of MOL is the calculation of accurate numerical solutions. This highlights a significant advantage of the MOL approach, since semi-discrete problems are much simpler to solve and far better to understand than PDEs. The standard construction is usually conducted by first discretising the spatial derivatives only and leaving the time variable continuous, which gives a large ODE system (mostly $n = 2, 3$) with each component of the system corresponding to some grid point as a function of time. In this context, the spatial derivatives are discretised with approximations through e.g. finite differences, finite volume or finite element methods.

Let us assume that the continuous linear parabolic model problem (1.1) subjected to some boundary conditions has been discretised in space on a certain grid with in total m grid point x_i. According to the MOL approach, (1.1) can be transformed into a semi-discretised and time-continuous linear system of ODEs in the form

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + \boldsymbol{w}(t), \quad t \in (0, t_F],\ \boldsymbol{u}(0) = \boldsymbol{u}^0,\ \boldsymbol{w}(0) = \boldsymbol{w}^0 \tag{1.3}$$

where the vector $\boldsymbol{u}(t) \in \mathbb{R}^m$ is the spatial discretisation of the unknown solution u, the negative semi-definite large sparse matrix $L \in \mathbb{R}^{m\times m}$ represents the *discrete Laplacian*, the time-dependent vector $\boldsymbol{w}(t) \in \mathbb{R}^m$ includes the terms related to the boundary conditions and heat sources/sinks, and $\boldsymbol{u}^0$, $\boldsymbol{w}^0 \in \mathbb{R}^m$ are the given initial vectors. The matrix L contains the coefficients arising from the discretisation of the Laplace operator Δ. Moreover, the *sparsity* of L depends on the spatial discretisation, in which only a small neighbourhood of a considered grid point is involved, so only a small number of entries relative to the matrix dimension are non-zero. The specific type (1.3) is known as linear *first order initial value problem*, and the linearity of the system refers to the fact that the matrix coefficients are constant. Obviously, the system remains linear for time-dependent matrices $L(t)$, but in this thesis we focus on ODE systems with time-invariant Laplacians in the form (1.3). We mention that time-invariant systems are also referred to as autonomous. In addition, if $\boldsymbol{w}(t) = \boldsymbol{0}$, such a system is said to be homogeneous, otherwise nonhomogeneous. On closer inspection, (1.3) is a coupled and typically large-scale system of m ODEs for the unknown variables in $\boldsymbol{u}(t)$ which vary continuously in time along the lines. In order to determine the solution of the underlying PDE, an approximate solution for the initial value ODE problem must be computed. The solution of this ODE system gives m functions $u_1(t), u_2(t), \ldots, u_m(t)$, one for each grid point, that approximate $u(\boldsymbol{x}, t)$ at the grid points $i = 1, 2, \ldots m$.

Since ODEs have a long tradition and are omnipresent in real-world application, their solution by discrete procedures is one of the oldest areas of numerical computation and is

successfully used in all scientific fields. Although linear ODE systems have been thoroughly studied and have exact solutions represented in simple terms, their numerical evaluation can be computationally intensive. This is true even for the simplest linear ODE system. For example, consider the linear autonomous system

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{1.4}$$

for which the exact solution is simply given by

$$\boldsymbol{u}(t) = \exp(Lt)\boldsymbol{u}^0 = e^{Lt}\boldsymbol{u}^0 \tag{1.5}$$

In simple terms, the exact solution (1.5) is solely based on the matrix exponential e^{Lt}. Nevertheless, the computation of the matrix exponential is still a tricky problem nowadays. There are many methods [187] for computing e^{Lt} based on results in analysis, approximation theory and matrix theory, but usually only the Krylov-type methods are generally useful for large sparse problems. For this reason, it is natural to seek efficient numerical methods to ODEs that are able to approximate the solution to any desired accuracy.

Basic Numerical Schemes Many methods for the numerical solution of general differential equations have been developed. In general, the construction of approximation schemes to solve ODE systems numerically is based on the discretisation of the continuous time interval into a set of discrete time points. The time points have either fixed or variable spacing, where the distance between two points is often called *time step size*. More precisely, a temporal grid is used analogously to spatial discretisation. On this basis, the approximation to the solution at a time level can be determined by using one or several former solutions computed. For designing a numerical scheme, the *time integration method* [138] is normally applied in which the ODE system is numerically integrated using the fundamental theorem of calculus. However, there are other methods like exponential integrator methods [137], alternatives such as Parker-Sochacki method [111], artificial neural networks [249], or symplectic integrator method [119] especially designed for special classes of ODEs.

Following the time integration method to the system (1.4), the popular first order *explicit scheme* reads

$$\boldsymbol{u}^{k+1} = (I + \tau L)\,\boldsymbol{u}^k \tag{1.6}$$

and the first order *implicit scheme* yields

$$(I - \tau L)\,\boldsymbol{u}^{k+1} = \boldsymbol{u}^k \tag{1.7}$$

with uniform time step size $\tau > 0$ and $\boldsymbol{u}^k \approx \boldsymbol{u}(k\tau)$. In both cases, of course, the semi-discrete problem is converted into a *fully discrete* problem. The quality of the approximation $\boldsymbol{u}^{k+1}$ depends on two aspects: the error made by the approximation itself and the error caused by continuing from approximate solution values. These two aspects are declared by the terms of consistency and stability. Besides quality aspects, the computational effort of the generated approximate solution is also of practical importance. In particular, the computational efficiency of a solver is a key requirement for all current and future practical purposes. To this end, the numerical results are typically reported in terms of accuracy and *central processing unit (CPU)* time to confirm the efficiency of a proposed numerical method.

Explicit methods like (1.6) are based on extremely easy evaluations as their computational costs correspond to simple sparse matrix-vector multiplications. They are very easy to implement and due to their explicit nature well-suited for parallel computing such as *graphics processing units (GPUs)*. Computations on GPUs offer a remarkable performance through multi-core parallelism so that the CPU time required can be drastically reduced. Unfortunately, the semi-discretised ODE systems related to parabolic problems are known to be stiff, meaning that explicit schemes suffer from severe time step size restrictions on τ to satisfy numerical stability. The use of a very small τ therefore makes explicit methods generally unusable in practice. In contrast, implicit methods like (1.7) may have no time step size restriction, but they require the inversion of large sparse matrices. This means that it is necessary to solve a very large sparse system of equations at each time level, which can be computationally costly and more difficult to parallelise. Fortunately, the underlying systems as considered here are linear and can be efficiently solved by a factorisation or preconditioning technique that is precomputed once only. As a result, they are relatively simple to implement or based on existing sophisticated software packages and have moderate computational and memory complexity. Hence, for linear parabolic problems, researchers clearly consider the use of implicit schemes as the best choice.

1.1 Problem Formulation and Motivation

As stated above, implicit schemes based on the solution of large sparse systems of linear equations are predestined for linear parabolic model problems in practice. However, for some applications in image processing and computer vision or engineering this statement is not a true one. Let us give three examples that we will focus on in this thesis that are still relevant applications and of practical importance. For this purpose, we give a short overview of the practical problems related to linear parabolic-based PDEs, see Figure 1.1 for an illustration.

Shape Correspondence The main task in shape correspondence is to retrieve similarities between two or more similar three-dimensional objects. An important building block of many methods constructed to achieve this goal is a simplified shape representation by means of a shape descriptor, which characterises geometry around the points that define the surface of a given shape. An interesting class of models for such descriptors is based on the Laplace-Beltrami operator which enables to describe intrinsic geometric properties of a surface. To this end, several geometric PDEs have been proposed such as the heat, wave or Schrödinger equation. Geometric PDEs are characterised by the fact that they take into account geometric surface information, although the geometry does not change during time evolution. The underlying PDEs that are used have to be solved for each point and on each shape for a fixed time interval. Consequently, the computational costs are directly related to the number of points of the regarded shapes, and thus solving a system of linear equations with multiple right-hand sides appears to be rather impractical for shapes with many thousands of points. For this reason, it is necessary to find a fast and accurate numerical scheme.

Geothermal Energy Storage Besides efficient energy generation it is also important to store it, ideally with minimal losses over long periods of time. The recent geothermal energy storage technology provides a potential solution to energy storage, in which excess energy

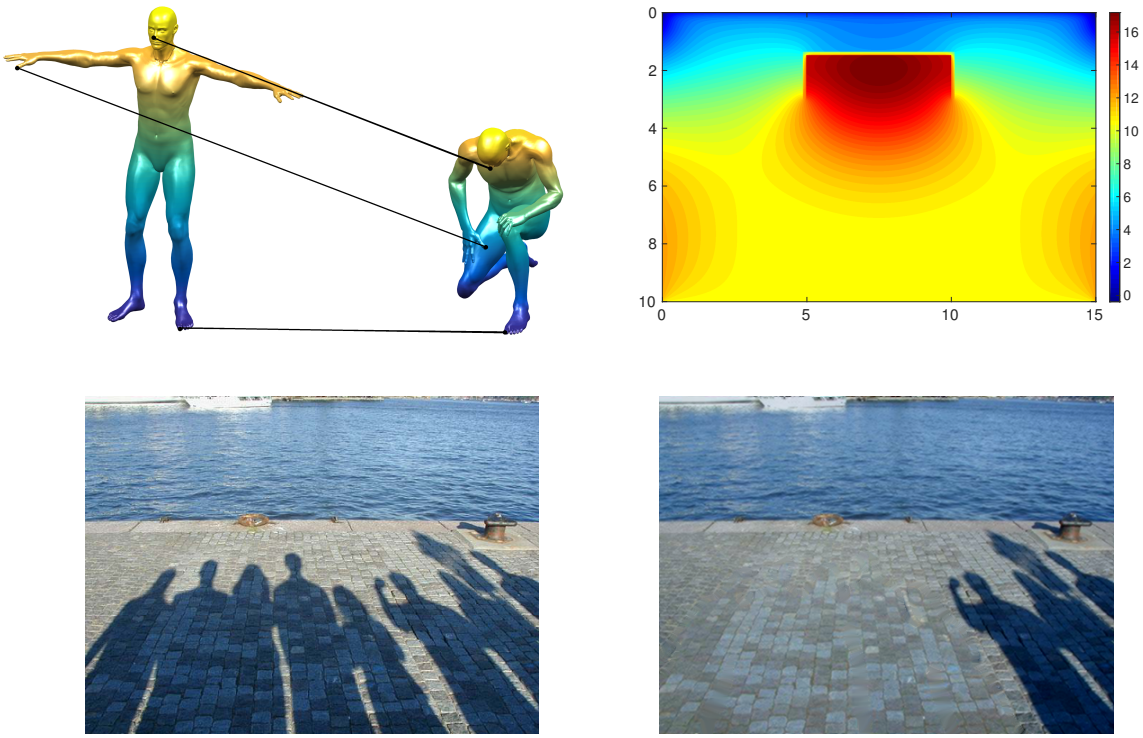


Figure 1.1: Main applications related to linear parabolic-based PDEs considered in this thesis. **Top left:** Shape correspondence. **Top right:** Geothermal energy storage. **Bottom:** Osmosis-based shadow removal with (**left**) initial image and (**right**) result.

generated during the summer can be easily stored. Interestingly, the latter heat tank is closed by insulating walls upwards and to the sides, is open downwards, and interacts with its environment, more precisely by the earth below the tank. The technical realisation by a downwardly open heat tank is cost effective, since in practice it provides a multiple of the capacity that is making up the actually relatively small tank. However, this setup becomes a very important issue when the entire heating system is considered over a year or even several years, depending on weather conditions and individual consumer demand. On the one hand, the supply for the consumer must be ensured, while the economic viability based on the optimal dimensioning of the heat tank is of great importance. Because of this, well-founded long-term heat evolution simulations of a geothermal energy storage are required to assess the profitability. In this context, in applications with source terms, it has to be kept in mind that the contributions of the sources must be updated at relatively small time intervals for obtaining an accurate simulation. Thus, implicit methods have to solve many large sparse systems of linear equations which leads to a high computational effort. For this purpose, the particular challenge is to find an efficient numerical method in connection with long-term simulation of a geothermal energy storage.

Osmosis Filtering Nowadays, digital image processing has become indispensable for industrial, medical and of course daily life applications. With the help of image processing, the quality of images can be improved by mathematical manipulation so that they are more useful to a human observer or a computer vision system. For example, two important classes of methods for image processing in connection with linear parabolic-based PDEs are linear diffusion and osmosis filtering. In particular, the latter technique in its linear form, closely related to its transport phenomenon in nature, provides a powerful tool for visual computing applications such as image cloning and shadow removal. Based on the evolving camera technology, the resolution of images tends to increase continually. Actually, conventional digital compact cameras have a standard maximum resolution of more than 12 megapixels. Therefore, implicit methods have to handle large images for which the matrix size easily exceeds the order of several millions. As a consequence, the computational costs of the implicit scheme used become very expensive. For this purpose, there is still the need for a numerical scheme that combines accuracy and reasonable computational efficiency for working with high resolutions.

1.1.1 Modern Numerical Schemes

As indicated in the applications above, while implicit schemes have good numerical stability properties, they may cause high computational costs in various practical real-world problems. Thus, enhanced and computationally more efficient numerical methods are needed, which bring the aspects of approximation accuracy and computational and storage complexity into balance. Of course, there are dozens of approaches such as higher order *Runge-Kutta (RK)* schemes, extrapolation methods, linear multi-step methods or splitting methods, see e.g. [119,138]. In the case of parabolic PDEs, however, only a modest order of accuracy is often required, which usually does not exceed the order of the spatial discretisation, and therefore low order methods are typically appropriate to yield accurate enough approximations of the actual solutions. This suggests that one may forego high accuracy in exchange for a faster CPU time.

Although many numerical methods have been developed, in the case of stiff linear ODE systems there is no class of methods that can be classified as superior, since the correct choice of the method used strongly depends on the underlying model problem. In this framework, there are two popular techniques that are frequently used in their respective scientific fields: *fast explicit methods* and *model reduction techniques*. However, a joint analysis and evaluation of these methods in the context of numerical issues as considered here is often overlooked. But exactly this is useful in finding the best numerical method that is specifically designed for the intended purpose. Let us give a short insight into both classes of methods.

Fast Explicit Methods In order to achieve a stable scheme, the stability function of the numerical scheme used must be bounded in unity. The stability function depends on the product of the time step size and the eigenvalues of the underlying discrete Laplacian. It is well known that for stiff problems the eigenvalues are very large in absolute value, so that the associated time step size of classical explicit schemes must be very small. This relationship is declared as the stability region, whereby this region is generally small for traditional explicit methods. Consequently, explicit schemes are normally not applied for stiff problems due to their stability restrictions.

In many situations, the eigenvalues of the discrete Laplacian related to parabolic-based problems are known to be in a long narrow strip along the negative real axis. In this type of problems, a special class of multi-stage RK methods is a very powerful tool, in which those methods are constructed such that they have regions of stability extended along the real negative axis. In contrast to standard RK methods, the modified construction is based on increasing the number of stages so that the stability region is as large as possible, rather than to increase the order of accuracy of the method. More simply, some of the stages are intended to meet the conditions of consistency, whereas the rest strive to extend the region of stability along a strip near the negative real axis as much as possible. In this way, the stability region increases quadratically with the number of stages which is very attractive and makes these methods suitable for problems with large negative real eigenvalues. Since the stability function of this class of methods is related to Chebyshev polynomials, these methods are called RK-Chebyshev methods [138, 281, 286].

Overall, these methods are totally explicit with low memory demand and therefore do not require the solution of large sparse system of linear equations, which is potentially expensive. Based on their explicit nature, they are easy to implement and remain well-suited for parallelisation on GPUs. Moreover, they can easily be applied to large problem classes such as nonlinear problems, assuming those problems have negative real eigenvalues or they are near to the negative real axis and the underlying discrete Laplacian is close to be normal. More precisely, stability and convergence are only guaranteed for symmetric negative semi-definite matrices, but in general these methods remain well applicable to general (nonlinear) parabolic problems for which the system matrix does not deviate too much from a normal matrix. Even if these requirements are satisfied, the modified RK schemes are only useful for modestly stiff problems, since for extremely large eigenvalues in magnitude a very large number of stages is required to achieve stability. Another weakness of the fast explicit methods is that the positivity property is not ensured when performing the time integration, even if the continuous and semi-discrete model problems provide this property. This of course could be undesirable, for example, in image processing tasks for which images are generally defined as functions with nonnegative range.

Model Reduction Techniques Another modern alternative to improve the performance of the integration approach related to (non)linear large-scale semi-discrete ODE systems is based on model reduction. Such techniques can be used to approximate the large-scale *dynamical system* by a reduced, low dimensional dynamical system, for which the main characteristics of the original one are essentially entirely preserved. In other words, model reduction aims to reduce the computational complexity by reducing the number of describing equations of a large dynamical system. An approximation to the original model is commonly referred to as a reduced order model. At this point it should be mentioned that dynamical systems are nothing else than the evolution of systems in time, in which continuous time systems are typically modelled by ODEs or PDEs.

The reduction of linear dynamical systems has been extensively studied in the last decades and is still considered nowadays as an important topic due to its wide range of applications in physics, mathematics and engineering. This follows directly from the increasing ability of methods and computers to accurately model real-world problems, but also to be able to use these generated models in an efficient way for the intended purpose of e.g. simulation,

optimisation and control. Although reduction methods have proven to be powerful tools for a wide variety of applications, not everyone is familiar with these techniques. Apart from that, a very large number of different methods and algorithms have been developed over time, each of them has its advantages and disadvantages. For this reason, the efficient use of model reduction techniques is not straightforward.

In general, existing reduction techniques can be classified into balancing based methods and moment matching methods. This classification is not always consistent in the literature, for a general overview see [10, 11, 193, 248]. Reduction techniques can enable to significantly speed up ODE integration. More precisely, the reduction procedure is understood as a preprocessing step, so that the reduced order model can be solved much more easily by time integration methods. Nevertheless, this proceeding should provide several requirements on a reduced model such as a good approximation quality of the original model, conservation of stability and characteristic behaviour, but also computational efficiency and robustness. In fact, ODE systems involving negative semi-definite matrices, as is normally the case with parabolic problems, guarantee preservation of stability in one-sided projection.

In particular, the model reduction process is divided into two main stages: the *offline* and the *online* phase. First, the linear dynamical system is reduced independently by standard techniques in an offline stage, then the reduced model is solved by a numerical method in an online step. Obviously, model reduction pays off, if the benefit of multiple cheap online evaluations outweighs the offline up-front costs required for computing the reduced model. It is therefore desirable to avoid intensive offline and online computations so that the reduction step is numerically justified. For a better understanding of this procedure, a visualisation is shown in Figure 1.2. It should be noted that explicit and implicit schemes can also cause offline costs, e.g. by transforming computations from CPU to GPU or by using factorisation and preconditioning techniques. In contrast to model reduction, however, these computational costs are low and can be neglected.

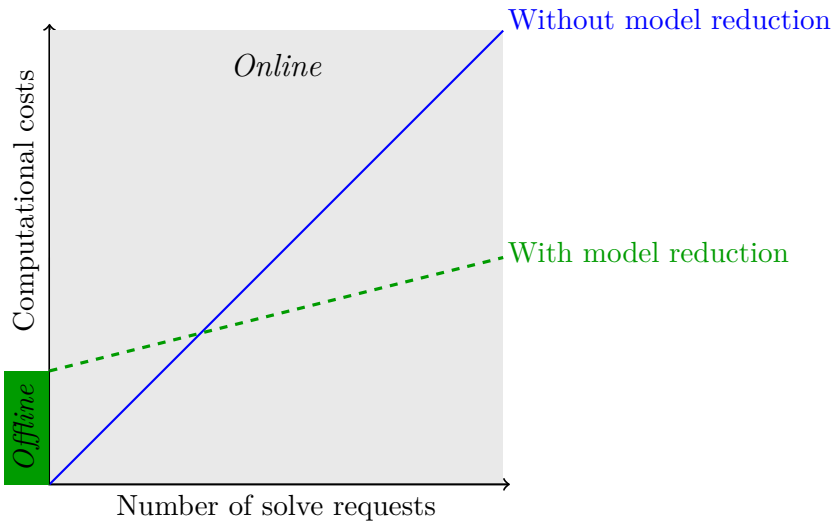


Figure 1.2: Computational costs in terms of offline and online computations using model reduction techniques.

1.1.2 Main Requirements for an Efficient Method

In total, there are many options to numerically solve linear parabolic-based model problems in an efficient manner. A general overview can be found in Figure 1.3. Among the many techniques available such as implicit methods, fast explicit methods and reduction techniques, there is no overall best method. Model problems usually have different model-dependent boundary conditions and therefore often require different approaches that specifically fit into the intended framework. In particular, an efficient solver should provide the following features that are important in practical use:

- good approximation quality
- preservation of system properties (stability, positivity, conservation laws)
- low computational costs and memory requirements
- numerically robust algorithm independent of data, size, parameters
- simple implementation and easy parallelisation
- small number of user-defined parameters

1.2 Contributions and Outline

As mentioned above, when dealing with large sparse linear semi-discretised ODE systems resulting from the spatial discretisation of linear parabolic multi-dimensional PDEs by the MOL approach, many methods for the numerical solution are available. However, the underlying model problems considered here are coupled with different settings, ranging from many different initial conditions over long-term simulation with relatively frequent model updates to dealing with very large-scale problems for which the matrix size can exceed several millions. Having a wide variety of solvers is a great advantage, but makes the decision more difficult to use the correct numerical scheme for the model problem that has to be solved. In simple terms, even nowadays it is still a challenging task to devise a method that combines high accuracy, robustness and computational efficiency. Therefore, the main objective is to find an easy and efficient ODE integration scheme for each individual problem.

The first goal of the thesis is to present a comprehensive overview and introduction to the state-of-the-art methods that are frequently used, but only often employed in their respective scientific fields. In this context, we will study the theoretical and numerical basics of implicit methods, fast explicit methods and reduction techniques. This is primarily important in order to fully understand the strengths and weaknesses of each technique. Furthermore, we clarify in detail the problems that arise in practice in connection with error bounds, convergence, stability, accuracy, implementation, parameters involved or other degrees of freedom. Besides compiling the knowledge of various methods that can be used to solve the problems dealt with in this thesis, we highlight why these numerical schemes need to be treated carefully and in which cases their practical application is challenging.

With the help of this, our second goal is to efficiently solve the mentioned problems of shape correspondence, geothermal energy storage and osmosis filtering. For each application we provide a complete setup that covers the problem formulation, the related work, the general framework, the numerical discretisation as well as the computational challenges and

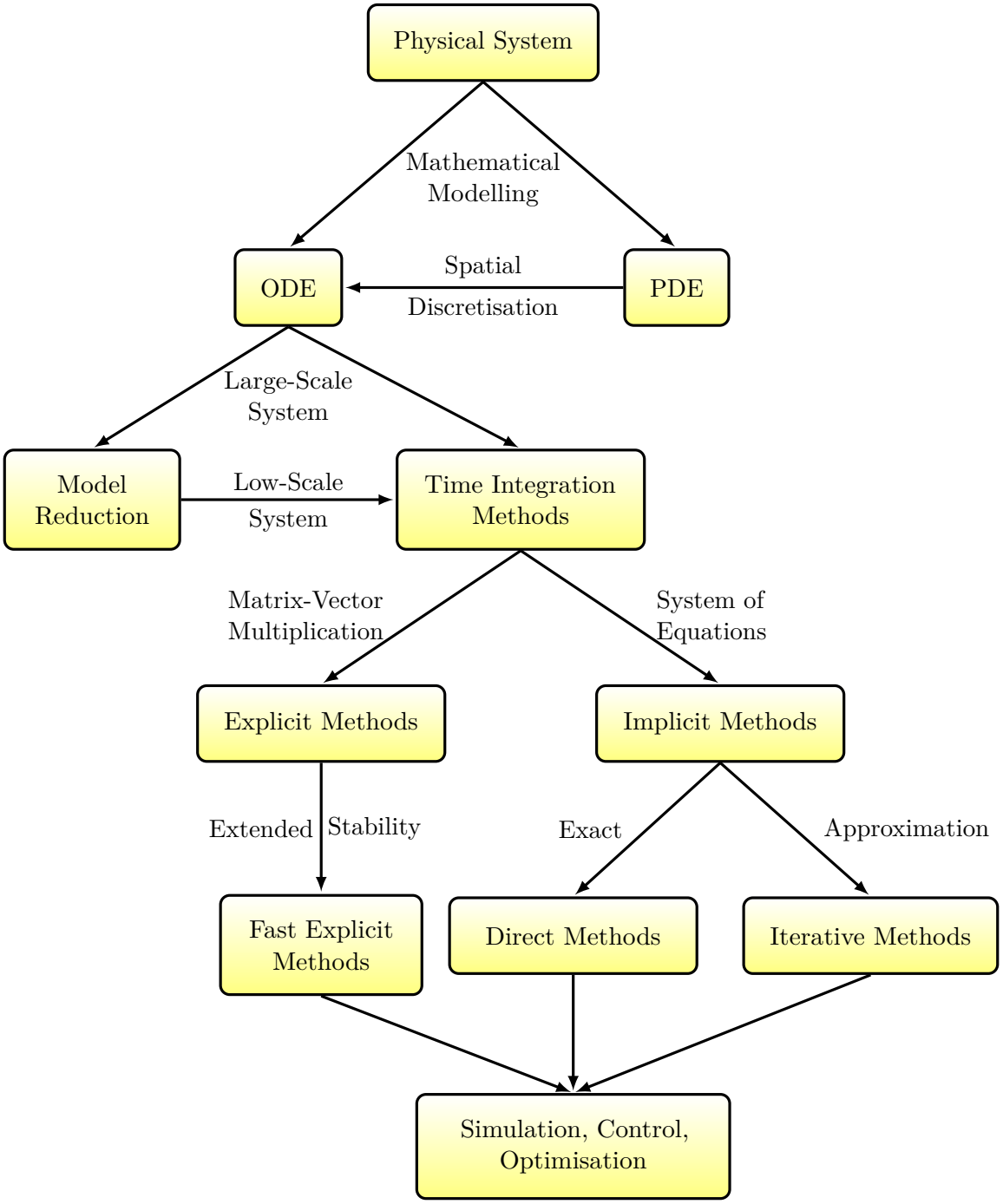


Figure 1.3: General structure for the numerical solution of PDEs or ODEs using time integration methods. In connection with the solution of (linear) parabolic model problems, we distinguish between implicit methods, fast explicit methods and model reduction techniques.

settings. In order to identify the best method for each model problem, we perform a thorough numerical study of all the solvers mentioned, but also compare them to those which have been alternatively proposed in the literature. Apart from that, we will also present several substantial improvements of the integration based approaches and some novel numerical strategies, so that a more accurate or efficient computational approach is achieved in contrast to the original use.

Let us mention that we refer the reader to the beginning of Chapters 5-7 for a more detailed description of our technical contributions to the practical problems discussed.

Outline of the Thesis This introductory chapter concludes with an overview of the different chapters of this dissertation. In general, this thesis is organised into two main parts, i.e. Chapters 2, 3 and 4 present the theoretical and numerical knowledge of the methods used, while Chapters 5, 6 and 7 covering the main contributions of the author.

The next chapter introduces the relevant basics of numerical analysis for the reader unfamiliar with the topics, including time integration methods, consistency and stability properties, explicit and implicit methods, sparse direct and sparse iterative methods, and matrix exponential approximations.

The third chapter provides a complete overview of the fast explicit solvers. The framework of this class of methods together with all the theoretical and numerical aspects is presented. Moreover, we will elaborate that there are two ways of designing such methods that conceptually belong to the class of RK methods, namely by direct and indirect approach. The former ansatz is widespread in the classical numerical community, whereas the latter technique is typically known in image processing. In addition to the description of the differences, a section is dedicated to give a numerical comparison of both techniques using two experiments based on linear and nonlinear PDE-based image diffusion.

In Chapter 4, as an alternative to considering the original large-scale ODE system, the model reduction techniques based on projection methods are introduced so that the high dimensional semi-discretised problem is replaced by one of reduced order in a suitable manner. In this way, the computational complexity can be reduced, since the projection methods generally rely on efficient numerical linear algebra techniques. In this sense, we focus on four commonly used and well-studied methods that are usually the best choices for solving real-world problems. Apart from providing the foundations and describing the necessity and usefulness of model reduction, we clarify how these methods are to be applied, but also indicate their limitations in practice.

The first practical problem in this thesis is discussed in Chapter 5, in which we investigate shape correspondences between three-dimensional shapes that rely on feature descriptors. An interesting class of models for such descriptors relies on simple linear PDEs based on the Laplace-Beltrami operator. Besides a variety of PDEs, also several ways to solve them have been considered in related works. In this chapter, we show how to define a computational framework, which can be distinguished as spectrum-free and spectrum-based computation, by using model reduction techniques that yield an efficient ODE integration and much more accurate shape signatures as in previous works. Furthermore, as part of the construction of our framework we elaborate several substantial details of the applied technique which are necessary in order to enhance the usability. In addition, we highlight the similarities and the differences to frequently used spectral methods that are derived from the series expansion of

analytic solutions of the PDEs. Through our experiments, we present that the numerical signatures obtained by time integration methods of the underlying PDEs lead to significant improvements over state-of-the-art-methods for finding correct shape correspondences.

In Chapter 6 we are tackling the long-term simulation of the recent geothermal energy storage technology that represents a potentially very attractive approach to energy storage. We will precisely elaborate the complete continuous model and the corresponding discretisation which is an important component for the numerical realisation. On this basis, we discuss which methods are suitable for long-term simulations, and show how to adapt some of the currently most efficient numerical approaches to the fundamental problem of heat evolution with internal and external boundary conditions as well as source terms. In order to provide an efficient and accurate enough simulation, we give a thorough discussion of the various numerical solvers along with many technical details and own adaptations. More precisely, we propose a heuristic procedure to resolve the essential problem of a large number of inputs in connection with the Krylov-based model order reduction, which contains some technical novelties. This is crucial for the application of model reduction techniques for the practical use related to geothermal energy storage. In particular, we present that we can obtain fast and accurate long-term simulations of typical geothermal energy storage facilities. We validate our numerical findings using synthetic and real-world data.

In Chapter 7 we consider linear osmosis filtering. We first give a description of the theoretical background from both continuous and discrete setting, before discussing the recent efficient numerical implementation based on standard splitting techniques. However, such methods suffer from splitting errors which have a strong influence on the accuracy of the approximation. Therefore, we will propose to use fast explicit methods for osmosis-based image processing tasks, as these combine accuracy and computational efficiency. Although this class of methods cannot guarantee the preservation of the natural osmotic properties and the numerical stability from a theoretical point of view, we show that these methods are well applicable for isotropic and also anisotropic osmosis filtering, and experimentally verify this by our numerical tests. To evaluate the performance, we conduct a thorough numerical study using various image processing applications.

The dissertation completes in Chapter 8 with a summary of the results, and we also give an outlook on some possible future works.

Chapter 2

Classic Numerical Methods

This chapter is intended for a non-expert audience as a guide towards the mathematical background for the numerical solution of time-continuous ODE systems that typically arise when following the MOL approach by discretising parabolic-type PDEs first in space variables and leaving the time variable continuous. The aim is to provide the reader with some basic notions that are essential when tackling large sparse linear ODE systems and to describe general challenges in using standard numerical methods for initial value problems. In Section 2.1 we introduce the time integration methods and the relevant criteria that are crucial for the performance of a numerical algorithm such as consistency, stability and convergence. Since the underlying linear ODE systems are known to be stiff, in Section 2.2 we provide information about implicit schemes, more precisely on sparse direct and sparse iterative methods, which are often considered due to their good stability properties. In Section 2.3 we present the exponential integrators as an alternative class of numerical methods for solving stiff ODE systems.

2.1 Time Integration Methods

As already described in the introductory chapter, the MOL approach for solving PDEs leads to a semi-discretised ODE system so that any numerical method for initial value problem can be applied. When considering linear parabolic PDEs in the context of heat conduction, the linear semi-discretised system can basically be expressed in the form of

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + \boldsymbol{w}(t), \quad t \in (0, t_F],\, \boldsymbol{u}(0) = \boldsymbol{u}^0,\, \boldsymbol{w}(0) = \boldsymbol{w}^0 \tag{2.1}$$

with a large sparse Laplacian matrix $L \in \mathbb{R}^{n\times n}$, the unknown solution vector $\boldsymbol{u}(t) \in \mathbb{R}^n$, a vector $\boldsymbol{w}(t) \in \mathbb{R}^n$ representing boundary conditions and/or source terms and given initial data $\boldsymbol{u}^0, \boldsymbol{w}^0 \in \mathbb{R}^n$. The most frequently used class for solving a system of ODEs are time stepping methods, in which the time variable in (2.1) is discretised by $0 = t_0 < t_1 < \cdots < t_J = t_F$. Discrete time stepping methods of ODEs can be done using standard numerical integration so-called *time integration methods*. Common time integration schemes are the *explicit Euler (EE)* method, the *implicit Euler (IE)* method and the *Crank-Nicolson (CN)* method. In general, all of these schemes belong to the well-known class of RK methods, which we will discuss later. In the following let us describe the classic time integration methods and discuss important properties such as consistency and numerical stability. For a complete and excellent overview of time integration methods, we refer to [119, 120, 138, 161, 189, 264, 272].

To apply time discretisation methods, time intervals $I_k = [t_k, t_{k+1}]$ are defined in order to subdivide the complete integration time $[0, t_F]$ into a partition. The resulting numerical

methods then generate approximations $\boldsymbol{u}(t_k)$ at the different time levels t_k. Uniform partitions of time intervals are often used, as is also the case in this thesis. Let us first derive the classic EE, IE and CN schemes, afterwards we will introduce in some detail important criteria for the analysis of numerical methods. For the sake of simplicity, we set $\boldsymbol{w}(t) = \mathbf{0}$ and consider the semi-discretised system

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad t \in (0, t_F],\, \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{2.2}$$

Explicit Euler Method As a first step, the application of the fundamental theorem of calculus for (2.2) over the time interval I_k yields

$$\int_{t_k}^{t_{k+1}} \dot{\boldsymbol{u}}(t)\,\mathrm{d}t = \int_{t_k}^{t_{k+1}} L\boldsymbol{u}(t)\,\mathrm{d}t \tag{2.3}$$

Assuming that $\dot{\boldsymbol{u}}$ is integrable on I_k, the left-hand side of (2.3) obviously results in

$$\int_{t_k}^{t_{k+1}} \dot{\boldsymbol{u}}(t)\,\mathrm{d}t = \boldsymbol{u}(t_{k+1}) - \boldsymbol{u}(t_k) \tag{2.4}$$

Second, the approximation of the integral on the right-hand side of (2.3) using the left-hand rectangle method gives

$$\int_{t_k}^{t_{k+1}} L\boldsymbol{u}(t)\,\mathrm{d}t \approx \tau L\boldsymbol{u}(t_k) \tag{2.5}$$

where $\tau = t_{k+1} - t_k$ is the uniform time step size. Finally, using the notation $\boldsymbol{u}(t_k) = \boldsymbol{u}^k$, the well-known fully discrete EE method

$$\boldsymbol{u}^{k+1} = (I + \tau L)\,\boldsymbol{u}^k \tag{2.6}$$

with $k \in \{0, \ldots, J-1\}$, the identity matrix I and the given initial condition $\boldsymbol{u}^0 = \boldsymbol{u}(0)$ is obtained. Due to the fact that the values $\boldsymbol{u}^k$ at time t_k are known, the new values $\boldsymbol{u}^{k+1}$ at time t_{k+1} can easily be computed by simple sparse matrix-vector multiplication. Schemes in this form are known as *explicit methods* and are well-suited for parallel computing like GPUs. However, it is known that explicit methods are only conditionally stable. In other words, the stability requirement leads to a severe limitation in the size of the time step τ. In general, the typical time step size restriction has a rather small upper bound in the case of stiff ODE systems. As a consequence, explicit schemes are usually considered to be extremely inefficient numerical methods from a computational point of view.

Implicit Euler Method In an analogous manner, the integral on the right-hand side of (2.3) can be approximated using the right-hand rectangle method via

$$\int_{t_k}^{t_{k+1}} L\boldsymbol{u}(t)\,\mathrm{d}t \approx \tau L\boldsymbol{u}(t_{k+1}) \tag{2.7}$$

so that the numerical method obtained is known as the fully discrete IE method

$$(I - \tau L)\, \boldsymbol{u}^{k+1} = \boldsymbol{u}^k \tag{2.8}$$

In order to compute the values $\boldsymbol{u}^{k+1}$ at time t_{k+1}, a large sparse system of linear equations has to be solved for each time step. Such schemes are called *implicit methods*. Consequently, implicit schemes are numerically more intensive than explicit methods when choosing the same time step size. However, the significant advantage of an implicit scheme is that most of them theoretically result in an unconditionally stable scheme without a time step size restriction on τ. For this reason, implicit methods are typically applied when dealing with stiff ODE systems. Nevertheless, solving large sparse linear systems requires an efficient solver to be effective in practice. We will discuss this issue in more detail in Section 2.2.

Crank-Nicolson Method Using the trapezoidal rule for the integral approximation of the right-hand side of (2.3) gives

$$\int_{t_k}^{t_{k+1}} L\boldsymbol{u}(t)\,\mathrm{d}t \approx \frac{\tau}{2}\left(L\boldsymbol{u}(t_{k+1}) + L\boldsymbol{u}(t_k)\right) \tag{2.9}$$

which leads to the popular fully discrete CN scheme

$$\left(I - \tfrac{\tau}{2}L\right)\boldsymbol{u}^{k+1} = \left(I + \tfrac{\tau}{2}L\right)\boldsymbol{u}^k \tag{2.10}$$

The CN method, like the previous IE scheme, is an unconditionally stable implicit method. In order to obtain the values $\boldsymbol{u}^{k+1}$ at time t_{k+1}, it requires the solution of a system of linear equations as well as a sparse matrix-vector multiplication in each time step. Usually, CN is preferably used because of its second order convergence in time and at the same time only marginally higher computational costs compared to IE. At this point it should be noted that the CN method is sensitive to problems with discontinuous initial conditions and can lead to undesirable oscillations in the numerical solution.

Let us emphasise that there are several ways to derive the numerical methods described above. Besides the numerical integration, the construction principle can also be based on numerical differentiation or truncated Taylor series expansion. Apart from that, the introduced schemes belong to the class of *one-step* methods, since the numerical solution at the current time step only refers to the previous solution. Later, we will consider RK methods, where intermediate solutions are used to obtain higher order or accelerated methods.

Properties of Numerical Methods When solving PDEs numerically using time integration methods several characteristics are crucial for the practical use of a numerical scheme. The most important criteria are *consistency*, *stability* and *convergence*. In particular, consistency is the condition that the numerical scheme converges towards the continuous differential equation as the grid size h and the time step size τ tend to zero and the truncation errors vanish. The stability of a numerical scheme is the property that all accumulated numerical errors (truncation, round-off, noisy initial conditions) generated during the numerical process remain bounded and should not be magnified. Convergence is the condition that the numerical

solution converges towards the exact solution of the PDE as the discretisation parameters h, τ tend to zero. The standard way to prove the convergence of a numerical approximation is generally shown using the Lax equivalence theorem which states that a consistent method is convergent if and only if it is stable. This framework is often expressed by the statement:

$$\text{consistency} \;+\; \text{stability} \;\Longrightarrow\; \text{convergence}$$

In addition, two other useful properties such as *conservation* and *boundedness* are of interest as well. The conversation property reflects that the underlying conservation laws should be preserved at the discrete level. The boundedness means that physical quantities like densities, temperatures or concentrations should remain nonnegative and oscillation-free.

In the following we provide a more detailed insight into the criteria consistency and stability. Let us stress that there are several alternative definitions in the field of numerical analysis, the choice of definition one prefers to use is just a matter of taste. Since our starting point is the time-dependent ODE system (2.2), we assume that the spatial discretisation of the continuous Laplace operator is consistent. Of course, the following investigations can be applied analogously to problems in the form of

$$\dot{\boldsymbol{u}}(t) = \boldsymbol{f}\left(t, \boldsymbol{u}(t)\right), \quad t \in (0, t_F],\, \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{2.11}$$

where $\boldsymbol{f}$ is a general right-hand side function.

2.1.1 Consistency

As stated above, consistency means that the discretised equation of an ODE should become exact as the time step size tends to zero. Checking consistency is usually simple by using the Taylor expansion and comparing the local error by which the exact solution misses satisfying the numerical scheme. More precisely, consistency is a local property of a numerical scheme. Let us explain this by investigating the EE method.

For consistency, the *local error* is defined as

$$\boldsymbol{E}^n = \boldsymbol{u}(t_{n+1}) - \widetilde{\boldsymbol{u}}(t_{n+1}), \quad n = 0, \ldots, J-1 \tag{2.12}$$

where $\boldsymbol{u}(t_{n+1})$ and $\widetilde{\boldsymbol{u}}(t_{n+1})$ denote the exact solution and the approximate solution, respectively. In particular, the approximation is exemplarily computed here by the EE method

$$\widetilde{\boldsymbol{u}}(t_{n+1}) = \boldsymbol{u}(t_n) + \tau L\boldsymbol{u}(t_n) \tag{2.13}$$

in just one-step using the exact solution at time t_n. In other words, the local error is the error after one time step when starting with the exact solution within the numerical scheme. A one-step method is said to be *consistent* if

$$\max_{0 \le n \le J-1} \|\boldsymbol{E}^n\| \to 0 \quad \text{for} \quad \tau \to 0 \tag{2.14}$$

is satisfied. Moreover, the consistency with order p of an one-step method is defined by

$$\max_{0 \le n \le J-1} \|\boldsymbol{E}^n\| = \mathcal{O}\Big(\tau^{p+1}\Big) \tag{2.15}$$

To analyse the EE method, the Taylor expansion of $\boldsymbol{u}$ (assuming that $\boldsymbol{u}$ is smooth) is performed around any point in $[0, t_F]$. For convenience, expanding around t_n gives

$$\begin{aligned}\boldsymbol{E}^n &= \boldsymbol{u}(t_{n+1}) - \widetilde{\boldsymbol{u}}(t_{n+1}) \\ &= \left[\boldsymbol{u}(t_n) + \tau\dot{\boldsymbol{u}}(t_n) + \tfrac{\tau^2}{2}\ddot{\boldsymbol{u}}(\boldsymbol{\xi}_n)\right] - \boldsymbol{u}(t_n) - \tau L\boldsymbol{u}(t_n) = \tfrac{\tau^2}{2}\ddot{\boldsymbol{u}}(\boldsymbol{\xi}_n)\end{aligned} \tag{2.16}$$

with an appropriate vector $\boldsymbol{\xi}_n \in (t_n, t_{n+1})$. The comparison with (2.15) clearly shows that the EE method has the consistency order one. In an analogous manner, it can be shown that the IE method and the CN method are of first and second order, respectively.

Obviously, a global statement about the difference between the exact and the approximate solution after a sequence of time steps is also of interest. It can be shown that (under smoothness conditions) the convergence of a one-step method follows directly from its consistency. In addition, the order of convergence is the same as the order of consistency. We will not go into the convergence analysis of a numerical scheme any further detail. However, unstable numerical schemes are not convergent. Because of this, the study of numerical stability is extremely important.

2.1.2 Numerical Stability

As mentioned, global convergence relies on two important properties: consistency and stability. A stable numerical scheme possesses the characteristic that numerical errors (round-off, truncation) do not grow as the calculation proceeds. In simple terms, small local errors should only lead to small global errors. More precisely, a numerical method is said to be stable if all numerical errors remain bounded for $\tau \to 0$. The stability of a numerical scheme is generally the most important property and usually the most difficult to verify. Let us explain the difficulties involved in dealing with numerical stability. A fundamental tool in stability analysis is the use of the *Dahlquist test equation* (2.17), which is used in various parts in this thesis.

Scalar Test Problem We consider the scalar ODE in the form of

$$u'(t) = \lambda u(t), \quad \lambda \in \mathbb{C},\, u(0) = u^0 \tag{2.17}$$

where the analytical solution is given by $u(t) = e^{\lambda t}u^0$. Let $\widetilde{u}(0) = u^0 + \delta$ be a slightly perturbed initial condition, then the solution of the perturbed problem is $\widetilde{u}(t) = e^{\lambda t}u^0 + e^{\lambda t}\delta$. Thus, for $\mathrm{Re}(\lambda) > 0$ the original problem (2.17) is an unstable problem[1], since it holds that

$$\left|u(t) - \widetilde{u}(t)\right| = |e^{\lambda t}\delta| \tag{2.18}$$

which becomes arbitrarily large for each $\delta \neq 0$ if t is sufficiently large. In contrast, for $\mathrm{Re}(\lambda) \leq 0$, a small change in the initial condition causes only a small change in the solution and therefore the problem (2.17) is called a *stable problem*. For this reason, stability investigations of numerical schemes are only of interest for the case $\mathrm{Re}(\lambda) \leq 0$. An ODE with a strictly negative λ is said to be *asymptotically stable*, with the solution converging to the zero equilibrium point as $t \to \infty$ for any initial state.

[1] A problem (2.17) is stable if, with small changes in the initial conditions, only small changes in the solution occur.

Based on this consideration, let us examine the EE method. Applying the EE scheme to (2.17) results in

$$u_{n+1} = u_n + \tau\lambda u_n = (1 + \tau\lambda)u_n =: R(z)u_n \tag{2.19}$$

The numerical behaviour of EE relies on

$$R(z) := 1 + z, \quad z = \tau\lambda \tag{2.20}$$

where $R(z)$ is called the *stability function*. Assuming $\mathrm{Re}(\lambda) < 0$, the analytical solution is an exponentially decaying function, so a stable numerical scheme should exhibit the same behaviour. This means that the approximations $\{u_n\}$ should yield a decreasing sequence with respect to $|\cdot|$, expressed as

$$|u_{n+1}| < |u_n|, \quad n = 0, \ldots, J-1 \tag{2.21}$$

In order to mimic this behaviour, the EE method (2.19) must satisfy

$$|u_{n+1}| < |R(z)||u_n| \quad \Longrightarrow \quad |R(z)| < 1 \tag{2.22}$$

with $z = \tau\lambda$. For $\mathrm{Re}(\lambda) \leq 0$, the latter observation is obviously tantamount to $|R(z)| \leq 1$. Consequently, for real $\lambda \leq 0$ the condition is equivalent to

$$-2 \leq \tau\lambda \leq 0 \quad \Longleftrightarrow \quad \tau \leq \frac{-2}{\lambda} =: \tau_{\max} \tag{2.23}$$

where $\tau_{\max}$ denotes the maximum stable time step size for the EE method. By nature is $\tau > 0$, so that with real $\lambda \leq 0$ the interval $z \in [-2, 0]$ is declared as the *interval of absolute stability* for the EE method. In the case of $\mathrm{Re}(\lambda) \leq 0$, stability is guaranteed if the complex number $\tau\lambda$ lies in the nonpositive complex number plane inside the disk of radius one centred at the point $(-1, 0)$. This shows that the EE method is only *conditionally stable*, since the time step size τ has to be chosen sufficiently small depending on λ to ensure stability. For the complex case, the investigation leads to the definition of the *stability region* (stability domain) or the so-called *region of absolute stability*, for which the set $z = \tau\lambda$ fulfils the stability condition in relation to the EE scheme via

$$\mathcal{S}_{EE} = \Big\{ z = \tau\lambda \in \mathbb{C} : \ |R(z)| = |1 + z| \leq 1 \Big\} \tag{2.24}$$

This proceeding can be done for both the IE method and the CN method as well. Applying the IE scheme to (2.17) gives

$$(1 - \tau\lambda)u_{n+1} = u_n \quad \Longleftrightarrow \quad u_{n+1} = (1 - \tau\lambda)^{-1}u_n =: R(z)u_n \tag{2.25}$$

with $R(z) = (1 - \tau\lambda)^{-1}$, which implies

$$|R(z)| \leq 1 \quad \Longleftrightarrow \quad \tau\lambda \leq 0 \tag{2.26}$$

The region of absolute stability is thus defined by

$$\mathcal{S}_{IE} = \Big\{ z = \tau\lambda \in \mathbb{C} : \ |R(z)| = \Big|(1 - z)^{-1}\Big| \leq 1 \Big\} \tag{2.27}$$

and the IE scheme is stable for the exterior of the open unit disk in the complex plane centred at $(1, 0)$. Finally, using the CN method it follows that

$$\frac{\left|1+\frac{\tau\lambda}{2}\right|}{\left|1-\frac{\tau\lambda}{2}\right|} = \frac{|2+z|}{|2-z|} = \big|R(z)\big| \leq 1 \quad \Longleftrightarrow \quad \tau\lambda \leq 0 \tag{2.28}$$

and the stability region is specified by

$$\mathcal{S}_{CN} = \left\{ z = \tau\lambda \in \mathbb{C} : \ \big|R(z)\big| = \frac{|2+z|}{|2-z|} \leq 1 \right\} \tag{2.29}$$

As a result, both implicit methods are called *unconditionally stable*, since the time step size τ can be arbitrarily large. For the sake of completeness, the stability regions of all methods are illustrated in Figure 2.1.

It should be noted that the consistency order can also be derived by analysing the stability function of the numerical method used. This can be explained as follows: the EE method can be written in the form

$$E^n := u(t_{n+1}) - R(z)u(t_n) \tag{2.30}$$

with $R(z) = 1 + z$, $z = \tau\lambda$ and where E^n denotes the consistency error. Using the analytical solution $u(t) = e^{\lambda t}$, the discrete setting $u(t_{n+1}) = e^{\tau\lambda}u(t_n) = e^z u(t_n)$ with $t_{n+1} = t_n + \tau$ implies that

$$E^n = \left[e^z - R(z)\right] u(t_n) \tag{2.31}$$

The Taylor series expansion of the exponential function reads

$$e^z = 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \mathcal{O}\Big(z^4\Big) \tag{2.32}$$

so that (2.31) can be interpreted as

$$e^z - (1+z) = \mathcal{O}\Big(z^2\Big) \quad \text{as} \quad z \to 0 \ \ (\tau \to 0) \tag{2.33}$$

In particular, the EE method exactly matches the first two terms (1 and z) with the expansion of the exponential function in the exact solution and thus implies the consistency order of one. As a result, the desired order of accuracy can be achieved by ensuring that the leading terms of the stability function exactly match with the Taylor series expansion of the exponential function. In an analogous manner, the implicit schemes yield

$$(1-z)^{-1} = 1 + z + z^2 + \mathcal{O}\Big(z^3\Big) \tag{2.34}$$

$$\frac{2+z}{2-z} = 1 + z + \frac{z^2}{2} + \frac{z^3}{4} + \mathcal{O}\Big(z^4\Big) \tag{2.35}$$

which shows first and second order accuracy for the IE method (2.34) and the CN method (2.35), respectively.

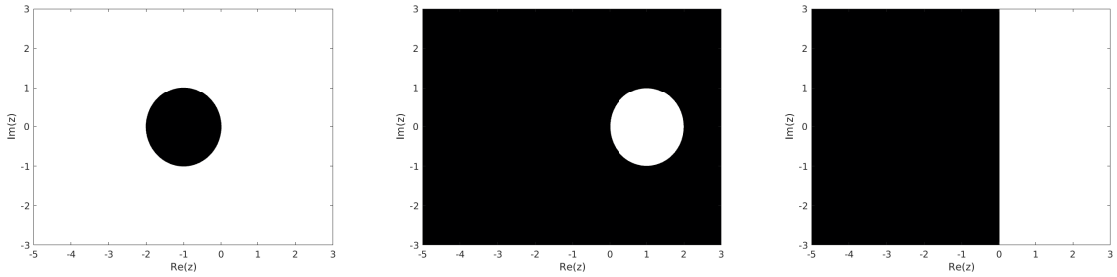


Figure 2.1: The stability region $\mathcal{S}$ of the EE, IE and CN methods. The black colour indicates the region of absolute stability. **Left:** Stability region $\mathcal{S}_{EE}$. **Middle:** Stability region $\mathcal{S}_{IE}$. **Right:** Stability region $\mathcal{S}_{CN}$.

Nonscalar Test Problem After examining the scalar case for exactly one ODE, we now study the numerical stability in relation to a system of ODEs

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{2.36}$$

with matrix $L \in \mathbb{R}^{m\times m}$ and where the analytic solution is given by $\boldsymbol{u}(t) = e^{Lt}\boldsymbol{u}^0$. Assuming that L is diagonalisable (complete set of m linearly independent eigenvectors) with the eigenvector matrix $\Phi \in \mathbb{R}^{m\times m}$ and the diagonal matrix $\Lambda \in \mathbb{R}^{m\times m}$ of corresponding eigenvalues, the following identities hold:

$$L = \Phi\Lambda\Phi^{-1} \quad \text{and} \quad \Lambda = \Phi^{-1}L\Phi \tag{2.37}$$

The multiplication of Φ^{-1} from the left to (2.36) and the introduction of the identity matrix $I = \Phi\Phi^{-1}$ gives

$$\Phi^{-1}\dot{\boldsymbol{u}}(t) = \Phi^{-1}L\Phi\Phi^{-1}\boldsymbol{u}(t) \tag{2.38}$$

Using the identity (2.37) and setting $\boldsymbol{w}(t) = \Phi^{-1}\boldsymbol{u}(t)$, finally a decoupled ODE system with m independent scalar equations for each component of $\boldsymbol{w}$ is obtained in the form

$$\dot{\boldsymbol{w}}(t) = \Lambda\boldsymbol{w}(t) \tag{2.39}$$

Applying the EE method to (2.39) yields

$$\boldsymbol{w}^{n+1} = (I + \tau\Lambda)\boldsymbol{w}^n =: R(z)\boldsymbol{w}^n \tag{2.40}$$

which for each decoupled component takes the form

$$\boldsymbol{w}_q^{n+1} = (1 + \tau\lambda_q)\,\boldsymbol{w}_q^n, \quad q = 1, \ldots, m \tag{2.41}$$

To be a stable method, each scalar problem simultaneously must be stable. This is obviously fulfilled if $\lambda_q \leq 0$ and $\tau\lambda_q$ lie in the stability region $\mathcal{S}_{EE}$ for all q. Because of this, the maximum time step size is directly related to the largest eigenvalue $\lambda_{\max}$ in absolute value by

$$\tau \leq \frac{-2}{\lambda_{\max}} =: \tau_{\max} \tag{2.42}$$

This in turn illustrates that $\tau_{\max}$ can be extremely small if the underlying system matrix possesses very large negative eigenvalues. As a consequence, the EE method often causes high computational costs for stiff ODE systems and is generally unusable for practical purposes.

The relation between the largest eigenvalue (in magnitude) and (2.40) using $A := I + \tau\Lambda$ leads to the *necessary condition* for numerical stability

$$|R(z)| \leq 1 \quad \Longleftrightarrow \quad \rho(A) \leq 1 \tag{2.43}$$

where $\rho(A)$ is the *spectral radius* of A. The spectral radius is defined as the number

$$\rho(A) = \max\{|\lambda| : \lambda \in \varrho(A)\} \tag{2.44}$$

for which the set ϱ is called the *spectrum* of A, given by

$$\varrho(A) = \{\lambda : \lambda \text{ is eigenvalue of } A\} \tag{2.45}$$

In this context, when using the well-known von Neumann stability analysis (also known as Fourier stability analysis), which decomposes the errors into Fourier series, the corresponding factor $R(z) \to g(\xi)$ with a complex wave number ξ is called the amplification factor, and the stability condition reads $|g(\xi)| \leq 1$.

In contrast to EE, the IE and CN methods allow arbitrarily large time step sizes τ and are used very frequently in practice. Although both methods are unconditionally stable, τ must usually be chosen to be relatively small in order to achieve an accurate numerical solution. Therefore, a fast solver for large sparse systems of linear equations is necessary.

From the observations above, a linear dynamical system like (2.36) is called *asymptotically stable* if the real part of all eigenvalues λ_i of L are strictly negative, i.e. $\mathrm{Re}(\lambda_i) < 0$. In this case, any initial condition converges towards the zero equilibrium $\boldsymbol{u}(t) = \mathbf{0}$ as $t \to \infty$, when no input signal is applied. Furthermore, a linear system is said to be *stable* if the eigenvalues of L satisfy

(i) $\mathrm{Re}(\lambda_i) \leq 0, \forall i$

(ii) L has no defective eigenvalue with $\mathrm{Re}(\lambda) = 0$

The condition (ii) is satisfied e.g. when L is diagonalisable. It should be emphasised that for a stable system the equilibrium solution does not necessarily converge to the origin, since $\det(L) = 0$ implies that infinitely many solutions for (2.36) exists.

As stated above, the eigenvalues of the system matrix are generally very important for determining stability. However, the pure focus on the eigenvalues is not always sufficient to guarantee the numerical stability. A typical example of this is the advection equation, let us give a brief explanation.

Example 2.1. *We consider the one-dimensional linear advection equation*

$$\partial_t u(x,t) + a\partial_x u(x,t) = 0, \quad (x,t) \in \mathbb{R} \times (0, t_F] \tag{2.46}$$

with velocity $a > 0$, some given initial data $u(x,0) = f(x)$ and homogeneous Dirichlet boundary conditions. In particular, the positive velocity describes the corresponding flow from

left to right of the domain. Assuming that there is a finite number m of spatial grid points with a uniform grid width $h = \frac{1}{m+1}$ and that the spatial derivative is approximated by means of a backward difference scheme (first order upwind scheme), the MOL approach results in a semi-discretised system that is given by

$$\dot{\boldsymbol{u}}(t) = B\boldsymbol{u}(t), \quad t \in (0, t_F],\, \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{2.47}$$

with a nonsymmetric matrix

$$B = -\frac{a}{h}\begin{pmatrix} 1 & & & & \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \end{pmatrix} \tag{2.48}$$

Applying the EE method, the fully discrete scheme reads as

$$\boldsymbol{u}^{k+1} = (I + \tau B)\boldsymbol{u}^k \tag{2.49}$$

Although the system matrix $B \in \mathbb{R}^{m\times m}$ is nonsymmetric and a defective Jordan, all eigenvalues are real and given by the diagonal entries $\lambda_i = -\frac{a}{h}$ for $i = 1, \ldots, m$. If the abovementioned strategy for analysing stability is followed, the condition $\tau\lambda_q \in \mathcal{S}_{EE}$ is required for all eigenvalues of B. Therefore, the stability restriction leads to

$$\left|1 - \frac{\tau a}{h}\right| \leq 1 \quad \Longleftrightarrow \quad 0 \leq \frac{\tau a}{h} \leq 2 \tag{2.50}$$

and $\mathcal{S}_{EE}$ contains the interval $[-2, 0]$ on the real axis. Nonetheless, the latter condition is wrong, since the eigenvalue examination for stability is only necessary, but not sufficient. The problem arises from a critical property of the matrix B that is highly nonnormal. Based on this fact, it is necessary to analyse the pseudospectrum, which is closely related to the numerical abscissa. Using the pseudospectra, the stability requirement is given by

$$0 \leq \frac{\tau a}{h} \leq 1 \tag{2.51}$$

which is smaller than (2.50) by a factor two. In other words, unlike the spectral radius the pseudospectrum is robust to perturbations. The latter problem can be better understood if the Euclidean norm is considered. In doing so, (2.49) is rewritten in the form

$$\boldsymbol{u}^{k+1} = (I + \tau B)^{k+1}\boldsymbol{u}^0 = R(\tau A)^{k+1}\boldsymbol{u}^0 \tag{2.52}$$

with $A = I + \tau B$, so that the following norm is analysed in more detail:

$$\left\|\boldsymbol{u}^{k+1}\right\|_2 = \left\|R(\tau A)^{k+1}\boldsymbol{u}^0\right\|_2 \leq \left\|R(\tau A)^{k+1}\right\|_2 \left\|\boldsymbol{u}^0\right\|_2 \tag{2.53}$$

For $0 \leq \frac{\tau a}{h} \leq 2$ it holds that $|R(\tau\lambda)| \leq 1$ and this directly implies $\|R(\tau A)^{k+1}\|_2 \to 0$ as

$k \to \infty$. However, stability does not imply a monotonic decrease of the solution towards zero equilibrium. Consequently, the term $\|R(\tau A)^{k+1}\|_2$ becomes very large before it decays to zero, so that unacceptable error propagations produce unstable results. The transient growth in the solution is a consequence of the nonnormality of the matrix B.

We mention that the stability requirement (2.51) *can also be derived using the von Neumann stability analysis, in which the growth rate of an initial condition in terms of a wave is analysed.*

Even if the latter example has shown that the spectrum analysis with regard to highly nonnormal matrices is only necessary. The eigenvalue analysis remains very important when the underlying system matrix is normal or close to normal, which is the case in this thesis. Let us therefore provide the relation between the necessary and sufficient condition for stability.

Sufficient Condition for Stability Applying the EE method to (2.36) leads to

$$\boldsymbol{u}^{k+1} = (I + \tau L)\boldsymbol{u}^k = A\boldsymbol{u}^k \tag{2.54}$$

with $A = I + \tau L$, which can be rewritten by means of

$$\boldsymbol{u}^{k+1} = A\boldsymbol{u}^k = A^2\boldsymbol{u}^{k-1} = \cdots = A^{k+1}\boldsymbol{u}^0 \tag{2.55}$$

The numerical solution $\boldsymbol{u}^{k+1}$ should remain stable (bounded) with respect to a suitable norm for all (bounded) initial conditions $\boldsymbol{u}^0$, expressed as

$$\left\|\boldsymbol{u}^{k+1}\right\| \leq C \left\|\boldsymbol{u}^0\right\| \tag{2.56}$$

with a positive number C independent of k, h and τ. The latter is also declared by

$$\left\|\boldsymbol{u}^{k+1}\right\| \leq Ke^{\beta t_F} \left\|\boldsymbol{u}^0\right\| \tag{2.57}$$

with nonnegative constants K and β, meaning the numerical solution can grow as t_F increases, but not with the number of time steps used. From (2.55) it follows that

$$\left\|\boldsymbol{u}^{k+1}\right\| = \left\|A^{k+1}\boldsymbol{u}^0\right\| \leq \left\|A^{k+1}\right\| \left\|\boldsymbol{u}^0\right\| \tag{2.58}$$

so that the method is stable if the condition

$$\left\|A^{k+1}\right\| \leq C \tag{2.59}$$

is satisfied. Using norm properties one has

$$\left\|A^{k+1}\right\| = \left\|A^k A\right\| \leq \left\|A^k\right\| \|A\| \leq \cdots \leq \|A\|^{k+1} \tag{2.60}$$

and the Lax-Richtmyer definition of *sufficient stability* is given by

$$\|A\| \leq 1 \tag{2.61}$$

This condition is necessary and sufficient for the EE scheme to be stable in relation to the given system (2.36). In fact, (2.61) holds true for all numerical one-step methods, since schemes can be written as (2.55). For instance, the IE method takes this form with $A = (I - \tau L)^{-1}$.

Satisfying the condition (2.61) implies that $\rho(A) \leq 1$ is also sufficient for stability, since $\rho(A) \leq \|A\|$. Conversely, this is not valid because $\rho(A) \leq 1$ can still imply $\rho(A) \leq 1 < \|A\|$, which is often the case for nonnormal matrices. Fortunately, symmetric matrices possess the property $\rho(A) = \|A\|_2$, so that an eigenvalue analysis is sufficient to ensure stability in the Euclidean norm.

It should be noted that the latter derivation is also often discussed in the way that numerical errors (round-off errors) should not be allowed to grow unboundedly in the course of the computation. Let $\widetilde{\boldsymbol{u}}^0$ be a perturbation of the initial vector and $\boldsymbol{e} = \boldsymbol{u} - \widetilde{\boldsymbol{u}}$ denotes the error associated with this perturbation. The use of (2.55) results in

$$\boldsymbol{e}^{k+1} = \boldsymbol{u}^{k+1} - \widetilde{\boldsymbol{u}}^{k+1} = A^{k+1}(\boldsymbol{u}^0 - \widetilde{\boldsymbol{u}}^0) = A^{k+1}\boldsymbol{e}^0 \tag{2.62}$$

which is nothing else than the considerations already mentioned above, since

$$\left\|\boldsymbol{e}^{k+1}\right\| \leq \left\|A^{k+1}\right\| \left\|\boldsymbol{e}^0\right\| \tag{2.63}$$

Finally, let us mention that the stability criterion (2.61) is also true for ODE systems in the form (2.1). Applying the EE method leads to

$$\boldsymbol{u}^{k+1} = (I + \tau L)\boldsymbol{u}^k + \tau \boldsymbol{w}^k = A\boldsymbol{u}^k + \tau \boldsymbol{w}^k \tag{2.64}$$

which can be recursively represented by

$$\boldsymbol{u}^{k+1} = A^{k+1}\boldsymbol{u}^0 + A^k \tau \boldsymbol{w}^0 + A^{k-1}\tau \boldsymbol{w}^1 + \cdots + \tau \boldsymbol{w}^k \tag{2.65}$$

By using a perturbation $\widetilde{\boldsymbol{u}}^0$, the same stability property as above is obtained because of

$$\boldsymbol{e}^{k+1} = \boldsymbol{u}^{k+1} - \widetilde{\boldsymbol{u}}^{k+1} = A^{k+1}(\boldsymbol{u}^0 - \widetilde{\boldsymbol{u}}^0) = A^{k+1}\boldsymbol{e}^0 \tag{2.66}$$

Moreover, the stability criterion (2.61) can be weakened in some cases. For example, if the solution of the underlying PDE increases exponentially with increasing t, as for the equation

$$u_t = au_{xx} + bu \tag{2.67}$$

with positive constants a and b. Consequently, the necessary and sufficient condition for stability can be specified by

$$\|A\| \leq 1 + \tau K \tag{2.68}$$

with the positive number K independent of h and τ.

2.1.3 Stiff Problems

As we know, the EE method is only numerically stable if the time step size τ satisfies (2.42). However, a very large negative eigenvalue $\lambda_{\max}$ leads to an extremely small τ, so that a huge number of EE iterations are required to reach the final stopping time t_F. This in turn implies high computational costs and makes the EE scheme useless in practice. In contrast, implicit methods such as IE and CN are unconditionally stable and generally produces (qualitatively) correct approximations for all (reasonable) time step sizes τ. Although an efficient method

for solving very large sparse systems of linear equations is required, implicit schemes provide a drastic improvement over the EE method.

The latter issue is related to the *stiffness* that typically arises in many practical problems. Stiffness has in general no precise definition, but indicates a class of problems for which implicit methods often perform much better than explicit methods. Let us give a characterisation of the stiffness. We consider again the system (2.36) and assume that $L \in \mathbb{R}^{n\times n}$ is diagonalisable with the eigenvalues $\mathrm{Re}(\lambda_k) < 0$ and the corresponding eigenvectors $\boldsymbol{\phi}_k$ for $k = 1, \dots, n$. The analytical solution $\boldsymbol{u}(t) = e^{Lt}\boldsymbol{u}^0$ takes the form

$$\boldsymbol{u}(t) = \sum_{k=1}^{n} a_k e^{\lambda_k t}\boldsymbol{\phi}_k \tag{2.69}$$

where a_k is related to the initial condition. The property $\mathrm{Re}(\lambda) < 0$ implies that each component $e^{\lambda_k t}\boldsymbol{\phi}_k \to 0$ as $t \to 0$. For large $|\mathrm{Re}(\lambda_k)|$ the corresponding term $e^{\lambda_k t}\boldsymbol{\phi}_k$ will decay quickly, otherwise if $|\mathrm{Re}(\lambda_k)|$ is small the term $e^{\lambda_k t}\boldsymbol{\phi}_k$ decays slowly. On this basis, stiffness is often defined in terms of the *stiffness ratio*:

$$\frac{\max|\mathrm{Re}(\lambda)|}{\min|\mathrm{Re}(\lambda)|} \tag{2.70}$$

which is based on the fastest and slowest eigenvalue. Nevertheless, a large stiffness ratio does not directly imply that an ODE system is necessarily stiff. For instance, the eigenvalues of the matrices A and $\widetilde{A}$, which are exemplarily given by $\lambda_{max} = |-1000|$ and $\lambda_{min} = |-1|$ as well as $\widetilde{\lambda}_{max} = |-1|$ and $\widetilde{\lambda}_{min} = |-0.001|$, yield the same stiffness ratio of 1000. Although the stiffness ratio is large, the maximum EE time step size $\widetilde{\tau}_{max}$, which corresponds to the relatively small $\widetilde{\lambda}_{max}$, may be reasonable for practical purpose. For this reason, a more suitable characterisation of the stiffness is obtained from the following statement:

$$\text{An ODE system is stiff if } \max|\mathrm{Re}(\lambda)| \text{ is large, e.g. } |\mathrm{Re}(\lambda)| \gg 1$$

In other words, a problem is called *stiff* for which explicit methods require an extremely small time step size and are therefore extremely computationally slow, so implicit methods are preferred to be used. This characterisation also holds for the nonlinear case in which the eigenvalues of the Jacobian matrix $J = \partial \boldsymbol{f}/\partial \boldsymbol{u}$ are analysed.

Since the present thesis deals with parabolic-type PDEs, let us clarify that these model problems belong to the class of stiff problems.

Example 2.2. *As an example, we consider the one-dimensional linear heat equation*

$$\partial_t u(x,t) = \partial_{xx} u(x,t), \quad (x,t) \in \mathbb{R} \times (0, t_F] \tag{2.71}$$

with some given initial data $u(x,0) = g(x)$ and equipped with homogeneous Dirichlet conditions. Assuming a finite number n of spatial grid points with a uniform grid width $h = \frac{1}{n+1}$, and approximating the spatial derivative by means of central differences, the resulting semi-discretised system using the MOL approach reads

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad t \in (0, t_F],\ \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{2.72}$$

with a symmetric matrix

$$L = \frac{1}{h^2}\begin{pmatrix} -2 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -2 \end{pmatrix} \tag{2.73}$$

The eigenvalues λ_k of the Laplacian matrix $L \in \mathbb{R}^{n\times n}$ are explicitly given by

$$\lambda_k = -\frac{4}{h^2}\sin^2\left(\frac{\pi k h}{2}\right), \quad k = 1, \ldots, n \tag{2.74}$$

Obviously, a large n implies that $nh = \frac{n}{n+1} \approx 1$ and the maximum eigenvalue in magnitude can thus be specified as

$$\lambda_{\max} := \lambda_n = -\frac{4}{h^2}\sin^2\left(\frac{\pi n h}{2}\right) \approx -\frac{4}{h^2} \tag{2.75}$$

To ensure stability the EE method must satisfy

$$|1 + \tau\lambda_{\max}| = \left|1 - \tau\frac{4}{h^2}\right| \leq 1 \quad \Longrightarrow \quad \tau \leq \frac{h^2}{2} =: \tau_{\max} \tag{2.76}$$

meaning the stability requirement depends quadratically on the spatial grid size h. The quadratic relation is also recognisable from the spectral condition number of the discrete Laplace operator given in terms $\mathcal{O}(h^{-2})$. With a finer discretisation, the stiffness rate therefore strives, i.e. the ODE system becomes increasingly stiff. As a consequence, the resulting maximum EE time step size $\tau_{\max}$ leads to an unacceptable amount of effort in practical computations.

The characterisation of moderate and high stiffness is widely used in the literature. This can be explained in terms of the spatial grid size as follows: as shown above, the maximum time step size (2.76) *depends quadratically on h. However, if the underlying spatial discretisation is chosen to be nonuniform, the smallest grid size $h_{\min}$ naturally determines the time step size in the sense of $\tau_{\max} \approx \frac{h_{\min}^2}{2}$. This means, in fact, that a very fine discretisation in a small area of the computational domain leads to an extremely small time step size $\tau_{\max}$, while a rough discretisation in the remaining area would be sufficient. Such a case is often referred to as high stiffness, whereas a uniform spatial discretisation (or a nonuniform with approximately equal grid sizes) can be specified as moderate stiffness.*

A-Stability and L-Stability Solving stiff problems requires the use of numerical methods that have a very large region of absolute stability, as is the case with implicit methods. Notably, not all implicit methods are unconditionally stable, e.g. Radau I or sometimes called the Hammer & Hollingsworth method of order $p = 3$, is a counterexample. Because of this, methods for which the stability region is unbounded, such as IE and CN, appear to be

optimal for solving stiff problems. Unconditionally stable one-step numerical methods, for which any time step size τ is allowed to ensure stability, belong to the class of the so-called *A-stable* methods, in which the region of absolute stability $\mathcal{S}$ contains the entire left half-plane

$$\mathcal{S} \supseteq \mathbb{C}^- \left\{ z \in \mathbb{C} : \operatorname{Re}(z) \leq 0 \right\} \tag{2.77}$$

The latter requirement is often weakened by introducing the $A(\alpha)$ stability with $\pi - \alpha \leq \arg(z) \leq \pi + \alpha$ so that the sectors are contained in the stability region $\mathcal{S}$. This modification can be used for many stiff problems where the eigenvalues lie far out into the left half-plane but near the real axis. In the case of symmetric matrices in which the eigenvalues lie exactly on the real axis, there is no reason to require that the entire left half-plane lies in the region of absolute stability.

Although the CN method is A-stable, which is a desirable property, the scheme possesses an undesirable feature. In fact, the stability function of the CN scheme is characterised by

$$R(z) = \frac{|2+z|}{|2-z|} = \frac{\left|\frac{2}{z}+1\right|}{\left|\frac{2}{z}-1\right|} \to 1 \quad \text{for} \quad z \to \infty \tag{2.78}$$

In certain situations this feature has a significant influence on the quality of the approximate solutions. Let us give a short explanation using an example.

Example 2.3. *Again we consider the system* (2.36) *with the eigenvalues* λ_k *and the corresponding eigenvectors* $\boldsymbol{\phi}_k$ *for* $k = 1, \ldots, n$*. Rewriting the initial condition as a linear combination of the* L_2*-normalised eigenvectors of* L *gives*

$$\boldsymbol{u}^0 = \sum_{k=1}^{n} \alpha_k \boldsymbol{\phi}_k, \quad \alpha_k \in \mathbb{R} \tag{2.79}$$

and applying the CN method the solution can be expressed as

$$\boldsymbol{u}^m = \sum_{k=1}^{n} \alpha_k \left[\frac{1 + \frac{\tau\lambda_k}{2}}{1 - \frac{\tau\lambda_k}{2}} \right]^m \boldsymbol{\phi}_k \tag{2.80}$$

If $R(\tau\lambda_k)$ *is close to -1, the associated component* $\alpha_k\boldsymbol{\phi}_k$ *of the initial condition is roughly retained, but with a changing sign. This is particularly noticeable in the case of initial conditions with discontinuities, since such discontinuities (high frequencies) are preserved by the oscillation and are distorted instead of damped. In order to avoid such problems in connection with discontinuous initial conditions, the time step size of the CN method must therefore be chosen to be sufficiently small.*

In contrast, the stability function for the IE method is specified by

$$R(z) = \frac{1}{|1-z|} \to 0 \quad \text{for} \quad z \to \infty \tag{2.81}$$

so that high frequencies are damped independently of $R(\tau\lambda_k)$, and without producing oscillations. Furthermore, (2.81) shows that the higher the frequency, the stronger the

damping. One-step numerical methods with a stronger stability (damping) property are referred to as *L-stable*, if they are *A*-stable and if their stability function fulfils

$$\lim_{z\to\infty} \left|R(z)\right| = 0 \tag{2.82}$$

Some authors denote *L*-stable methods as strongly *A*-stable. Let us stress that *L*-stability (strong *A*-stability) is a favourable property for solving parabolic-type PDEs.

While conventional implicit methods appear optimal for large stiff problems, in Chapter 3 we will point out a special class of explicit methods (explicit RK-Chebyshev methods) for solving moderately stiff problems.

2.1.4 Explicit Runge-Kutta Methods

The EE method suffers from stability requirements on the time step size, especially in the case of stiff problems. Hence, it might be interesting to investigate higher order explicit schemes and their region of absolute stability. The following short introduction forms the basis for the numerical methods introduced as well as for the fast explicit methods which we consider in more detail in Chapter 3.

The EE, IE and CN schemes belong to the famous class of RK methods. The RK methods are a family of explicit and implicit methods and were developed by Carl Runge and Wilhelm Kutta around 1900. The basic idea is that the new approximation $\boldsymbol{u}^{n+1}$ at time t_{n+1} is computed by $\boldsymbol{u}^n$ at time t_n and intermediate approximations $\boldsymbol{u}_i^n \approx \boldsymbol{u}(t_n + c_i\tau)$ with $i = 1, \ldots, s$, where s is the number of stages. The use of intermediate stages leads to methods with higher order accuracy, but also to higher computational costs due to the increase of function evaluations. In order to briefly describe the basic idea of the RK methods, in particular for the explicit case, we start with an introductory example and consider the semi-discretised system

$$\dot{\boldsymbol{u}}(t) = \boldsymbol{f}\left(t, \boldsymbol{u}(t)\right), \quad \boldsymbol{u}(t_0) = \boldsymbol{u}^0 \tag{2.83}$$

The application of the fundamental theorem of calculus over $[t_0, t_1]$ leads to

$$\boldsymbol{u}(t_1) = \boldsymbol{u}^0 + \int_{t_0}^{t_1} \boldsymbol{f}\left(t, \boldsymbol{u}(t)\right)\, \mathrm{d}t \tag{2.84}$$

The integral can then be approximated, e.g. using the midpoint rule such that

$$\boldsymbol{u}^1 := \boldsymbol{u}(t_1) \approx \boldsymbol{u}^0 + \tau \boldsymbol{f}\left(\frac{t_0+t_1}{2}, \boldsymbol{u}\left(\frac{t_0+t_1}{2}\right)\right) \tag{2.85}$$

with $\tau = t_1 - t_0$ and for which the vector $\boldsymbol{u}(\frac{t_0+t_1}{2})$ is unknown. With the Taylor expansion

$$\boldsymbol{u}\left(\frac{t_0+t_1}{2}\right) = \boldsymbol{u}(t_0) + \frac{\tau}{2}\dot{\boldsymbol{u}}(t) + \mathcal{O}\left(\tau^2\right) \tag{2.86}$$

the unknown $\boldsymbol{u}(\frac{t_0+t_1}{2})$ can approximated by a small EE step

$$\boldsymbol{u}\left(\frac{t_0+t_1}{2}\right) \approx \boldsymbol{u}(t_0) + \frac{\tau}{2}\boldsymbol{f}\left(t_0, \boldsymbol{u}\left(t_0\right)\right) \tag{2.87}$$

Finally, the approximation at time t_1 is given by

$$\boldsymbol{u}(t_1) \approx \boldsymbol{u}^0 + \tau \boldsymbol{f}\left(t_0 + \frac{\tau}{2}, \boldsymbol{u}^0 + \frac{\tau}{2}\boldsymbol{f}\left(t_0, \boldsymbol{u}^0\right)\right) \tag{2.88}$$

which can also be rewritten as

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}\left(t_0, \boldsymbol{u}^0\right) \\ \boldsymbol{k}_2 &= \boldsymbol{f}\left(t_0 + \frac{\tau}{2}, \boldsymbol{u}^0 + \frac{\tau}{2}\boldsymbol{k}_1\right) \\ \boldsymbol{u}^1 &= \boldsymbol{u}^0 + \tau \boldsymbol{k}_2 \end{aligned} \tag{2.89}$$

with the associated stages $\boldsymbol{k}_1$ and $\boldsymbol{k}_2$. Based on the fact that the stage $\boldsymbol{k}_2$ is computed with only a half Euler step, the approximate solution $\boldsymbol{u}^1$ has a smaller error compared to the EE method. In other words, the consistency order of the midpoint method is of order two. An alternative way of determining the order of accuracy is to analyse the condition (2.31) using the Dahlquist test equation. The corresponding stability function can be obtained by

$$\begin{aligned} \boldsymbol{u}^{n+1} &= \boldsymbol{u}^n + z\left(\boldsymbol{u}^n + \frac{z}{2}\boldsymbol{u}^n\right) = \boldsymbol{u}^n + z\boldsymbol{u}^n + \frac{z^2}{2}\boldsymbol{u}^n \\ &= \left(1 + z + \frac{z^2}{2}\right)\boldsymbol{u}^n = R(z)\boldsymbol{u}^n = e^z\boldsymbol{u}^n + \mathcal{O}\left(z^3\right) \end{aligned} \tag{2.90}$$

with $z = \tau\lambda$. By using more accurate quadrature formulas, the number of stages can be increased and thus higher order schemes are derived. In particular, an *s-stage explicit RK method* for (2.83) has the form:

$$\begin{aligned} \boldsymbol{k}_1 &= \boldsymbol{f}\left(t_0, \boldsymbol{u}^0\right) \\ \boldsymbol{k}_2 &= \boldsymbol{f}\left(t_0 + c_2\tau, \boldsymbol{u}^0 + \tau a_{21}\boldsymbol{k}_1\right) \\ \boldsymbol{k}_3 &= \boldsymbol{f}\left(t_0 + c_3\tau, \boldsymbol{u}^0 + \tau\left(a_{31}\boldsymbol{k}_1 + a_{32}\boldsymbol{k}_2\right)\right) \\ &\vdots \\ \boldsymbol{k}_s &= \boldsymbol{f}\left(t_0 + c_s\tau, \boldsymbol{u}^0 + \tau\sum_{i=1}^{s-1}\left(a_{si}\boldsymbol{k}_i\right)\right) \\ \boldsymbol{u}^1 &= \boldsymbol{u}^0 + \tau\sum_{i=1}^{s} b_i k_i \end{aligned} \tag{2.91}$$

with s the number of stages and a_{21}, a_{31}, $a_{32}, \ldots, a_{s1}, a_{s2}, \ldots, a_{s,s-1}, b_1, \ldots, b_s, c_2, \ldots, c_s$ be real coefficients. Usually, for consistency the coefficients require the conditions

$$\begin{aligned} &\sum_{j=1}^{i-1} a_{ij} = c_i, \quad i = 2, \ldots, s \\ &\sum_{j=1}^{s} b_j = 1 \end{aligned} \tag{2.92}$$

Note that these coefficients are often linked to the well-known Butcher tableau. An equivalent definition of the explicit RK method (2.91) is specified in a more general form

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_j &= \boldsymbol{u}^n + \tau \sum_{l=0}^{j-1} a_{jl} \boldsymbol{f}(t_n + c_l \tau, \boldsymbol{y}_l), \quad 1 \leq j \leq s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{2.93}$$

with the time step size $\tau = t_{n+1} - t_n$ and the intermediate solutions $\boldsymbol{y}_j$ at stages j.

Classic examples of explicit RK methods are for instance the first order EE method, the second order midpoint method, the third order Heun method and the original fourth order RK method. As exemplarily shown in (2.90), the stability function of an explicit RK method is always a polynomial:

Theorem 2.1 ([120]). *If the explicit RK method is of order p, then*

$$R(z) = 1 + z + \frac{z^2}{2!} + \cdots + \frac{z^p}{p!} + \mathcal{O}\left(z^{p+1}\right) \tag{2.94}$$

Obviously, explicit RK methods can never be A-stable (2.77). Analysing the real interval of absolute stability using the stability function (2.94) for $p = 1, 2, 3, 4$ gives

$$I = \begin{cases} [-2, 0], & \text{for } \ p = 1 \\ [-2, 0], & \text{for } \ p = 2 \\ [-2.51, 0], & \text{for } \ p = 3 \\ [-2.78, 0], & \text{for } \ p = 4 \end{cases} \tag{2.95}$$

The latter indicates that one-step explicit RK methods are generally only practicable for nonstiff initial value problems.

In contrast, for an s-stage implicit RK method, the stability function $R(z)$ becomes a rational function with numerator and denominator polynomials of degree $\leq s$, represented by

$$R(z) = \frac{P(z)}{Q(z)} \tag{2.96}$$

For most implicit methods the degree of the denominator polynomial is greater than or equal to the degree of the numerator polynomial, so at least A-stability is guaranteed.

2.1.5 Alternative Numerical Methods

There are dozens of numerical methods for solving ODE systems, with each method has advantages and disadvantages and is often used to solve a specific class of problem. Let us give a brief overview of other common numerical methods that are generally unsuitable for our purposes.

Extrapolation Methods Another technique for obtaining higher order accuracy is based on the idea of extrapolation. This approach is often closely related to the *Richardson*

extrapolation and includes the Romberg integration to solve ODEs. Since this class of methods only constructs a higher accuracy rather than a computational speed-up, the extrapolation of explicit methods is usually employed to solve nonstiff ODEs. We will describe this approach in more detail later, especially when considering higher order fast explicit diffusion methods, see Section 3.4.5.

Splitting Methods A widespread class of numerical integration schemes are the splitting methods, whereby this technique can basically be divided into two classes. First, the splitting of a mixed PDE, for example of the parabolic-convection type, in which diffusive and convective forces are considered as two separate sub-problems, and second the splitting of a pure multi-dimensional PDE into a sum of one-dimensional problems. The latter splitting technique, known as *dimensional splitting*, is of more relevant in our setting as the underlying problems in this work are of the pure type.

In this context, efficient numerical schemes are the *alternating direction implicit (ADI)* method first proposed by Peaceman and Rachford, or additive and multiplicative operator splitting schemes, where the latter schemes are often applied in image processing, e.g. for nonlinear diffusion [21, 296, 297] or linear osmosis filtering [53, 206]. Such methods remain implicit by their construction so that again large sparse systems of linear equations have to be solved. Fortunately, the resulting systems are tridiagonal and allow the use of highly efficient Gaussian elimination algorithms. Nevertheless, operator splitting methods need to be handled more carefully when considering more complex (time-dependent) boundary conditions or nonrectangular/unstructured grids. Another delicate issue is that undesirable splitting errors normally occur, which can have a strong influence on the numerical solution. To overcome this problem, the time step size must often be reduced significantly. As a consequence, a greater number of iterations are required to reach the stopping time, which is directly linked to the efficiency of these methods. We are going into more detail about the operator splitting methods in Chapter 7.

Multi-Step Methods There is also another class, namely *multi-step methods*, that use several previous solutions to compute the numerical solution at the current time step. Based on the nature of multi-step methods, they can be computationally more efficient when higher order schemes are applied, as fewer calculations are needed to produce the same order compared to one-step methods. A common and effective representative of multi-step methods is the implicit backward differentiation formula method. However, higher order schemes are generally only better suited for smaller time step sizes τ, so their use is directly linked to high effort when solving large sparse linear systems.

Let us emphasise that there exists even an explicit unconditionally stable method, the *DuFort-Frankel scheme*. On closer examination, it can be analysed that this method is not necessarily consistent (only conditionally consistent) with the underlying differential equation. Therefore, the DuFort-Frankel scheme only converges to the solution if $\frac{\tau}{h^2} \to 0$, more precisely, an accurate solution is only obtained if $\tau \ll h$.

Since we deal with linear parabolic model problems coupled with large stopping times, in general only low order (first or second order) accurate methods are of interest. Consequently, we do not consider multi-step methods in this thesis.

2.1.6 Summary

In general, schemes for the temporal integration of semi-discretised ODE systems can be broadly categorised into explicit and implicit methods. Explicit schemes are very easy to implement and well-suited for parallelisation on GPUs. In order to ensure the required numerical stability, however, they have severe time step size restrictions. In simple terms, explicit methods are always conditionally stable, where a too large time step size leading to a significant numerical error. Hence, stiff problems require a huge number of explicit steps to reach the desired stopping time of the underlying physical process, and this usually makes explicit methods unsuitable for practical purposes.

In contrast, most implicit schemes possess stability regions that contain the entire left half-plane (unconditionally stable) and provide accurate enough approximate solutions for a reasonable time step size. Because of this, implicit methods are often considered to be the best choice for solving stiff problems, which are typically arise from semi-discretised parabolic or hyperbolic-parabolic equations. Nonetheless, the beneficial stability properties come at the expense of solving large sparse (non)linear systems at each time level. This is known to be computationally intensive, requires large memory space, and the underlying solution process is also more difficult to parallelise. Moreover, especially in the case of very large systems for which iterative methods are generally applied, several parameters must be taken into account that make implicit schemes more parameter-sensitive.

2.2 Solution of Sparse Systems of Linear Equations

As described in the last section, implicit methods such as the IE method (2.8) and the CN method (2.10) are unconditionally stable, but require the solution of a large sparse system of linear equations given by

$$A\boldsymbol{x} = \boldsymbol{b} \tag{2.97}$$

with a large sparse matrix $A \in \mathbb{R}^{n\times n}$, a right-hand side $\boldsymbol{b} \in \mathbb{R}^n$ and a solution vector $\boldsymbol{x} \in \mathbb{R}^n$ at each time level. For this reason, the application of time integration methods leads to a new nonnegligible challenge, namely solving a system of linear equations with multiple right-hand sides. Dealing with large sparse systems implies two main issues: the accuracy of the solution and the computational efficiency of the numerical solver used. Consequently, the main component for efficient algorithms is a fast procedure to solve linear systems multiple times. There are several numerical solvers for solving linear systems, which have different advantages in terms of computational effort and accuracy of the computed solution, but also in their use and implementation. In general, sparse direct and sparse iterative methods are the most common solvers to compute the solution of linear systems. However, the correct choice of the solver still strongly depends on the model problem considered.

The use of sparse direct solvers, developed on the basis of Gaussian elimination, computes highly accurate solutions and is generally predestined for solving a linear system with multiple right-hand sides. In this case, the underlying matrix will be factorised once only, and subsequently the system is solved for each right-hand side by forward and backward substitution. Nevertheless, this type of solver can cause high memory and computational costs, and appears to be rather impractical for very large systems due to the use of a complete factorisation. In contrast, sparse iterative methods are naturally not tweaked for extremely

high accuracy, but are very fast in computing approximate solutions. These methods are based on sparse matrix-vector multiplications and require less memory space, and are thus inherently attractive candidates for solving very large systems of linear equations. Although iterative solvers are characterised by low algorithmic complexity, their run time depends on the data, size, sparsity and accuracy required, making these methods a tool that is not straightforward to use.

In the following we describe sparse direct and sparse iterative solvers in more detail. Further information on the efficient solution of general large linear systems can be found in the works [104, 181].

Sparse Direct Methods Two classical approaches to compute the solution of (2.97) are based on the explicit computation of the inverse or the determinant of A. Assuming that the inverse of A exists, the solution is simply given by $\boldsymbol{x} = A^{-1}\boldsymbol{b}$. Otherwise, according to Cramer's rule, the solution reads as

$$\boldsymbol{x}_i = \frac{\det(A_i)}{\det(A)}, \quad i = 1, \ldots, n \tag{2.98}$$

where A_i are the matrices that result from A by substituting the i-th column with the right-hand side b. While both approaches may seem practical, their use is normally impossible.

In general, direct methods refer to *Gaussian elimination* and the *lower-upper (LU)* factorisation, and its modifications and extensions. In simple terms, the classic Gaussian elimination transforms the original system (2.97) into an equivalent upper triangular system $U\boldsymbol{x} = \boldsymbol{y}$ which can easily be solved for $\boldsymbol{x}$. This approach is not applied directly for each right-hand side, but rather the Gaussian elimination is employed as LU factorisation, in which a lower triangular matrix is also constructed within the first elimination algorithm. In fact, the given matrix A is factorised by lower and upper triangular $n \times n$ matrices L and U, respectively, into $A = LU$. Subsequently, the solution $\boldsymbol{x}$ can be computed highly efficiently from the right hand side $\boldsymbol{b}$ by first solving $L\boldsymbol{y} = \boldsymbol{b}$ for $\boldsymbol{y}$, and then $U\boldsymbol{x} = \boldsymbol{y}$ for $\boldsymbol{x}$. The latter procedure is known as *forward* and *backward* substitution. In the special case of a symmetric and positive definite matrix A, which often arises in connection with the Laplace operator Δ, the so-called *Cholesky* factorisation $A = CC^\top$ with a lower triangular matrix C is mainly used. As a result, the LU factorisation can be avoided that causes higher CPU time and memory requirements. In addition, it is known that the Cholesky factorisation is numerically very robust due to the advantageous properties of symmetric structures.

Although factorisation is a straightforward process from a theoretical point of view, the factors usually do not preserve the sparsity even if the matrix A is sparse. Thus, the main objective is to determine the factorisation in an efficient manner so that the factors are close to being optimally sparse. This task is not trivial and includes the importance of bandwith minimisation and reordering, control fill-in strategies and the connections to graph theory. In this regard, a good knowledge of techniques from numerical linear algebra, graph algorithms and permutations is required. We refer the reader to [47, 73, 75, 104, 181] and the reference therein for more details on sparse direct methods.

Overall, direct methods are a powerful tool and theoretically provide the exact solution if rounding errors that typically occur in computational practice are neglected. In the case of very large linear systems, however, the complete factorisation is associated with a high level of computation and storage complexity and leads to the impracticability of the direct methods

in practice. Because of this, cheaper iterative methods are becoming a viable alternative and are commonly used.

Sparse Iterative Methods The general idea of iterative methods is to successively compute a sequence of approximate solutions $\{\boldsymbol{x}_k\}$, starting from a given initial $\boldsymbol{x}_0$, which converges to $\boldsymbol{x} = A^{-1}\boldsymbol{b}$, i.e.

$$\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_k \overset{k\to\infty}{\longrightarrow} \boldsymbol{x} \in \mathbb{R}^n : A\boldsymbol{x} = \boldsymbol{b} \tag{2.99}$$

Typically, effective iterative methods are based on matrix-vector multiplications, which are a relatively cheap process from a computational point of view, and that is exactly what makes this framework highly attractive when the underlying matrix A is large and sparse. In connection with the procedure (2.99), important issues arise such as the rate of convergence, the computational effort per iteration, the memory requirement, the error control and the construction principles of the iterative process.

In general, there are three types of iterative methods: splitting methods, Krylov subspace methods and multigrid methods. Splitting methods such as the Jacobi method, the Gauss-Seidel method or the relaxation methods are the simplest and most classic iterative methods. These types of methods are usually only of theoretical significance in modern scientific computing because their rate of convergence is very slow. Therefore, Krylov subspace methods and multigrid methods are frequently applied due to their powerful performance. For a more detailed insight into iterative methods we refer to [22, 104, 145, 178, 181, 238].

Based on the fact that a large class of standard problems related to the Laplace operator often leads to sparse, *symmetric and positive definite* matrices, we further focus on this class since exploiting its special structure enables a numerically robust and efficient solution. In this situation, the *conjugate gradient (CG)* method [128] is best suited, as it provides fast convergence with monotonically decreasing error and is generally the most efficient and numerically robust scheme. In what follows, we present the basic theory and discuss the practical implementation that sheds light on the behaviour of the CG method, which is one of the most popular methods in numerical linear algebra.

2.2.1 Conjugate Gradient Method

A particular class of iterative solvers designed for use with large sparse linear systems is the class of *Krylov subspace solvers*. The main idea behind the Krylov approach is to search for an approximate solution of (2.97) in a suitable low-dimensional subspace $\mathbb{R}^l$ of $\mathbb{R}^n$ that is constructed iteratively with l being the number of iterates. The aim in the construction is to have a good representation of the solution after a few iterates. It should be noted that this construction is often not directly visible in the formulation of a Krylov subspace method.

The well-known CG method of Hestenes and Stiefel [128] is probably the most famous Krylov subspace method and a widely used iterative solver for problems with large, sparse symmetric and positive definite matrices. For the CG method it can be shown that the approximate solutions $\boldsymbol{x}_l$ of (2.97) are optimal in a sense as they minimise the so-called energy norm of the error vector [181]. In other words, the CG method gives in the l-th iteration the best solution available in the generated subspace. Since the dimension of the Krylov subspace is increased in each iteration, theoretical convergence is achieved at latest after the n-th step of the method if the sought solution is in $\mathbb{R}^n$.

The derivation of the CG method is basically connected with a projection method or the minimisation of an optimisation problem. Let us describe both viewpoints in some detail.

Projection Method The basic task of a projection method is to seek for an approximate solution $\boldsymbol{x}_m$ to (2.97) from an m-dimensional subspace $K_m \subset \mathbb{R}^n$, so that the so-called *residual* vector

$$\boldsymbol{r}_m = \boldsymbol{b} - A\boldsymbol{x}_m \tag{2.100}$$

is orthogonal to another m-dimensional subspace $L_m \subset \mathbb{R}^n$, i.e.

$$\boldsymbol{w}^\top \boldsymbol{r}_m = \boldsymbol{w}^\top(\boldsymbol{b} - A\boldsymbol{x}_m) = 0 \quad \longleftrightarrow \quad \boldsymbol{w}^\top \perp \boldsymbol{r}_m, \quad \forall \boldsymbol{w} \in L_m \tag{2.101}$$

and thus makes use of an orthogonality condition. Assuming that an initial vector $\boldsymbol{x}_0$ for $\boldsymbol{x}$ is given, the solution then lies in the space $\boldsymbol{x}_0 + K_m$, more precisely, it holds $\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{d}$ with some vector $\boldsymbol{d}$ in K_m. In simple terms, the problem is to find $\boldsymbol{x}_m \in \boldsymbol{x}_0 + K_m$ such that

$$\boldsymbol{w}^\top \boldsymbol{r}_m = \boldsymbol{w}^\top(\boldsymbol{b} - A(\boldsymbol{x}_0 + \boldsymbol{d})) = \boldsymbol{w}^\top(\boldsymbol{r}_0 - A\boldsymbol{d}) = 0, \quad \forall \boldsymbol{w} \in L_m \tag{2.102}$$

with an initial residual $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$. Suppose now $V = [v_1, \ldots, v_m]$ and $W = [w_1, \ldots, w_m]$ are two matrices of size $n \times m$ whose columns form a basis for K_m and L_m, respectively. Then, the approximate solution can be written as $\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{d} = \boldsymbol{x}_0 + V\boldsymbol{y}$ with some $\boldsymbol{y} \in \mathbb{R}^m$, and the orthogonality implies that

$$\boldsymbol{w}^\top(\boldsymbol{r}_0 - AV\boldsymbol{y}) = 0, \quad \forall \boldsymbol{w} \in L_m \tag{2.103}$$

which is equivalent to

$$W^\top(\boldsymbol{r}_0 - AV\boldsymbol{y}) = 0 \quad \longleftrightarrow \quad W^\top AV\boldsymbol{y} = W^\top \boldsymbol{r}_0 \tag{2.104}$$

Finally, assuming the inverse of $W^\top AV$ exists, the approximate solution reads

$$\boldsymbol{x}_m = \boldsymbol{x}_0 + \boldsymbol{d} = \boldsymbol{x}_0 + V\boldsymbol{y} = \boldsymbol{x}_0 + V\left(W^\top AV\right)^{-1} W^\top \boldsymbol{r}_0 \tag{2.105}$$

If $K_m = L_m$, the projection is *orthogonal*, otherwise for $K_m \neq L_m$ the projection method is called *skewed*. In practice, (2.105) is iterated such that a new pair of subspaces K_m and L_m is used in each iteration, with the initial $\boldsymbol{x}_0$ equal to the approximate solution from the previous iterate.

The iterative methods for sparse linear systems use the projection onto so-called *Krylov subspaces*, where the m-th Krylov subspace $\mathcal{K}_m(A, \boldsymbol{v}) \subset \mathbb{R}^n$ is defined by

$$\mathcal{K}_m := \mathcal{K}_m(A, \boldsymbol{v}) = \operatorname{span}\left(\boldsymbol{v}, A\boldsymbol{v}, A^2\boldsymbol{v}, \ldots, A^{m-1}\boldsymbol{v}\right) \tag{2.106}$$

In fact, the Krylov subspace $\mathcal{K}_m$ is generated by the matrix A and the vector $\boldsymbol{v}$. On closer inspection, the procedure (2.106) is nothing else than an algorithm to increase successively the dimension of $\mathcal{K}_m$. By combining the projection method (2.105) with the Krylov subspaces, the CG method can finally be derived. In this case, an orthogonal projection with $L_m = K_m = \mathcal{K}_m(A, \boldsymbol{r}_0)$ is used. In particular, this means $\mathcal{K}_m$ is generated from an initial residual vector $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$ by successive cheap multiplications with the sparse matrix A. The

nature of an iterative Krylov subspace method is that the computed approximate solution $\boldsymbol{x}_m$ belongs to $\boldsymbol{x}_0 + \mathcal{K}_m(A, \boldsymbol{r}_0)$, which means that it is determined by the m-th Krylov subspace. Thereby, the index m is also the m-th iteration of the iterative scheme.

In general, Krylov subspace methods are often derived by reformulating (2.97) as a minimisation problem. In the case of the CG scheme, the construction combines the method of steepest descent with the method of conjugate directions.

Optimisation Problem The basis for the derivation of the CG method is based on the fact that for such a matrix the solution of $A\boldsymbol{x} = \boldsymbol{b}$ is exactly the minimum of the quadratic form

$$F(\boldsymbol{x}) = \frac{1}{2}\langle \boldsymbol{x}, A\boldsymbol{x}\rangle_2 - \langle \boldsymbol{b}, \mathbf{x}\rangle_2 \tag{2.107}$$

since for a minimiser it holds that

$$\nabla F(\boldsymbol{x}) = \tfrac{1}{2}(A + A^\top)\boldsymbol{x} - \boldsymbol{b} = A\boldsymbol{x} - \boldsymbol{b} \stackrel{!}{=} \mathbf{0} \quad \Longleftrightarrow \quad A\boldsymbol{x} = \boldsymbol{b} \tag{2.108}$$

Here, $\langle \cdot, \cdot \rangle_2$ means the Euclidean scalar product. In other words, the minimum $\boldsymbol{x}^\star$ of F in (2.107) solves the linear system $A\boldsymbol{x}^\star = \boldsymbol{b}$. The equivalence (2.108) is always true if A is symmetric and positive definite, since F is convex. Moreover, the solution vector that satisfies (2.108) is unique. Let us mention that the quadratic form in (2.107) is often referred to as the energy functional.

Thus, if $\boldsymbol{x}_m$ is an approximate minimiser of F, then $\boldsymbol{x}_m$ can be understood as an approximate solution to $A\boldsymbol{x} = \boldsymbol{b}$. More precisely, let $\|\cdot\|_A$ be the A-norm given by $\|\boldsymbol{y}\|_A := \sqrt{\boldsymbol{y}^\top A\boldsymbol{y}}$. Because of

$$F(\boldsymbol{x}_m) = \frac{1}{2}\boldsymbol{x}^\top A\boldsymbol{x} - \boldsymbol{b}^\top \boldsymbol{x} = \frac{1}{2}(\boldsymbol{x}_m - \boldsymbol{x}^\star)^\top A(\boldsymbol{x}_m - \boldsymbol{x}^\star) - \frac{1}{2}\boldsymbol{b}^\top A^{-1}\boldsymbol{b} \tag{2.109}$$

and $F(\boldsymbol{x}^\star) = -\frac{1}{2}\boldsymbol{b}^\top A^{-1}\boldsymbol{b}$, it follows that

$$F(\boldsymbol{x}_m) = \frac{1}{2}\|\boldsymbol{x}_m - \boldsymbol{x}^\star\|_A^2 + F(\boldsymbol{x}^\star) \tag{2.110}$$

Consequently, an algorithm that generates a sequence of even better approximate minimisers for F, at the same time produces even better approximate solutions to $A\boldsymbol{x} = \boldsymbol{b}$ with respect to the A-norm.

These observations now imply the aim of minimising F successively, starting from some point $\boldsymbol{x} \in \mathbb{R}^n$ and along specific directions $\boldsymbol{p} \in \mathbb{R}^n$. In doing so, the function $f_{\boldsymbol{x},\boldsymbol{p}}(d) = F(\boldsymbol{x} + d\boldsymbol{p})$ is defined for $\boldsymbol{x}, \boldsymbol{p}$ and the following proposition holds:

Proposition 2.1 ([178]). *Let the matrix $A \in \mathbb{R}^{n\times n}$ be symmetric and positive definite and the vectors $\boldsymbol{x}, \boldsymbol{p} \in \mathbb{R}^n$, with $\boldsymbol{p} \neq \mathbf{0}$, given. The global minimum of $f_{\boldsymbol{x},\boldsymbol{p}}$, i.e. the minimum of F, starting in $\boldsymbol{x}$ and searching along $\boldsymbol{x} + d\boldsymbol{p}$, is given by*

$$d_{\text{opt}} = \frac{\langle \boldsymbol{r}, \boldsymbol{p}\rangle_2}{\langle A\boldsymbol{p}, \boldsymbol{p}\rangle_2} \tag{2.111}$$

where $\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x}$.

As a result, for a sequence $\{\boldsymbol{p}_m\}$ with $\boldsymbol{p}_m \neq \boldsymbol{0}$ and given $\boldsymbol{x}_0$, the algorithm with three calculations in each iteration reads:

$$\boldsymbol{r}_m = \boldsymbol{b} - A\boldsymbol{x}_m, \quad d_m = \frac{\langle \boldsymbol{r}_m, \boldsymbol{p}_m \rangle_2}{\langle A\boldsymbol{p}_m, \boldsymbol{p}_m \rangle_2}, \quad \boldsymbol{x}_{m+1} = \boldsymbol{x}_m + d_m \boldsymbol{p}_m, \quad m = 0, 1, \ldots \tag{2.112}$$

Obviously, a calculation rule is still required for the *search directions* $\{\boldsymbol{p}_m\}$. Furthermore, without loss of generality it is assumed $\|\boldsymbol{p}_m\|_2 = 1$. The local optimal search direction is then defined via the *downhill direction*. Using

$$\nabla F(\boldsymbol{x}) = A\boldsymbol{x} - \boldsymbol{b} = -\boldsymbol{r} \tag{2.113}$$

the well-known direction of steepest descent $-\nabla F(\boldsymbol{x})$ at $\boldsymbol{x}$ is obtained as

$$\widetilde{\boldsymbol{p}}_m = -\nabla F(\boldsymbol{x}_m) = \boldsymbol{r}_m \tag{2.114}$$

with additional normalisation

$$\boldsymbol{p}_m = \frac{\widetilde{\boldsymbol{p}}_m}{\|\widetilde{\boldsymbol{p}}_m\|_2} = \frac{\boldsymbol{r}_m}{\|\boldsymbol{r}_m\|_2} \tag{2.115}$$

Thus, the residual vectors define the search directions. Let us mention that the latter construction is often called *the method of steepest descent*.

The use of the negative gradient direction is cheap to compute and extremely advantageous. Nevertheless, this procedure causes a natural weakness, namely the typical *zig-zagging behaviour* of the method. This means that the directions of the negative gradient can oscillate rapidly during the gradient descent process and often producing zigzag paths, so a large amount of iterations are needed to reach a near minimiser. The reason for this is that previously used search directions $\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{m-1}$ are neglected and are not considered in the current search direction $\boldsymbol{p}_m$. For this reason, the CG method has to be modified in such a way that the successive minimisation is performed along a set of linearly independent search directions $\{\boldsymbol{p}_m\}$, so that

$$\{\boldsymbol{x}_m\} = \operatorname{argmin} \left\{ F(\boldsymbol{x}) \,\middle|\, \boldsymbol{x} \in \boldsymbol{x}_0 + \operatorname{span}\{\boldsymbol{p}_0, \ldots, \boldsymbol{p}_m\} \right\} \tag{2.116}$$

To be more precise, the method of steepest descent gives an orthogonal projection with $K = L = \operatorname{span}\{\boldsymbol{r}_{m-1}\}$ in each iteration. However, an optimality of the iterates with respect to the entire subspace $U = \operatorname{span}\{\boldsymbol{r}_0, \ldots, \boldsymbol{r}_{m-1}\}$ would be desirable, since for linearly independent residual vectors the exact solution $\boldsymbol{x}^\star$ is found after at most n iterations if rounding errors are neglected.

On this basis, the procedure can be generalised by extending the optimality of the approximations $\boldsymbol{x}_m$ onto the entire subspace $U_m = \operatorname{span}\{\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{m-1}\}$ with linearly independent search directions. This optimality can be specified as follows: for F as in (2.107), $\boldsymbol{x} \in \mathbb{R}^n$ is optimal, first with respect to $\boldsymbol{p} \in \mathbb{R}^n \backslash \{\boldsymbol{0}\}$, if

$$F(\boldsymbol{x}) \leq F(\boldsymbol{x} + d\boldsymbol{p}), \quad \forall d \in \mathbb{R} \tag{2.117}$$

and second with respect to the subspace $U \subset \mathbb{R}^n$, if

$$F(\boldsymbol{x}) \leq F(\boldsymbol{x} + \boldsymbol{\xi}), \quad \forall \boldsymbol{\xi} \in U \tag{2.118}$$

The following can be shown:

Proposition 2.2 ([178]). *For F as in (2.107), $\boldsymbol{x} \in \mathbb{R}^n$ is optimal with respect to $U \subset \mathbb{R}^n$, if*

$$\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{x} \perp U \tag{2.119}$$

holds.

For the definition of a constructive procedure, first the question arises about the dimension of U. In simple terms, a collection of search directions $\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{m-1}$ should ensure to span an m-dimensional subspace of $\mathbb{R}^n$, so that

$$\dim U_m = m \quad \text{for } U_m = \operatorname{span}\{\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{m-1}\} \tag{2.120}$$

This can be ensured, if the search directions are pairwise conjugate or A-orthogonal:

$$\langle \boldsymbol{p}_i, \boldsymbol{p}_j \rangle_A := \langle A\boldsymbol{p}_i, \boldsymbol{p}_j \rangle_2 = 0, \quad \forall i, j \in \{0, \ldots, m-1\}, i \neq j \tag{2.121}$$

Using the A-orthogonality the following proposition holds:

Proposition 2.3 ([178]). *For a symmetric and positive definite matrix $A \in \mathbb{R}^{n \times n}$ and search directions $\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{m-1} \in \mathbb{R}^n \backslash \{\boldsymbol{0}\}$ pairwise A-orthogonal, then*

$$\dim(\operatorname{span}\{\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{m-1}\}) = m \tag{2.122}$$

Let $\boldsymbol{p}_0, \ldots, \boldsymbol{p}_{m-1} \in \mathbb{R}^n \backslash \{\boldsymbol{0}\}$ be pairwise conjugate search directions and $\boldsymbol{x}_m$ be optimal with respect to U_m. Then the optimality of $\boldsymbol{x}_{m+1} = \boldsymbol{x}_m + d\boldsymbol{p}_m$ with respect to U_{m+1} for $j = 0, \ldots, m$ using (2.119) and (2.121) is obtained by

$$0 = \langle \boldsymbol{b} - A\boldsymbol{x}_{m+1}, \boldsymbol{p}_j \rangle_2 = \underbrace{\langle \boldsymbol{b} - A\boldsymbol{x}_m, \boldsymbol{p}_j \rangle_2}_{=0 \text{ for } j \neq m} + d \underbrace{\langle A\boldsymbol{p}_m, \boldsymbol{p}_j \rangle_2}_{=0 \text{ for } j \neq m} \tag{2.123}$$

Moreover, the representation for d yields

$$d = \frac{\langle \boldsymbol{r}_m, \boldsymbol{p}_m \rangle_2}{\langle A\boldsymbol{p}_m, \boldsymbol{p}_m \rangle_2} \tag{2.124}$$

The resulting algorithm for given $\boldsymbol{x}_0, \boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$ and $m = 0, 1, \ldots, n-1$, where $\{\boldsymbol{p}_m\}$ is still to be determined, results in:

$$d_m = \frac{\langle \boldsymbol{r}_m, \boldsymbol{p}_m \rangle_2}{\langle A\boldsymbol{p}_m, \boldsymbol{p}_m \rangle_2}, \quad \boldsymbol{x}_{m+1} = \boldsymbol{x}_m + d_m \boldsymbol{p}_m, \quad \boldsymbol{r}_{m+1} = \boldsymbol{r}_m - d_m A\boldsymbol{p}_m \tag{2.125}$$

The current residual within the algorithm (2.125) results from

$$\boldsymbol{r}_{m+1} = \boldsymbol{b} - A\boldsymbol{x}_{m+1} = \boldsymbol{b} - A(\boldsymbol{x}_m + d_m \boldsymbol{p}_m) = \boldsymbol{b} - A\boldsymbol{x}_m - Ad_m \boldsymbol{p}_m = \boldsymbol{r}_m - d_m A\boldsymbol{p}_m \tag{2.126}$$

We emphasise that the latter construction is often called *the method of conjugate directions.*

Finally, as indicated earlier, the CG method [128] combines the method of steepest descent with the method of conjugate directions. To this end, the following ansatz is used with the

residual vectors as search directions:

$$\boldsymbol{p}_0 = \boldsymbol{r}_0, \quad \boldsymbol{p}_m = \boldsymbol{r}_m + \sum_{j=0}^{m-1} \alpha_j \boldsymbol{p}_j \tag{2.127}$$

For $\alpha_j = 0$, the method of steepest descent is recovered. Let now the α_j's, m degrees of freedom for ensuring A-orthogonality, then with (2.127) this implies

$$0 = \langle A\boldsymbol{p}_m, \boldsymbol{p}_i \rangle_2 = \langle A\boldsymbol{r}_m, \boldsymbol{p}_i \rangle_2 + \sum_{j=0}^{m-1} \alpha_j \langle A\boldsymbol{p}_j, \boldsymbol{p}_i \rangle_2 \tag{2.128}$$

for $i = 0, \ldots, m-1$. With $\langle A\boldsymbol{p}_j, \boldsymbol{p}_i \rangle_2 = 0$ for $i, j \in \{0, \ldots, m-1\}$ and $i \neq j$ it follows that

$$\alpha_i = -\frac{\langle A\boldsymbol{r}_m, \boldsymbol{p}_i \rangle_2}{\langle A\boldsymbol{p}_i, \boldsymbol{p}_i \rangle_2} \tag{2.129}$$

Although all building blocks are now available, the representation (2.127) leads to inefficient computations of the current $\boldsymbol{p}_m$ and the storage of all the search directions is required. This problem can be tackled with the A-orthogonality and some mathematical expressions (see [178]) using the simplified formula

$$\boldsymbol{p}_m = \boldsymbol{r}_m - \frac{\langle A\boldsymbol{r}_m, \boldsymbol{p}_{m-1} \rangle_2}{\langle A\boldsymbol{p}_{m-1}, \boldsymbol{p}_{m-1} \rangle_2} \boldsymbol{p}_{m-1} = \boldsymbol{r}_m + \frac{\langle \boldsymbol{r}_m, \boldsymbol{r}_m \rangle_2}{\langle \boldsymbol{r}_{m-1}, \boldsymbol{r}_{m-1} \rangle_2} \boldsymbol{p}_{m-1} \tag{2.130}$$

More precisely, each new conjugate vector $\boldsymbol{p}_m$ can only be computed using the previous vector $\boldsymbol{p}_{m-1}$ and is still automatically conjugate to all the other previously computed vectors.

In total, the storage costs are cheap and independent of the number of iterations performed, moreover, only one sparse matrix-vector multiplication per iteration is needed. The final CG algorithm is shown in Figure 2.2. In theory, the CG method converges to the exact solution after at most n iterations, since the Krylov subspace spans the entire space $\mathbb{R}^n$. In practice, however, this cannot hold due to the finite precision of computer arithmetic and rounding errors that occur.

Of course, it is desirable to terminate the CG algorithm long before k approaches n. Finally, let us consider the efficiency of the CG method which depends on the so-called condition number, which for a regular matrix A is defined as

$$\mathrm{cond}_a := \|A\|_a \|A^{-1}\|_a \tag{2.131}$$

where

$$\|A\|_a := \sup_{\|\boldsymbol{x}\|_a = 1} \|A\boldsymbol{x}\|_a \tag{2.132}$$

is the matrix norm induced by the vector norm $\|\cdot\|_a$. For the CG method one can show the following convergence statement on the basis of the condition number of the system matrix using the A-norm:

Theorem 2.2 ([178]). *Let the matrix $A \in \mathbb{R}^{n \times n}$ be symmetric and positive definite and let $\{\boldsymbol{x}_m\}_{m \in \mathbb{N}_0}$ be the sequence of approximated solutions generated by the CG method. Then the*

corresponding error vector $\boldsymbol{e}_m = \boldsymbol{x}_m - A^{-1}\boldsymbol{b}$ *satisfies the inequality*

$$\|\boldsymbol{e}_m\|_A \leq 2\left(\frac{\sqrt{\text{cond}_2(A)}-1}{\sqrt{\text{cond}_2(A)}+1}\right)^m \|\boldsymbol{e}_0\|_A \tag{2.133}$$

The latter indicates a weakness as the upper error bound depends on the condition number of A. More precisely, if discretised parabolic PDEs are considered, the inequality $\text{cond}(A) \leq \mathcal{O}(h^{-2})$ with the minimum grid size h holds. Consequently, with small h the quotient $(\sqrt{\text{cond}_2(A)}-1)/(\sqrt{\text{cond}_2(A)}+1)$ is close to unity, with end up that no convergence is achieved from a theoretical point of view. This observation gives no hope for an early termination of the algorithm in practice. Fortunately, this problem can be overcome by proper so-called *preconditioning*, which is a way to induce rapid convergence. We are going into this in the next Section. First, let us turn to some computational details about using the CG method in practice.

Computational Aspects Although the theory presented above is generally based on the symmetry and positive definiteness of A, the CG method is also applicable even if the matrix A is just positive semi-definite, but there is no guarantee that $A\boldsymbol{x} = \boldsymbol{b}$ will be solved. The positive definiteness is important for avoiding division by zero within the CG algorithm.

Algorithm 2.1 Conjugate gradient method

Input: Matrix $A \in \mathbb{R}^{n\times n}$; right-hand side $\boldsymbol{b} \in \mathbb{R}^n$; initial vector $\boldsymbol{x}_0 \in \mathbb{R}^n$

Output: Approximate solution for $A\boldsymbol{x} = \boldsymbol{b}$

1.) $\boldsymbol{p}_0 := \boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0, \quad \alpha_0 = \|\boldsymbol{r}_0\|_2^2$

2.) **Iterate** for $m = 0, 1, \ldots, n-1$

a) If $\alpha_m \neq 0$ proceed, else STOP

b) $\boldsymbol{v}_m = A\boldsymbol{p}_m, \quad d_m = \frac{\alpha_m}{\langle \boldsymbol{v}_m, \boldsymbol{p}_m\rangle_2}$

c) $\boldsymbol{x}_{m+1} = \boldsymbol{x}_m + d_m\boldsymbol{p}_m$

d) $\boldsymbol{r}_{m+1} = \boldsymbol{r}_m - d_m\boldsymbol{v}_m$

e) $\alpha_{m+1} = \|\boldsymbol{r}_{m+1}\|_2^2$

f) $\boldsymbol{p}_{m+1} = \boldsymbol{r}_{m+1} + \frac{\alpha_{m+1}}{\alpha_m}\boldsymbol{p}_m$

Figure 2.2: The CG scheme as an orthogonal Krylov subspace method for computing an approximate solution to $A\boldsymbol{x} = \boldsymbol{b}$. The basic algorithm stops, if $\boldsymbol{r}_m = \boldsymbol{0}$. Overall, the CG algorithm is cheap both in terms of computational complexity and memory requirements, since it only requires one sparse matrix-vector multiplication per iteration and storage of four vectors.

However, if A is only positive semi-definite, it may happen that a restart with a different initialisation is necessary due to breakdowns. Apart from that, CG can also be practically used for slightly nonsymmetric matrices, see [255].

Another aspect is the termination of the CG method, as illustrated in Algorithm 2.1. Of course, the CG algorithm will not run until the exact solution has been found, so that the choice $\alpha_m \neq 0$ must be clarified. Since the error vector $\boldsymbol{e}_m = \boldsymbol{x}_m - A^{-1}\boldsymbol{b}$ is not available, it makes sense to have a termination criterion based on $\|\boldsymbol{r}_m\|$. In a practical setting, the norm of the residual vector $\boldsymbol{r}_m = \boldsymbol{b} - A\boldsymbol{x}_m$ usually serves as a measure for the quality of $\boldsymbol{x}_m$. One reason for this is that $\boldsymbol{r}_m = \boldsymbol{0}$ implies $\boldsymbol{x}_m = \boldsymbol{x}$ with $A\boldsymbol{x} = \boldsymbol{b}$. Therefore, most of the iterative methods terminate when the residual is sufficiently small. One termination criterion is

$$\frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_0\|} \leq \varepsilon \tag{2.134}$$

since for an iterative scheme in the k-th iteration the following condition holds:

$$\frac{\|\boldsymbol{e}_k\|}{\|\boldsymbol{e}_0\|} \leq \mathrm{cond}(A)\frac{\|\boldsymbol{r}_k\|}{\|\boldsymbol{r}_0\|} \tag{2.135}$$

Obviously, the norm of $\boldsymbol{r}_k$ is only really meaningful for small condition numbers. Since for a larger value $\mathrm{cond}(A)$, even small values $\|\boldsymbol{r}_k\|$ does not show that $\boldsymbol{x}_k$ is a good approximate of $\boldsymbol{x}$ as $\mathrm{cond}(A)\|\boldsymbol{r}_k\|\|\boldsymbol{r}_0\|^{-1}$ can be considerably large. The latter fact together with (2.133) also indicates another issue of iterative methods. A typical problem with these methods is that their convergence becomes slower for larger time step sizes τ, as we might recall, $A := I - \tau L$ for the IE method, because the condition number of the system matrix A increases. As a consequence, the time step size chosen within the implicit scheme has a direct impact on the convergence and termination of the CG algorithm. This highlights a difference to the direct methods, which can be understood to be independent on the time step size within the forward and backward solution procedure.

The termination criterion (2.134) depends on the initial iterate $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$. On the one hand, this can lead to unnecessary work if the initial iterate is good, otherwise to a worse approximation if the initial iterate is far from the solution. Because of this, the general stopping criterion is normally based on the *relative residual*:

$$\frac{\|\boldsymbol{r}_k\|_2}{\|\boldsymbol{b}\|_2} \leq \varepsilon \tag{2.136}$$

Both conditions (2.134) and (2.136) are equal if $\boldsymbol{x}_0 = \boldsymbol{0}$, which is a common choice. At this point it should be emphasised that the reduction in $\|\boldsymbol{r}\|_2$ does not necessarily decrease monotonically in general, unlike the energy norm of the error vector, i.e. $\|\boldsymbol{e}\|_A$, see [145, 166]. As mentioned earlier, the convergence of the CG scheme is only theoretically monotonic with respect to the A-norm as the iteration progresses. However, it has been shown that the relative residual is almost monotone in both the A-norm and the Euclidean norm, see the work [292].

One of the main advantages of the relative residual (2.136) is its easy and cheap computation. Another still user-defined parameter is the tolerance ε, for which no a priori optimal selection is known and which directly influences the convergence rate of the method and the corresponding

approximate solution. Increasing ε naturally leads to faster computations, but slightly worse results and vice versa. Typically, $\varepsilon = 10^{-6}$ is used, nevertheless the selected value depends on the model problem being solved. Some previous works [16, 17] built upon the discretised linear Laplace operator verified a favourable acceptance range of $\varepsilon \in [10^{-5}, 10^{-3}]$.

Besides a well-chosen tolerance ε, the Krylov subspace solver may be accelerated once more by using an appropriate initialisation instead of the typical choice $\boldsymbol{x}_0 = \mathbf{0}$. In other words, a useful observation on the Krylov subspace methods is that they can obviously benefit a lot from a good educated guess of the solution which could be used as the initial iterate $\boldsymbol{x}_0$ for the CG algorithm. Thus, the initial can be considered as another degree of freedom of the method, which can have an important influence on the rate of convergence. That a suitable initialisation can be computationally advantageous is shown, e.g. in [16] in the context of surface normal integration by solving a discrete Poisson equation. We will take a closer look at this aspect in Chapters 6 and 7.

Overall, the CG method is easy to implement and a cheap procedure in terms of computational complexity and memory requirement. As stated above, there are various problems that make a reliable and efficient application of the method not straightforward. In general, a practical solution can be reached after a small number l of iterations, resulting in a quick termination of the CG method. In practice, however, numerical rounding errors occur and one may suffer from convergence problems for very large systems. Suitable *preconditioning* is therefore recommended in order to enforce all the beneficial properties of the algorithm along with fast convergence at higher resolutions. Preconditioning is a very complex topic with decades of research, for an excellent survey we refer the reader to [42].

Other Krylov Subspace Methods for Regular Matrices As explained previously, the CG method is theoretically limited to the solution of linear systems $A\boldsymbol{x} = \boldsymbol{b}$ with underlying symmetric and positive definite matrix A. In the context of complex applications, it is often impossible to make statements about the properties of the systems that arise. Even for basic model problems such as the convection-diffusion equation, the symmetry of the matrix cannot be guaranteed, as in a similar case in Chapter 7. Consequently, other Krylov subspace methods must be applied which, apart from the regularity of the matrix, initially do not require further requirements for the linear system.

Clearly, all linear problems $A\boldsymbol{x} = \boldsymbol{b}$ with regular matrix A can be translated into a linear system with symmetric and positive definite matrix using the normal equations $A^\top A\boldsymbol{x} = A^\top \boldsymbol{b}$. Unfortunately, it is known that the condition number increases quadratically, since $\mathrm{cond}_2(A^\top A) = \mathrm{cond}_2(A)^2$ and thus the normal equation ansatz is in general a bad idea.

There are many other Krylov subspace methods for solving general linear systems, assuming the system matrix being invertible. One popular Krylov solver is the GMRES method which was developed by Saad and Schultz in 1986. As with CG, the GMRES method is based on the minimisation approach (least squares problem) and does not require the computation of the action of $A^\top$ on a vector. In the procedure the residual is constructed in such a way that it is explicitly minimised over the Krylov subspace in each iteration. As a consequence, large memory is required, so it is usually proposed to restart the algorithm in practice. It should be mentioned that methods based on short-term recurrences such as CG and which also fulfil the minimisation principle cannot be constructed for general matrices.

To overcome this problem, but simultaneously possess the favourable properties such

as matrix-vector multiplications and modest memory space independent of the number of iterations needed, one must therefore deviate from the ideal. In particular, such methods use the residual of the original system to control the termination. The first such method was BICG developed by Fletcher in 1975, which stems from choosing $L_m = K_m^\top = \text{span}\{\boldsymbol{r}_0, A^\top \boldsymbol{r}_0, \ldots, (A^\top)^{m-1}\boldsymbol{r}_0\}$. To avoid multiplications with $A^\top$ in BICG, the CGS method was proposed. On this basis, the *biconjugate gradient stabilized (BiCGSTAB)* method was developed that attempts to smooth the convergence of CGS. All of these methods mentioned are computationally cheap, but their convergence oscillates and a breakdown often occurs. This should be carefully handled with this possibility in mind. Usually, once breakdown has occurred, one can restart the algorithm or transfer the computation to another stable algorithm such as GMRES.

Let us also mention that there is another family of algorithms that generally combine the approaches of GMRES and BICG as a quasi-minimisation idea. The first proposed representative was the QMR method, followed by its transpose-free TFQMR variant and the QMRCGSTAB method derived using BiCGSTAB. These methods may suffer from breakdowns and are more sensitive to rounding errors.

For a detailed description of the Krylov subspace methods for solving nonsymmetric linear systems $A\boldsymbol{x} = \boldsymbol{b}$, we refer to the references mentioned in this chapter. In practice, CGS, BiCGSTAB, TFQMR, QMRCGSTAB are often used, but the right choice of the Krylov subspace solver depends on the model problem, as each of these methods has its advantages and disadvantages.

2.2.2 Preconditioned Conjugate Gradient Method

The introduced CG scheme is the method of choice for problems with sparse symmetric and positive definite matrices. While the CG method can be very efficient, it may only show its full potential for very large systems when combined with a suitable *preconditioning*. In simple terms, preconditioning is used to reduce the condition number and consequently improve the performance of the Krylov solver. The combination of CG and the preconditioning technique is known as the *preconditioned conjugate gradient (PCG)* method. Of course, the convergence of all Krylov subspace methods can be significantly improved by using preconditioners. For nonsymmetric linear systems, a preconditioner is applied to either the left or right of A, which is referred to as *left* and *right* preconditioning. For an overview and a comparison of preconditioned Krylov subspace methods for large sparse nonsymmetric linear systems, we refer to e.g. [102].

The basic idea of preconditioning is to multiply the original system $A\boldsymbol{x} = \boldsymbol{b}$ from the left with a matrix P that is close to A^{-1}. The modified system $PA\boldsymbol{x} = P\boldsymbol{b}$ is generally better conditioned (small condition number close to one), has the same solution and is much more efficient to solve. However, PA is typically not sparse, symmetric and positive definite, so the CG method cannot be used. Therefore, a *two-sided* preconditioning strategy is chosen.

The two-sided preconditioning is applied in such a way that the original system is transformed into the equivalent preconditioned system

$$\begin{aligned} P_L A P_R \boldsymbol{x}^p &= P_L \boldsymbol{b} \\ \boldsymbol{x} &= P_R \boldsymbol{x}^p \end{aligned} \tag{2.137}$$

with regular P_L, $P_R \in \mathbb{R}^{n\times n}$. The task now is to construct the preconditioning in a manner that firstly $P_L A P_R$ approximates the identity matrix I as closely as possible, i.e. $\text{cond}(P_L A P_R) \ll \text{cond}(A)$, secondly P_L, P_R are be easy to compute and thirdly P_L, P_R shall cause a small amount of memory. Apart from that it must be easy to solve the linear systems that involve the matrices P_L and P_R. In total, proper preconditioning reduces the iterations required, but the correspondingly higher cost of an iteration is an essential issue and should not be overlooked. As a result, preconditioning is usually only beneficial for very large systems, where the CG method requires hundreds or thousands of iterations to converge.

When using CG, the two-sided preconditioning with $P_L^\top = P_R$ is useful, because $P_L A P_L^\top$ is again symmetric and positive definite. Moreover, with $P := P_L^\top P_L$ the matrices PA and $P_L A P_L^\top$ have the same eigenvalues. The corresponding preconditioned system

$$A^p \boldsymbol{x}^p = \boldsymbol{b}^p \tag{2.138}$$

is obtained with

$$A^p = P_L A P_L^\top, \quad \boldsymbol{x}^p = P_L^{-T}\boldsymbol{x}, \quad \boldsymbol{b}^p = P_L \boldsymbol{b} \tag{2.139}$$

where $P_L^{-T} := (P_L^{-1})^\top$. If $\boldsymbol{x}_{m+1}^p$, $\boldsymbol{r}_{m+1}^p$, $\boldsymbol{p}_{m+1}^p$ are the iterate, residual and search direction for CG applied to (2.138) then it holds

$$\boldsymbol{r}_{m+1}^p = \boldsymbol{b}^p - A^P \boldsymbol{x}_{m+1}^p \quad \Longrightarrow \quad P_L^{-1}\boldsymbol{r}_{m+1}^p = P_L^{-1}\boldsymbol{b}^p - P_L^{-1}A^P\boldsymbol{x}_{m+1}^p = \boldsymbol{b} - A\boldsymbol{x} \tag{2.140}$$

Consequently, with

$$\begin{aligned} \boldsymbol{x}_{m+1} &= P_L^\top \boldsymbol{x}_{m+1}^p, \quad \boldsymbol{r}_{m+1} = P_L^{-1}\boldsymbol{r}_{m+1}^p, \\ \boldsymbol{p}_{m+1} &= P_L^\top \boldsymbol{p}_{m+1}^p, \quad \boldsymbol{z}_{m+1} = P_L^\top \boldsymbol{r}_{m+1}^p = P_L^\top P_L \boldsymbol{r}_{m+1} = P\boldsymbol{r}_{m+1} \end{aligned} \tag{2.141}$$

one can perform the iteration directly in terms of $\boldsymbol{x}_{m+1}$, A and P. Finally, the PCG algorithm is illustrated in Figure 2.3. It should be noted that steps 2 and 3e) highlight the difference between CG and PCG. In particular, the work associated with PCG is essentially the CG work for one iteration plus the cost of solving the preconditioned system. More precisely, $\boldsymbol{z}_{m+1} = P\boldsymbol{r}_{m+1}$ reflects a linear system, since $P \approx A^{-1}$. Apart from that, a stopping criterion based on α_m is not suitable, because $\alpha_m = \|P_L \boldsymbol{r}_m\|_2^2$ is not a direct evaluation of the residual. Therefore, the relative residual (2.136) is used again.

The main challenge now is to construct a suitable preconditioner P with a good cost balance for the computation of the preconditioner itself and the resulting efficiency per PCG iteration. The simplest preconditioners are the symmetric splitting methods like Jacobi preconditioning, where P is the inverse of the diagonal part of A. When dealing with symmetric matrices, the incomplete Cholesky factorisation is mainly used to build a common and very efficient preconditioner for the CG method. Since, the analysis of incomplete factorisations is both difficult and important, let us briefly describe the technical approach without going into too much detail.

Incomplete Cholesky Factorisation The Cholesky factorisation decomposes the matrix A into $A = CC^\top$ with a lower triangular matrix C, which is relatively simple to implement, very robust and an efficient scheme. On this basis, $\boldsymbol{z}_{m+1} = C^{-\top}C^{-1}\boldsymbol{r}_{m+1}$ must be solved for each PCG iteration via forward and backward substitution. However, the Cholesky factor C

Algorithm 2.2 Preconditioned conjugate gradient method

Input: Matrix A; preconditioner $P = P_L^\top P_L$; right-hand side $\boldsymbol{b}$; initial vector $\boldsymbol{x}_0$

Output: Approximate solution for $A\boldsymbol{x} = \boldsymbol{b}$

1.) $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$

2.) $\boldsymbol{p}_0 = P\boldsymbol{r}_0, \quad \alpha_0 = \langle \boldsymbol{r}_0, \boldsymbol{p}_0 \rangle_2$

3.) **Iterate** for $m = 0, 1, \ldots, n-1$

a) If $\alpha_m \neq 0$ proceed, else STOP

b) $\boldsymbol{v}_m = A\boldsymbol{p}_m, \quad d_m = \frac{\alpha_m}{\langle \boldsymbol{v}_m, \boldsymbol{p}_m \rangle_2}$

c) $\boldsymbol{x}_{m+1} = \boldsymbol{x}_m + d_m \boldsymbol{p}_m$

d) $\boldsymbol{r}_{m+1} = \boldsymbol{r}_m - d_m \boldsymbol{v}_m$

e) $\boldsymbol{z}_{m+1} = P\boldsymbol{r}_{m+1}, \quad \alpha_{m+1} = \langle \boldsymbol{r}_{m+1}, \boldsymbol{z}_{m+1} \rangle_2$

f) $\boldsymbol{p}_{m+1} = \boldsymbol{z}_{m+1} + \frac{\alpha_{m+1}}{\alpha_m} \boldsymbol{p}_m$

Figure 2.3: The PCG scheme for computing an approximate solution to $A\boldsymbol{x} = \boldsymbol{b}$. The basic algorithm stops, if $\boldsymbol{r}_m = \boldsymbol{0}$. In contrast to CG, the PCG algorithm causes extra cost for solving linear systems in steps 2 and 3e).

is normally not sparse and consequently causes high costs for memory storage and solving the underlying triangular linear systems. Another important aspect is the computational effort for the factorisation process performed. In the context of sparse matrices A, preconditioners are typically defined over the same sparse structure of entries as in A. Because of this, a more elaborate preconditioner relies on the sparse Cholesky factorisation, known as *incomplete Cholesky (IC)* factorisation, which constructs a sparse lower triangular matrix $\widetilde{C}$ that is close to C in some sense. As a result, $(\widetilde{C}\widetilde{C}^\top)^{-1}$ and its action on a vector are linked by two sparse triangular solves. The use of an incomplete factorisation was originally proven by Meijerink and Van der Vorst [177].

The complete factorisation of A is given by $A = CC^T + F$. If the lower triangular matrix C is allowed to have nonzero entries anywhere in the lower matrix, then F is the zero matrix and the factorisation is the original one. If only the structure of the entries in A is used to define C, the factorisation will be incomplete. In other words, the lower triangular matrix C might be restricted to have the same nonzero pattern as that of the lower triangular part of A. One natural and simple idea for an incomplete Cholesky factor is to set any entry in C to zero if the corresponding entry in A is also zero.

This approach, also called IC by position, which uses the same sparsity pattern of A for C, is often referred to as *no-fill* IC factorisation or IC(0). The no-fill preconditioner is very easy to implement and less computationally intensive, but more difficult model problems essentially require sophisticated preconditioners that allow for some fill-in. Adding additional

nonzero elements to the sparsity pattern of C potentially improve the closeness between the product CC^T and A. This proceeding is often denoted as numerical fill-in strategy IC(γ), where the parameter $\gamma > 0$ (called *drop tolerance*) describes a dropping criterion, cf. [104,238]. The approach can be explained as follows: new fill-ins are only accepted if the absolute value of the elements is greater than the (local) drop tolerance γ. The drop tolerance approach is an example of IC by value. Adding fill-ins can lead to a better preconditioner and a potentially better convergence rate. On the other hand, it becomes more computationally intensive to solve the underlying triangular systems and the preconditioner itself. Therefore, a good selection of the parameter in the preconditioning method is essential.

Let us mention that from a computational point of view, there is no guarantee that the IC factorisation will exist, as it is possible that the procedure will break down due to a cancellation error. Typically, breakdowns occur if very small pivot elements during the factorisation process arise. For certain classes of matrices such as M-matrices, the existence of the IC factorisation can be ensured so that no breakdown is possible.

Modified Incomplete Cholesky Factorisation When dealing with parabolic PDEs, the *modified incomplete Cholesky (MIC)* factorisation can lead to an even better preconditioner, for an overview of MIC see [42,114,181]. In particular, it can be shown that for IC with small levels of fill-in the condition number of PA still behaves as $\text{cond}(PA) \leq \mathcal{O}(h^{-2})$. However, in some cases this condition number can be improved by using MIC to $\text{cond}(PA) \leq \mathcal{O}(h^{-1})$. Consequently, fewer PCG iterations have to be performed, which generally results in a significant improvement over the unmodified IC factorisation.

The basic idea behind the modification is to force the preconditioner to have the same row sums as the original matrix A. This can be accomplished by adding the discarded fill-ins to the diagonal. The latter approach is known as MIC(0) and can also be combined with the abovementioned drop tolerance strategy to MIC(γ). An essential requirement for the safe use of this technique is that A is a diagonally dominant M-matrix, which is often the case for parabolic model problems. Otherwise, MIC can breakdown which occurs much more frequently compared to unmodified factorisations. This fact indicates why MIC is not as widely used, even though this technique can significantly improve the rate of convergence.

Computational Aspects Using PCG leads to a potentially better convergence rate and speeds up the CG method dramatically. While the basics of the PCG method are relatively simple to implement, it may require fine-tuning of the preconditioner parameter involved. The difficulty is to choose a good value for the drop tolerance. A high drop tolerance yields a dense preconditioner that achieves a better convergence, but is linked to higher computational costs. In contrast, lower values provide faster PCG iterations, but due to worse convergence rates much more iterations are required in total. Usually, a trial and error approach is employed until a satisfactory value of γ is found. In practice, cf. [16,42], good results are obtained for values $\gamma \in [10^{-4}, 10^{-2}]$, the optimal value naturally depends on the model problem.

As indicated above, both factorisations IC and MIC can fail due to possible breakdowns for general symmetric and positive definite matrices. To guarantee the existence of an incomplete factorisation of A, a simple and effective approach is to increase the diagonal dominance of A by diagonal shifts, as proposed by Manteuffel in [173]. More precisely, IC and MIC are applied to the shifted matrix $\tilde{A} = A + \alpha \,\text{diag}(A)$, where $\alpha > 0$ and $\text{diag}(A)$ is the diagonal part of A.

If the factorisation fails again, α is increased until an incomplete factorisation is successfully computed. Obviously, the shifted versions IC and MIC exist because $\tilde{A}$ becomes diagonally dominant for some $\alpha^\star$. However, larger values for α lead to an inefficient preconditioner, so $\alpha \ll \alpha^\star$ is preferred. Since a safe and good α is not known a priori, a trial and error strategy is required again. We mention that the shifted technique is very useful for elliptic PDEs like the Poisson equation equipped with Neumann boundary conditions [16], in which the system matrix is only positive semi-definite. In this situation, the computational costs for computing the preconditioner and solving the underlying triangular systems depend on both γ and α. Thus, a fine-tuned selection of the two parameters is essential.

2.2.3 Multigrid Methods

Another special class of sparse iterative solvers is formulated by the *multigrid (MG)* methods. The main idea of MG is to define a coarsening procedure, more precisely a hierarchy of increasingly coarse grids, until a coarse grid is reached so that the cost of directly solving this coarse problem is negligible. The coarse grid approximation is then transferred back to the fine grid. Although the idea sounds simple, the technical realisation is tricky and specific operators such as smoothing, restriction, interpolation and correction are used within the classic MG process. Originally, MG algorithms were developed to numerically solve time-independent large-scale elliptic boundary value problems for which solutions can be computed highly efficiently. Over time, MG and its variants have been enhanced and successfully applied to various problems in many disciplines. For a detailed introduction we refer to [117, 275, 298].

Besides solving time-independent or stationary PDEs such as the Poisson equation, MG methods are also a particularly good choice for discretised parabolic problems [283]. Let us mention that there are other practically important extensions of MG methods, e.g. algebraic MG, geometric MG, combinatorial MG, lean algebraic MG, hierarchical MG or adaptive basis MG. Apart from that we stress that MG and its variants can be used alone or as preconditioners for iterative Krylov subspace methods. For example, classic MG-based [103, 269], AMG-based [139, 214] as well as CMG-based [151, 158] preconditioners are successfully employed in connection with CG. For a more detailed overview and comparison see e.g. [102, 154].

Of course, MG methods can be very powerful, but implementing them correctly is extremely complex and cumbersome. In contrast, sparse iterative Krylov subspace methods are at the same time considerably easier to use and implement, so MG is not dealt with in this thesis.

2.2.4 Summary

Overall, sparse direct methods with an effective factorisation are the first choice for solving sparse linear system $A\boldsymbol{x} = \boldsymbol{b}$ until computer memory and CPU time become excessive. For fine discretised two-dimensional problems or three-dimensional problems, the size of the linear systems typically exceeds the order of $\mathcal{O}(10^6)$ and the computational costs in time and memory space increase sharply. Therefore, sparse direct methods are not considered viable for solving very large linear systems.

In this context, sparse iterative methods are a popular tool where most of these methods are well described in many textbooks and are relatively easy to implement (except MG methods). Compared to direct (factor-solve) methods, iterative methods are data dependent, need to be coupled with an effective preconditioner to achieve high performance, and are often to be

tailored to specific system types in order to converge well. Hence, more expertise is required as one has to choose the Krylov subspace method and the preconditioner, which is usually problem-dependent. In addition, it is difficult to properly choose optimal preconditioning parameters. As a consequence, a significant amount of time is needed to find a reasonable combination of the Krylov subspace method and the preconditioner through typical trial and error approaches.

In the case of symmetric and positive definite system matrices A, this selection procedure can be neglected, since it is known that PCG is widely recognised as the best Krylov method. The IC factorisation is commonly used in combination with the CG method, whereby the modified variant MIC is even a more efficient preconditioner for special classes of matrices. The use of IC and MIC accelerates the CG method drastically, however, a good choice of parameters (drop tolerance γ; shift parameter α) is essential in the preconditioning method. Therefore, it will require a thorough study to identify the most useful parameters.

In total, a sparse iterative solver with an optimised preconditioner (and a suitable initialisation instead of $\boldsymbol{x}_0 = \mathbf{0}$) can efficiently solve extremely large systems. Another important aspect is the flexible handling of the desired accuracy of the approximate solution by means of the relative residual ε. For special model problems an ε with a large value is accepted, so fewer iterations are needed. Consequently, in such situations the iterative solver can outperform the direct solver even for small systems, cf. Chapter 5.

As already indicated, the greatest challenge with sparse direct and sparse iterative methods is the required factorisation rather than the solution of the resulting factorised system. Ideally, the (incomplete) factors should be easy to compute and require a modest amount of storage space. Fortunately, most factorisations are based on existing sophisticated software packages and are also generally available or importable into software like MATLAB. On this basis, it is possible to find the best solver for the underlying model problem.

Obviously, both direct and iterative solvers lose their high performance when considering nonlinear PDEs in which the spatial discretisation yields a nonlinear ODE system. In the case of nonlinear systems, the underlying system matrix is nonconstant and a one-time factorisation cannot be applied. To use implicit schemes such as IE and CN, one can employ popular nonlinear system solvers e.g. the Newton-Krylov methods [145, 146]. The latter technique combines Newton's method with a sparse iterative linear solver that is inherently computationally intensive.

2.3 Exponential Integrators

Lastly, we consider an alternative class of numerical methods for solving stiff ODE systems, namely the *exponential integrators* or the so-called *exponential time differencing*. This class of methods is based on the exact integration of the given initial value problem

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + \boldsymbol{w}(t), \quad t \in (0, t_F],\ \boldsymbol{u}(0) = \boldsymbol{u}^0,\ \boldsymbol{w}(0) = \boldsymbol{w}^0 \tag{2.142}$$

The exact integration of (2.142) yields the corresponding integral equation

$$\boldsymbol{u}(t) = e^{Lt}\boldsymbol{u}^0 + \int_0^t e^{L(t-z)}\boldsymbol{w}(z)\,\mathrm{d}z \tag{2.143}$$

Using a temporal discretisation, the exact solution (2.143) at time t_{k+1} is then given by

$$\boldsymbol{u}(t_{k+1}) = e^{L\tau}\boldsymbol{u}(t_k) + \int_0^\tau e^{L(\tau-z)}\boldsymbol{w}(t_k+z)\,\mathrm{d}z = e^{L\tau}\boldsymbol{u}(t_k) + e^{L\tau}\int_0^\tau e^{-Lz}\boldsymbol{w}(t_k+z)\,\mathrm{d}z \quad (2.144)$$

with the uniform time step size $\tau = t_{k+1} - t_k$. A numerical scheme is now obtained by approximating the integral in (2.144) using an appropriate quadrature rule. The simplest approximation is constructed under the assumption that $\boldsymbol{w}(t_k)$ is constant within the interval $[t_k, t_{k+1}]$, which is often referred to as the *exponential Euler method*. Consequently, it holds

$$\int_0^\tau e^{-Lz}\boldsymbol{w}(t_k+z)\,\mathrm{d}z = \int_0^\tau e^{-Lz}\boldsymbol{w}(t_k)\,\mathrm{d}z = \boldsymbol{w}(t_k)\int_0^\tau e^{-Lz}\,\mathrm{d}z = \boldsymbol{w}(t_k)\left(\frac{-e^{-L\tau}+1}{L}\right) \quad (2.145)$$

and with (2.144) the numerical scheme leads to

$$\boldsymbol{u}^{k+1} = e^{L\tau}\boldsymbol{u}^k + e^{L\tau}\left(\frac{-e^{-L\tau}+1}{L}\right)\boldsymbol{w}^k = e^{L\tau}\boldsymbol{u}^k + \left(\frac{e^{L\tau}-1}{L}\right)\boldsymbol{w}^k \quad (2.146)$$

For a more detailed introduction and discussion of exponential integrators, we refer the reader to [64, 135–137, 186] and the references therein. On closer inspection, the exponential matrix operator within the numerical scheme (2.146) is treated exactly. Obviously, the main effort of the scheme is the computation of the matrix exponential $e^{L\tau}$. For this reason, an efficient method for computing the matrix exponential is of great importance.

For the sake of simplicity let us assume $\boldsymbol{w}(t) = \boldsymbol{0}$. The solution of the corresponding homogeneous semi-discretised system reads

$$\boldsymbol{u}(t) = e^{Lt}\boldsymbol{u}^0 \quad (2.147)$$

Although the latter analytical solution has a simple form, the problem of computing the exponential of a matrix is an omnipresent issue in scientific computing and is therefore still relevant today. The exponential function of a matrix is defined by the following series:

$$e^{Lt} = \sum_{k=0}^{\infty}\frac{(Lt)^k}{k!} \quad (2.148)$$

where the power series is uniformly convergent [32], meaning that e^{Lt} is well defined for all t and L. The naive approach of computing an approximation of e^{Lt} by truncating the series (2.148) after the first k terms is known to be one of the worst practices cf. [187]. Many methods for computing e^{Lt} have been introduced in the past, for an excellent survey see [187]. The most popular methods for computing an approximation of the matrix exponential are certainly the Padé approximation, the scaling and squaring method (based on Padé or Taylor series approximation), the Chebyshev polynomials, the matrix decomposition methods or the Krylov subspace methods. However, the techniques mentioned (with the exception of the matrix decomposition methods and the Krylov subspace methods) are generally only suitable in computing exponentials of small matrices and are thus not a viable option for sparse large discrete Laplacians as in our case.

Because of this, we only give a brief discussion of the methods based on matrix decomposition and Krylov subspace techniques. The former methods are of interest for some simple model problems. The Krylov methods are well applicable to general problems, especially they approximate the product of a matrix exponential function with a given vector rather than computing the matrix exponential directly.

2.3.1 Matrix Decomposition Methods

In some special applications in the field of image processing and computer vision, the underlying semi-discretised model problem must be solved multiple times e.g. for problems with many different initial conditions. In this situation, it can be useful to approximate the matrix exponential using an eigendecomposition of the system matrix and consider a reduced order model based on the dominant eigenvalues and eigenvectors. Let us give a short insight.

One possible efficient method for problems with large sparse matrices and repeated evaluations of e^{Lt} are those which are based on matrix decompositions. Assuming that L can be decomposed in the form $L = PSP^{-1}$, then the matrix exponential can be written as

$$e^{Lt} = Pe^{St}P^{-1} \tag{2.149}$$

The key idea of the latter approach is to find a suitable decomposition for which e^{St} is easily computable. In this context, standard approaches can be based on eigendecomposition, QR algorithms, Jordan canonical form or Schur decomposition, see [187].

The simplest technique is to use an eigendecomposition. Let $L \in \mathbb{R}^{n\times n}$ be diagonalisable, then matrices $P = \Phi$ and $S = \Lambda$ exist, where any diagonal element of Λ is an eigenvalue for L and the corresponding column of Φ is an eigenvector for this eigenvalue. On this basis, the solution (2.147) can easily be specified via

$$\boldsymbol{u}(t) = e^{Lt}\boldsymbol{u}^0 = \Phi e^{\Lambda t}\Phi^{-1}\boldsymbol{u}^0 = \sum_{i=1}^{n} e^{\lambda_i t}\boldsymbol{\phi}_i\widetilde{\boldsymbol{\phi}}_i^{\top}\boldsymbol{u}^0 \tag{2.150}$$

where $\widetilde{\boldsymbol{\phi}}_i^{\top}$ is the i-th row of Φ^{-1}. To ensure stability, the inequality

$$\|\boldsymbol{u}(t)\| \leq \left\|e^{tL}\right\| \left\|\boldsymbol{u}^0\right\| \leq C \left\|\boldsymbol{u}^0\right\|, \quad \forall t \geq 0 \tag{2.151}$$

must be satisfied with a positive constant C. Suppose L only has nonpositive eigenvalues, it follows that

$$\|\boldsymbol{u}(t)\| \leq \left\|\Phi e^{\Lambda t}\Phi^{-1}\right\| \left\|\boldsymbol{u}^0\right\| \leq \|\Phi\| \left\|e^{\Lambda t}\right\| \left\|\Phi^{-1}\right\| \left\|\boldsymbol{u}^0\right\| = \operatorname{cond}(\Phi) \max_k \left|e^{\lambda_k t}\right| \left\|\boldsymbol{u}^0\right\| \tag{2.152}$$

Since the eigenvalues are negative, it holds $\lim\limits_{t\to\infty} e^{\lambda_k t} = 0$ for all k, or $e^{\lambda_0 t} = 1$ for a zero eigenvalue λ_0. Clearly, $e^{\lambda_k t} = 1$ for $t = 0$ so that $\max\limits_k |e^{\lambda_k t}| \leq 1$. Therefore, with $\operatorname{cond}(\Phi) = \|\Phi\|\|\Phi^{-1}\| \leq C$ the inequality (2.151) is true.

It should be emphasised that the major drawback of this approach is the computation of all eigenvalues and eigenvectors, which is considered to be very computationally intensive. However, in certain cases, for example in the field of geometry processing and shape analysis

(see Chapter 5) it is sufficient to consider only a small dominant subset of the spectrum, i.e.

$$\boldsymbol{u}(t) \approx \sum_{i=1}^{r} e^{\lambda_i t} \boldsymbol{\phi}_i \widetilde{\boldsymbol{\phi}}_i^{\top} \boldsymbol{u}^0, \quad r \ll n \tag{2.153}$$

Obviously, the stability is preserved because the eigenvalues within the truncated approximation (2.153) are contained in the original spectrum. The use of only a few dominant modes speeds up the computations drastically, but is linked with less accurate approximations. Consequently, the truncated approximation is typically practicable when a lower number of modes are able to capture the important contributions of the original physical system responses. This technique is often applied to simple PDEs without source terms and complex boundary conditions.

2.3.2 Krylov-Based Matrix Exponential Approximation

In many applications the underlying matrix L is large and sparse. Unfortunately, although the discrete Laplacian is sparse, the matrix e^{Lt} is usually dense, which aims to avoid this intensive computation. In fact, only the product $e^{Lt}\boldsymbol{u}^0$ needs to be computed rather than the exponential of the full matrix. In this case, the powerful Krylov subspace methods have become very important and are preferred over traditional methods for large-scale problems. Based on the fact that Krylov subspace methods generally only require sparse matrix-vector multiplications to compute an approximation, we discuss the basic idea and the implementation below. Further details on the description and analysis can be found in the works [98, 134, 237].

The principal idea of the Krylov subspace methods is to approximate the original large sparse matrix L by a matrix $H_m \in \mathbb{R}^{m\times m}$ with $m \ll n$, so that the computational costs for the construction of the matrix exponential e^{H_m} are comparatively low. To this end, the aim is to find the best approximation to the matrix exponential operation $e^L\boldsymbol{v}$ in the form of

$$e^L\boldsymbol{v} \approx p_{m-1}(L)\boldsymbol{v} \tag{2.154}$$

where $\boldsymbol{v}$ is any nonzero vector and p_{m-1} is a polynomial of degree $m-1$ in L that is a linear combination of the vectors $\boldsymbol{v}, L\boldsymbol{v}, \ldots, L^{m-1}\boldsymbol{v}$ and thus an element of the Krylov subspace

$$\mathcal{K}_m\left(L, \boldsymbol{v}\right) = \operatorname{span}\left\{\boldsymbol{v}, L\boldsymbol{v}, \ldots, L^{m-1}\boldsymbol{v}\right\} \tag{2.155}$$

Since this approximation of $e^L\boldsymbol{v}$ is an element of the Krylov subspace, the problem can be reformulated to find an element of $\mathcal{K}_m(L, \boldsymbol{v})$. Notably, using the vectors $L^j\boldsymbol{v}$ as the basis itself is not a good idea as the vectors naturally become almost linearly dependent, meaning that they point in almost the same direction as the dominant eigenvector of L based on the power iteration properties. To find an appropriate basis, the numerically stable Arnoldi algorithm is usually used, which is based solely on simple matrix-vector multiplications. The Arnoldi algorithm is presented in Figure 2.4. To construct a basis $V_m \in \mathbb{R}^{m\times m}$ of $\mathcal{K}_m$ and an upper Hessenberg matrix H_m, the Gram-Schmidt orthonormalisation process is normally applied. More precisely, the Arnoldi algorithm constructs

$$LV_m = V_m H_m + h_{m+1,m}\boldsymbol{v}_{m+1}\boldsymbol{e}_m^{\top}, \quad \boldsymbol{e}_m = (0, \ldots, 0, 1)^{\top} \in \mathbb{R}^m \tag{2.156}$$

Algorithm 2.3 Arnoldi algorithm

Input: Matrix A; vector $\boldsymbol{v}$; Krylov subspace order m

Output: $V_m = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m]$; $H_m = (h_{i,j}) \in \mathbb{R}^{m \times m}$

1.) $\boldsymbol{v}_1 = \dfrac{\boldsymbol{v}}{\|\boldsymbol{v}\|_2}$

2.) **Iterate** for $j = 1, 2, \ldots, m$

a) $\boldsymbol{w} = A\boldsymbol{v}_j$

b) **Iterate** for $i = 1, 2, \ldots, j$

i) $h_{i,j} = \langle \boldsymbol{w}, \boldsymbol{v}_i \rangle_2$

ii) $\boldsymbol{w} = \boldsymbol{w} - h_{i,j}\boldsymbol{v}_i$

c) $h_{j+1,j} = \|\boldsymbol{w}\|_2, \quad \boldsymbol{v}_{j+1} = \dfrac{\boldsymbol{w}}{h_{j+1,j}}$

Figure 2.4: Computation of a sequence of orthonormal vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m$ so that these vectors span the Krylov subspace $\mathcal{K}_m = \{\boldsymbol{v}, A\boldsymbol{v}, A^2\boldsymbol{v}, \ldots, A^{m-1}\boldsymbol{v}\}$ and an upper Hessenberg matrix $H_m \in \mathbb{R}^{m \times m}$.

with an orthonormal matrix V_m and the vector $\boldsymbol{v}_{m+1}$ satisfying $V_m^\top \boldsymbol{v}_{m+1} = \boldsymbol{0}$ such that $H_m = V_m^\top L V_m$ is obtained. Since for Krylov subspaces it holds that $\mathcal{K}_m(tL, \boldsymbol{v}) = \mathcal{K}_m(L, \boldsymbol{v})$ for any arbitrary scalar t, V_m remains unchanged and only H_m is replaced by tH_m. Therefore, there is no loss of generality when directly applying the Arnoldi procedure to L. Let us now concretise how the approximate solution $p_{m-1}(tL)\boldsymbol{v}$ can be computed in a simple manner.

Let $\boldsymbol{w}_{\text{opt}}$ be the optimal Krylov approximation in the least squares sense to $\boldsymbol{w}(t) = e^{tL}\boldsymbol{v}$. Since V_m is a basis of the Krylov subspace, $\boldsymbol{w}_{\text{opt}}$ can be written as $\boldsymbol{w}_{\text{opt}} = V_m \boldsymbol{y}_{\text{opt}}$ with $\boldsymbol{y}_{\text{opt}} \in \mathbb{R}^m$. Using this fact, the goal is now to determine $\boldsymbol{w}_{\text{opt}}$ by minimising

$$\|\boldsymbol{w}(t) - \boldsymbol{w}_{\text{opt}}\|_2 = \min_{\boldsymbol{x} \in \mathcal{K}_m(tL, \boldsymbol{v})} \|\boldsymbol{w}(t) - \boldsymbol{x}\|_2 = \min_{\boldsymbol{y} \in \mathbb{R}^m} \|\boldsymbol{w}(t) - V_m \boldsymbol{y}\|_2 \tag{2.157}$$

Obviously, $\boldsymbol{y}_{\text{opt}} = V_m^\top e^{tL}\boldsymbol{v}$ is the solution to this problem. To avoid e^{tL} when computing $\boldsymbol{y}_{\text{opt}}$, the term $\boldsymbol{v} = \beta \boldsymbol{v}_1$ with $\beta = \|\boldsymbol{v}\|_2$ is used. This implies

$$\boldsymbol{y}_{\text{opt}} = \beta V_m^\top e^{tL} \boldsymbol{v}_1 = \beta V_m^\top e^{tL} V_m \boldsymbol{e}_1 \tag{2.158}$$

where the first basis vector is specified via $\boldsymbol{v}_1 = V_m \boldsymbol{e}_1$ and $\boldsymbol{e}_1 = (1, 0, \ldots, 0)^\top \in \mathbb{R}^m$. Using now $H_m = V_m^\top L V_m$ with $V_m^\top e^{tL} V_m = e^{V_m^\top tLV_m} = e^{tH_m}$, the approximation can finally be represented by

$$e^{tL}\boldsymbol{v} \approx \boldsymbol{w}_{\text{opt}} = V_m \boldsymbol{y}_{\text{opt}} = \beta V_m e^{tH_m} \boldsymbol{e}_1 \tag{2.159}$$

As a result, the matrix exponential of tL is approximated by the exponential over the much smaller matrix H_m. On this basis, the reduced exponential e^{tH_m} can be computed by any

suitable scheme for matrices of moderate size such as the popular scaling and squaring algorithm [130]. It should be stressed that the Krylov subspace concept works well, since the eigenvalues of H_m (also called Ritz values) strongly match with the well-separated extreme eigenvalues of L, see [273]. In simple terms, H_m preserves one important property, namely the extreme eigenvalues of L, which are useful in approximating the matrix exponential e^{tL}. The latter is shown in [81] using Schwerdtfeger's formula.

Although this approach introduced by Saad [98, 237] causes very low computational costs, it shows a major weakness with regard to the accuracy of the approximate solution (2.159) when using large values of t. Let us briefly discuss the theoretical findings related to stability and error analysis is some detail.

Stability As we know, when dealing with stiff ODE systems the stability property is an important aspect for the computation of an efficient approximate solution. In particular, explicit time integration methods suffer from stability requirements for the time step size and lose their benefits of using sparse matrix-vector multiplications. In contrast, implicit schemes require to solve systems of linear equations which is coupled with significant computational costs. In this framework, it can be shown that matrix exponential approximations using Krylov subspace methods are unconditionally stable in the Euclidean norm for negative semi-definite matrices. This requires a more general concept based on the logarithmic norm.

A useful concept for the stability analysis of continuous dynamical systems is the *logarithmic norm* introduced in [70], for further details see also [138, 265, 268]. The logarithmic norm of a matrix A is defined as

$$\mu(A) = \lim_{h \to 0^+} \frac{\|I + hA\| - 1}{h} \tag{2.160}$$

for which the limit exists and $\mu(A)$ is well-defined. It should be noted that the logarithmic norm, despite its name, is not a matrix norm because $\mu(A)$ can be negative. The following property of the logarithmic norm is crucial for deriving error bounds for initial value problems:

Theorem 2.3 ([138]). *Let $A \in \mathbb{C}^{n \times n}$ and $t \geq 0$. The matrix exponential is bounded by*

$$\left\| e^{tA} \right\| \leq e^{t\mu(A)} \tag{2.161}$$

More precisely, this means that a continuous system has stable solutions $e^{Lt}\boldsymbol{u}^0$ if $\mu(L) \leq 0$. Furthermore, the logarithmic norm can also be expressed by means of the inner product $\langle \cdot, \cdot \rangle$ in the form

$$\mu(A) = \sup_{\boldsymbol{x} \neq 0} \frac{\mathrm{Re}\langle \boldsymbol{x}, A\boldsymbol{x} \rangle}{\langle \boldsymbol{x}, \boldsymbol{x} \rangle} \tag{2.162}$$

which serves as an alternative definition. Consequently, the Euclidean inner product yields

$$\mu_2(A) = \sup_{\boldsymbol{x} \neq 0} \frac{\mathrm{Re}\langle \boldsymbol{x}, A\boldsymbol{x} \rangle_2}{\langle \boldsymbol{x}, \boldsymbol{x} \rangle_2} = \max \left\{ \lambda : \lambda \in \sigma \left(\tfrac{1}{2}(A + A^*) \right) \right\} \tag{2.163}$$

and for real matrices it holds that

$$\mu_2(A) \leq 0 \quad \Longleftrightarrow \quad \langle \boldsymbol{x}, A\boldsymbol{x} \rangle_2 \leq 0, \, \forall \boldsymbol{x} \in \mathbb{R}^n \tag{2.164}$$

For the stability analysis of Krylov-based matrix exponentials, the following lemma is required:

Lemma 2.1 ([98]). *For any real matrix $A \in \mathbb{R}^{n\times n}$ and corresponding upper Hessenberg matrix $H_m \in \mathbb{R}^{m\times m}$ generated by the Arnoldi algorithm holds*

$$\mu_2(H_m) \leq \mu_2(A) \tag{2.165}$$

Proof. By construction, V_m is an orthogonal matrix, H_m satisfies $H_m = V_m^\top A V_m$ and the inequality $\max_k \lambda_k(V_m^\top A V_m) \leq \max_k \lambda_k(A)$ is true. Therefore, it easily follows that

$$\mu_2(H_m) = \max_k \lambda_k \left(\frac{V_m^\top A V_m + V_m^\top A^\top V_m}{2} \right) \leq \max_k \lambda_k \left(\frac{A + A^\top}{2} \right) = \mu_2(A) \tag{2.166}$$

□

Based on the latter foundations, the unconditional stability of the Krylov subspace method can be shown by the theorem below:

Theorem 2.4 ([98]). *The matrix exponential operation $p_{m-1}(tL)\boldsymbol{v} = \boldsymbol{w}_{opt}$ computed by the Krylov subspace approximation* (2.159) *is unconditionally stable for negative semi-definite matrices L in the Euclidean norm.*

Proof. To ensure unconditional stability, the term $\|\beta V_m e^{tH_m} \boldsymbol{e}_1\|_2$ must be bounded for all $t \geq 0$. Since V_m is orthogonal, one has

$$\|\boldsymbol{w}_{\text{opt}}\|_2 = \left\| \beta V_m e^{tH_m} \boldsymbol{e}_1 \right\|_2 \leq \beta \left\| e^{tH_m} \right\|_2 \tag{2.167}$$

Setting $\beta = C$ and using (2.161), (2.165) as well as (2.164) finally yields

$$\beta \left\| e^{tH_m} \right\|_2 \leq C e^{\mu_2(tH_m)} \leq C e^{\mu_2(tL)} \leq C \tag{2.168}$$

so that $\|\boldsymbol{w}_{\text{opt}}\|_2 \leq C$ is true for all $t \geq 0$. □

The latter theorem generally holds in the case that e^{tH_m} is evaluated exactly. Otherwise, the stability depends on the calculation method used. Due to the unconditional stability, the time step size t of the Krylov-based matrix exponential computation is free from restrictions. At first glance, it appears that these approaches represent the best practise for solving stiff ODEs numerically. However, it can be shown that the approximation error depends on $\|tL\|_2$, which strongly limits the effectiveness of the approximation technique.

Error Analysis Using the approximation techniques described above, several practical questions arise. For instance, how large should the dimension m be chosen to obtain highly accurate approximations for a given $t, \boldsymbol{v}$ and L. Obviously, for $m = n$, the Krylov approximation is exact, since $\boldsymbol{v}_{m+1} = \boldsymbol{0}$ and it becomes $AV_m = V_m H_m$. In the case $m \ll n$, error bounds for the computation of $e^{tL}\boldsymbol{v}$ using the Krylov subspace method (2.159) are given a priori by the following theorem:

Theorem 2.5 ([98]). *Let A be any matrix and let $\rho = \|tA\|_2$, $\beta = \|\boldsymbol{v}\|_2$ and $\eta = \mu(tA)$. Then the error of the approximation* (2.159) *with respect to the spectral norm is given by*

$$\left\| e^{tL}\boldsymbol{v} - \beta V_m e^{tH_m} \boldsymbol{e}_1 \right\|_2 \leq 2\beta \frac{\rho^m e^\rho}{m!} \tag{2.169}$$

A sharper error bound holds for the logarithmic norm:

$$\left\| e^{tL}\boldsymbol{v} - \beta V_m e^{tH_m}\boldsymbol{e}_1 \right\|_2 \leq 2\beta \frac{\rho^m}{m!} \max(1, e^{\eta}) \tag{2.170}$$

since $\mu(A) \leq \|A\|$.

Some more sophisticated and refined error bounds based on the Arnoldi method were studied in [134]. In practice, the computation of the error bounds is expensive, alternatively [98] proposes a posteriori error estimation given by

$$E(m,t) = \beta h_{m+1,m} \boldsymbol{e}_m^{\top} \varphi(tH_m)\boldsymbol{e}_1 \boldsymbol{v}_{m+1} \tag{2.171}$$

with $\varphi(x) = \frac{e^x - 1}{x}$.

The bounds (2.169) and (2.170) indicate that the error depends on m and $\|tL\|_2$. Thus, by reducing t, the scheme produces more accurate approximations without changing the dimension of m. For stiff problems, however, it is known that the spectral norm of L is typically very large. This may limit the time step parameter t to be too small, in general $t \ll 1$ must be used. The Krylov subspace technique can therefore be considered as an explicit ODE solver. Alternatively, by increasing the dimension m, a larger t can be used while preserving accuracy. Nevertheless, the cost of computing and storing the Krylov basis vectors $\boldsymbol{v}_m$ increases. In addition, the practical calculation of e^{tH_m} becomes more computationally intensive. Consequently, a good selection of t and m is essential.

Due to the latter observations, a more efficient approach [257] is to apply the Krylov-based matrix exponential approximation iteratively rather than computing the approximation in one-step. This means that the time integration is split into a sum $t = \tau_1 + \tau_2 + \cdots + \tau_q$ of smaller time steps τ_j, so that $\boldsymbol{u}(t)$ is computed with

$$\boldsymbol{u}(t) = e^{tL}\boldsymbol{u}^0 = e^{(\tau_1+\tau_2+\cdots+\tau_q)L}\boldsymbol{u}^0 = e^{\tau_q L}\Big(\ldots \Big(e^{\tau_2 L}\Big(e^{\tau_1 L}\boldsymbol{u}^0 \Big)\Big)\Big) \tag{2.172}$$

This is equivalent to compute iteratively

$$\boldsymbol{u}^j = e^{\tau_j L}\boldsymbol{u}^{j-1}, \quad j = 1, \ldots, q \tag{2.173}$$

which normally requires an adaptive procedure based on (2.171) to control the errors of the method depending on τ_j and m. From (2.173) it follows that the Krylov subspace $\mathcal{K}_m(L, \boldsymbol{u}^{j-1})$ has to be recomputed at each time level, which strongly influences the efficiency of this technique. An algorithm for the procedure (2.173) is shown in Figure 2.5.

Preconditioning As mentioned earlier, the matrix H_m constructed by the Arnoldi algorithm generally preserves the dominant eigenvalues with large magnitude of L. However, the eigenvalues with small magnitude are mostly more relevant for the approximation of e^{tL}. Therefore, the Arnoldi process requires a large dimension m to capture the important (small) eigenvalues. To overcome this problem, a transformation of the underlying spectrum [278] can be useful in order to determine the dominant (small) eigenvalues more quickly.

The basic idea is to apply the Arnoldi method to the matrix $(I - \gamma L)^{-1}$ instead of L, where $\gamma > 0$ is a user-defined parameter. Accordingly, this means that the transformed matrix can be considered as a kind of preconditioning. The corresponding Krylov subspace is then

Algorithm 2.4 Krylov-based matrix exponential approximation

Input: Matrix L; initial condition $\boldsymbol{u}^0$; stopping time t_F; error tolerance ε

Output: $\boldsymbol{u}(t_F)$

1.) $t = 0, \boldsymbol{v} = \boldsymbol{u}^0$

2.) $\beta = \|\boldsymbol{v}\|_2$

3.) **While** $t \leq t_F$ **do**

a) Construct V_m and H_m by Algorithm 2.3 until $E(m, \tau) \leq \varepsilon$ using adaptive procedure

b) $\boldsymbol{u}(t + \tau) = \beta V_m e^{\tau H_m} \boldsymbol{e}_1$

c) $t := t + \tau, \boldsymbol{v} = \boldsymbol{u}(t + \tau)$

d) $\beta = \|\boldsymbol{v}\|_2$

Figure 2.5: Solving (2.173) using the Krylov-based matrix exponential approximation (2.159). In addition, a suitable adaptive procedure is needed in order to control the error of the method depending on the time step size τ and the dimension m.

generated via

$$\mathcal{K}_m \left((I - \gamma L)^{-1}, \boldsymbol{v} \right) = \text{span} \left\{ \boldsymbol{v}, (I - \gamma L)^{-1} \boldsymbol{v}, \ldots, (I - \gamma L)^{-(m-1)} \boldsymbol{v} \right\} \tag{2.174}$$

It should be noted that the adaptation of the Arnoldi process in Algorithm 2.3 using (2.174) is straightforward. To this end, the matrix-vector multiplication in step 2a) is replaced by $\boldsymbol{w} = (I - \gamma A)^{-1} \boldsymbol{v}_j$. Consequently, the approximation on the transformed matrix reads

$$(I - \gamma L)^{-1} V_m = V_m H_m + h_{m+1,m} \boldsymbol{v}_{m+1} \boldsymbol{e}_m^\top, \quad \boldsymbol{e}_m = (0, \ldots, 0, 1)^\top \in \mathbb{R}^m \tag{2.175}$$

In this regard, the function

$$f_\gamma^t(h) = \exp \left(\frac{t(1 - h^{-1})}{\gamma} \right), \quad \text{for} \quad h \in (0, 1], \quad f_\gamma^t(0) = 0 \tag{2.176}$$

is employed, which fulfils $f_\gamma^t((I - \gamma L)^{-1}) = e^{tL}$. On this basis, the approximation of $e^{tL} \boldsymbol{v}$ is given by

$$e^{tL} \boldsymbol{v} \approx \beta V_m f_\gamma^t(H_m) \boldsymbol{e}_1 = \beta V_m e^{t \widetilde{H}_m} \boldsymbol{e}_1 \tag{2.177}$$

where $\widetilde{H}_m = \frac{1}{\gamma}(I - H_m^{-1})$ is usually constructed with a small m. In contrast to the nonpreconditioned approximation (2.159), the solution of large sparse systems of linear equation is now required. In practice, a factorisation of $I - \gamma L$ is only computed once, so a fine-tuned value γ is advantageous. Some remarks for an appropriate selection of γ can be found in [278].

Another difference to the basic method (2.159) concerns the a priori error bound. More

precisely, the error estimate is independent of the norm of $\|tL\|$ and only the first (smallest in magnitude) eigenvalue of L is a significant factor for the convergence of this method. For this reason, an iterative computation like (2.173) cannot be avoided. Therefore, the Krylov subspace $\mathcal{K}_m((I-\gamma L)^{-1}, \boldsymbol{u}^{j-1})$ has to be constructed again at each time level, but only a small dimension m is required for a good approximate solution. To control the time step size τ_j, an adaptive procedure can be used which is based on the following posteriori error bound:

$$\widetilde{E}(m,t) = \frac{\beta h_{m+1,m}}{\gamma} \left| (I-\gamma L)\boldsymbol{v}_{m+1}\boldsymbol{e}_m^\top H_m^{-1} e^{t\widetilde{H}_m}\boldsymbol{e}_1 \right| \tag{2.178}$$

In total, the algorithm for the preconditioned Krylov-based approximation remains structurally unchanged compared to the basic Algorithm 2.4, only a factorisation $(I-\gamma L)$ with a suitable γ is needed and the error bound (2.171) is replaced by (2.178).

2.3.3 Summary

The key element of exponential integrators is the efficient approximation of matrix exponentials. A simple and effective technique can be based on eigendecomposition, but this proceeding is practically limited to certain model problems.

In contrast, Krylov subspace techniques are well applicable to general model problems. While these methods are unconditionally stable, the approximation accuracy depends on the norm $\|tL\|$. Since the spectral norm of L is typically very large for stiff problems, the Krylov technique can be understood as explicit integrators that suffer from accuracy for large time steps t. In this situation, a preconditioning can be useful, nevertheless both Krylov variants require an iterative computation procedure. As a consequence, the Krylov basis has to be recomputed at each time level, which leads to high computational costs.

One advantage of the Krylov-based matrix exponential approximation is that they are well-suited for highly oscillatory problems with purely imaginary eigenvalues of large modulus, as well as for nonlinear problems see [137].

Chapter 3

Fast Explicit Methods

As shown in Section 2.1, explicit schemes like the EE method (2.6) offer a very simple way to solve parabolic heat or diffusion equations. They are based solely on cheap sparse matrix-vector multiplications and can be easily accelerated with GPUs. However, the allowed time step size is limited by a rather small upper bound which is given by the numerical stability condition. Because of the time step size restriction, explicit schemes are generally considered to be unsuitable for many practical applications, especially for long-term simulations of parabolic-type equations where the underlying ODE system is typically stiff. To overcome this problem while exploiting the advantages of an explicit scheme, acceleration techniques have been developed which are based on extended stability regions along the negative real axis. In this way, stiff problems can be numerically integrated using simple explicit evaluations that would normally require the use of implicit methods. In the past, several elegant strategies have been introduced [99,108,244,281], which conceptually belong to the class of RK methods. Let us mention that there is also a novel framework [24] which is defined by time-accurate and highly-stable explicit operators that act as preconditioners on the stiff terms and can be employed straightforwardly to any existing explicit methods.

3.1 Introduction

In the following sections we provide a more detailed overview and description of fast explicit methods known as *Runge-Kutta-Chebyshev (RKC)* or *super time stepping (STS)* methods, which are simple and effective to speed up explicit schemes for parabolic or hyperbolic-parabolic equations with dominant diffusion. Exactly for this problem class, these methods offer a very attractive alternative for unconditionally stable implicit ones, assumed that the corresponding eigenvalues lie in a long narrow strip along the negative real axis. The main idea is to develop an explicit method whose stability region extends along into the left half-plane as far as possible. In doing so, an s-stage RK scheme in connection with Chebyshev polynomials forms the basis for the construction of an extended stability region. In particular, the number of stages s is specified in such a way that the stability region of the underlying stability polynomial is as large as possible rather than increasing the order of the accuracy of the method. Although the stability interval for ensuring absolute stability is finite, this interval is much larger than with standard explicit methods. More precisely, the RKC methods possess stretched real stability intervals with a length proportional to s^2, so that the maximal stability domain on the negative real axis increases quadratically with the number of stages s. The quadratic dependence is therefore the crucial factor for the success of the RKC methods. Based on this derivation, the multi-stage RKC schemes remain explicit and, compared to implicit methods, do not require the solution of sparse large linear (or

nonlinear) algebraic equations when solving multi-dimensional problems. In addition, these methods can easily be applied to large problem classes with little computational effort and memory demand, and they avoid undesirable splitting errors like operator splitting methods.

This class of schemes has been successfully used in the context of nonlinear/anisotropic heat conduction in magneto- and radiation hydrodynamics (computational astrophysics) applications, e.g. [54, 63, 182, 183, 185, 198, 277], or in image processing, e.g. [6, 108, 116, 118, 171, 172]. In general, RKC methods are used efficiently to solve problems with moderate stiffness or for specific problem classes.

The fundamental concept of the fast explicit methods is the extension of the stability region by using an explicit multi-stage RK scheme and suitable stability polynomials. In order to better understand how these methods work, we first describe the basic idea [244] introduced[1] in the 1960s and then give a general insight into the theoretical framework.

Basic Concept For the sake of simplicity, let us consider the linear ODE system

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{3.1}$$

where $\boldsymbol{u}^0$ is the initial state and the boundary conditions are supposed to be contained in (3.1). Furthermore, let the matrix L be symmetric and negative semi-definite, which is often the case when considering parabolic problems. However, this approach can also be applied to hyperbolic problems, in which L may consist (more or less) negative and imaginary eigenvalues. To describe the main idea we follow [7], which made the relatively unknown so-called STS scheme a popular numerical method. At this point it should be mentioned that the general concept [244] is originated from iterative methods for solving linear systems and is transferred to the parabolic case.

The basic principle is essentially based on the use of varying time step sizes τ_i within a cycle of EE steps

$$\boldsymbol{u}^{k+1} = \left(\prod_{i=1}^{s} (I + \tau_i L) \right) \boldsymbol{u}^k \tag{3.2}$$

over a super time step $\Delta\tau_s = \sum_{i=1}^{s} \tau_i$, in which some of them violate the theoretical stability limit $\tau_{\max}$ originating from EE method. In particular, the STS scheme replaces the sufficient stability requirement

$$\rho\left(I + \tau L\right) \leq 1 \tag{3.3}$$

at the end of a time step τ by using a cycle of s nonuniform time steps $\tau_1, \ldots, \tau_s$, which ensure relaxed stability after each cycle

$$\rho\left(\prod_{i=1}^{s} (I + \tau_i L) \right) \leq 1 \tag{3.4}$$

and for which the numerical solution at $\Delta\tau_s$ approximates the solution of the problem, while inner values are only be considered as intermediate calculations. As a result, methods based on this approach are very simple, explicit and at the same time impressively effective. The

[1] We stress that Yuan' Chzao-Din, Franklin and Guillou & Lago proposed the basic idea at the same time.

challenging task now is to select optimal values for the time step sizes τ_i in such a way that an efficient explicit method is achieved.

The technique of finding an optimal set of time steps that relate to the optimality properties of Chebyshev polynomials can be understood as a direct approach to a computation rule. Later we will also present an indirect way [108] which is based on the relationship between iterated box filtering and Gaussian convolution.

Super Time Stepping Method The STS scheme explicitly determines the time steps τ_i in such a way that the relaxed stability condition (3.4) is ensured, while the *super time step* $\Delta\tau_s$ of such a cycle is maximised. More precisely, the condition (3.4) is fulfilled when

$$\left| \prod_{i=1}^{s} (1 + \tau_i \lambda) \right| \leq 1, \quad \forall \lambda \in [-\lambda_{\max}, -\lambda_{\min}] \tag{3.5}$$

where $|\lambda_{\min}|$ and $|\lambda_{\max}|$ are the smallest and largest eigenvalues of L, respectively. The condition (3.5) can be further restricted by the requirement

$$\left| \prod_{i=1}^{s} (1 + \tau_i \lambda) \right| \leq K, \quad \forall \lambda \in [-\lambda_{\max}, \mu] \tag{3.6}$$

with $\mu \in [-\lambda_{\min}, 0)$, $K \in (0, 1)$. The latter term is called *damping* and ensures numerical stability. On this basis, hyperbolic-parabolic problems can also be realised. The damping technique is described in more detail in the next section.

The condition (3.6) is now the starting point for finding an optimal set of varying τ_i so that the polynomial

$$p_s(\lambda) = \prod_{i=1}^{s} (1 + \tau_i \lambda) \tag{3.7}$$

finally satisfies:

$$\begin{aligned} &|p_s(\lambda)| \leq K, \quad \forall \lambda \in [-\lambda_{\max}, \mu] && \text{(Stability)} \\ &|p_s'(0)| = \sum_{i=1}^{s} \tau_i \quad \text{maximal} && \text{(Optimality)} \end{aligned} \tag{3.8}$$

The optimality is achieved if the properties of the Chebyshev polynomials

$$T_s(x) = \cos\left(s \arccos(x)\right) \tag{3.9}$$

of degree s in the sense of $p_s(\lambda) = T_s(\lambda)$ are exploited. Based on the zeros of these polynomials, the set of optimal values τ_i is explicitly given by

$$\tau_i = \tau_{\max} \left((-1 + \nu) \cos\left(\frac{(2i-1)\pi}{2n} \right) + 1 + \nu \right)^{-1}, \quad i = 1, \ldots, s \tag{3.10}$$

with $\tau_{\max}$ being the theoretical upper bound for a stable EE scheme and $\nu = \frac{\mu}{\lambda_{\max}}$, $0 < \nu < \frac{\lambda_{\min}}{\lambda_{\max}}$ a damping factor. In particular, if ν is increased, the size of each τ_i is decreased, so

more computations are required to reach the same diffusion time. The super time step size with s intermediate time steps (stages) corresponds to

$$\Delta\tau_s = \sum_{i=1}^{s} \tau_i = \tau_{\max} \frac{N}{2\sqrt{\nu}} \left(\frac{(1+\sqrt{\nu})^{2s} - (1-\sqrt{\nu})^{2s}}{(1+\sqrt{\nu})^{2s} + (1-\sqrt{\nu})^{2s}} \right) \tag{3.11}$$

which results in

$$\Delta\tau_s \xrightarrow[\nu \to 0]{} s^2 \tau_{\max} \tag{3.12}$$

Thus, STS is up to s times faster than the EE scheme with only marginal additional computational costs. In fact, the STS scheme can be characterised as a multi-stage RK method in which the intermediate stages are selected for stability rather than higher order accuracy. The global error bound of this method is given by the following theorem:

Theorem 3.1 ([7]). *Let L be a symmetric negative semi-definite matrix and the function $\boldsymbol{u}$ the exact solution of* (3.1). *The time step sizes $\tau_1, \ldots, \tau_s$ are determined according to* (3.8) *with $\Delta\tau_s = \sum_{i=1}^{s} \tau_i$, then the numerical solution after k cycles satisfies*

$$\left\| \boldsymbol{u}(k\Delta\tau_s) - \boldsymbol{u}^k \right\|_2 \leq \frac{k\lambda_{\max}}{2} \sum_{i=1}^{s} \tau_i^2 \left\| \boldsymbol{u}^0 \right\|_2 \tag{3.13}$$

From (3.13) it follows that the corresponding approximation order with respect to $\Delta\tau_s$ is one. In particular, the size of $\Delta\tau_s$, which is explicitly determined by the spectral radius of L and the choice of s as well as ν, is only limited by the accuracy. Thus, the STS scheme can be understood as an unconditionally stable method.

Here, ν is interpreted as a damping parameter, which means that the results are very sensitive with respect to high frequencies. Accordingly, it has been suggested to use larger values for ν which makes the method more robust to higher frequencies. However, this is associated with a reduction in the cycle time $\Delta\tau_s$, so that the value ν implies a compromise between efficiency and damping quality.

Although the STS scheme is unconditionally stable in theory, the numerical stability is not guaranteed due to internal instabilities (numerical rounding errors) and generally makes the method unusable in practice. In order to overcome the numerical instability, a suitable rearrangement [99] of the internal time steps τ_i can reduce the round-off errors.

The concept of STS is essentially based on extending the stability region so that there is a less strict limitation of the time step size. Due to its explicit construction, this class of methods represents a compromise between cheap computation and the avoidance of small time step sizes of classic explicit methods. Based on the STS concept presented, let us describe the derivation of this method class in more detail.

3.2 Explicit Runge-Kutta-Chebyshev Methods

In the following let us explain the theoretical insights of the abovementioned basic methodology. In doing so, we follow the works [1, 120, 138, 281, 286]. A historical survey of the development of explicit RK methods can also be found in [280]. Fundamentally, it can be noted that the central concept of the RKC methods is based on a special construction in two steps.

First, optimal stability polynomials are designed that maximise the stability region along the negative real axis. Second, appropriate numerical methods with such a favourable stability function are finally constructed.

The RKC methods are dedicated to solve ODE systems of the form

$$\dot{\boldsymbol{u}}(t) = \boldsymbol{f}\left(t, \boldsymbol{u}(t)\right), \quad t \in (0, t_F],\, \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{3.14}$$

whereby the boundary conditions are supposed to be contained in (3.14). The derivation and stability analysis of such schemes is generally independent of the class of PDEs or the space discretisation. The only requirements for using RKC methods are: first, the eigenvalue spectrum of the Jacobian matrix $J = \partial \boldsymbol{f}/\partial \boldsymbol{u}$ should lie in a narrow strip along the negative axis of the complex plane, and second, the Jacobian matrix should be close to a normal matrix. These two properties are trivially true if the Jacobian matrix is symmetric and negative semi-definite.

Stability Function As mentioned earlier, the stability function according to the Dahlquist test equation must satisfy two requirements: accuracy and stability. Regarding the consistency, we recall that the stability function $R(z)$ must approximate e^z as $z \to 0$ with some order of accuracy. In particular, $R(z) = e^z + \mathcal{O}(z^p)$ implies the p-th order of consistency. For example, the stability function of a first order accurate time stepping scheme must fulfil $R(z) = 1 + z + \mathcal{O}(z^2)$, in contrast, a second order scheme satisfies $R(z) = 1 + z + \frac{z^2}{2} + \mathcal{O}(z^3)$. Otherwise, the condition $|R(z)| \leq 1$ must be fulfilled to ensure the stability of a numerical method used.

The aim is now to design a stability function $R(z)$ in such a way that the stability region

$$\mathcal{S} = \left\{ z \in \mathbb{C} : \, |R(z)| \leq 1 \right\} \tag{3.15}$$

contains a large (still bounded) interval on the negative real axis $[-\gamma, 0]$ with $\gamma > 0$, the *boundary of absolute stability*, as large as possible. After such a suitable stability function has been found, it is then possible to construct a numerical method.

3.2.1 Optimal Stability Polynomials

The first step in constructing RKC methods is to find a p-th order polynomial of the form

$$R_s(z) = 1 + \sum_{k=1}^{p} \frac{z^k}{k!} + \sum_{k=p+1}^{s} \beta_k \frac{z^k}{k!} \tag{3.16}$$

where s is the number of stages and $p \leq s$, in which γ_s is maximised subject to $|R_s(z)| \leq 1$. Assuming that the order p is fixed, the remaining free coefficients β_k (for $k = p+1, \ldots, s$) of the stability polynomial (3.16) are naturally used to obtain a larger stability boundary γ_s. In this context, the closed-form solutions for the polynomials with maximal real stability boundaries are known as *optimal stability polynomials*.

First Order Polynomials Finding optimal polynomials depends on two terms: on the one hand, the zeros of the stability polynomial $R_s(z)$ are contained in the stability region. It is thus obvious that an appropriate distribution of these real zeros gives a maximal value of γ_s.

On the other hand, the condition of $|R_s(z)| \leq 1$ must be fulfilled in between the zeros. The key component of these two issues is related to the Chebyshev polynomials. It is known that the Chebyshev polynomials $T_s(x)$ of the first kind in the interval $x \in [-1, 1]$ realise such an optimal distribution of the zeros for a given s, while $|T_s(x)| \leq 1$ is satisfied. On this basis, a shift then leads to the optimal stability polynomial $R_s(z) := T_s(1 + \frac{z}{s^2})$ with $\gamma_s = 2s^2$. The latter can be specified in the following theorem:

Theorem 3.2 ([138]). *For any explicit, consistent RK method, the boundary of absolute stability γ_s depends on the number of stages s with $\gamma_s \leq 2s^2$, and the optimal stability polynomial is the shifted Chebyshev polynomial of the first kind*

$$R_s(z) = T_s\left(1 + \frac{z}{s^2}\right) \tag{3.17}$$

In order to verify (3.17), the evaluation of $R_s(z)$ confirms the first order accuracy for any s, since

$$R_s(z) = T_s\left(1 + \frac{z}{s^2}\right) = 1 + z + \mathcal{O}\left(z^2\right) \tag{3.18}$$

In particular, one obtains

$$\begin{aligned} R_2(z) &= 1 + z + \frac{1}{8}z^2 \\ R_3(z) &= 1 + z + \frac{4}{27}z^2 + \frac{4}{729}z^3 \\ R_4(z) &= 1 + z + \frac{5}{32}z^2 + \frac{1}{128}z^3 + \frac{1}{8192}z^4 \end{aligned} \tag{3.19}$$

Furthermore, the stability interval and the boundary of absolute stability γ_s are explicitly obtained by analysing $|R_s(z)| \leq 1$. For the Chebyshev polynomials hold

$$|T_s(x)| \leq 1 \quad \Longrightarrow \quad |x| \leq 1 \tag{3.20}$$

and therefore it follows that

$$|R_s(z)| = \left|T_s\left(1 + \tfrac{z}{s^2}\right)\right| \leq 1 \quad \Longrightarrow \quad \left|1 + \tfrac{z}{s^2}\right| \leq 1 \tag{3.21}$$

In general, z is complex-valued, however, for finding a real interval one can assume that z is real that consequently implies

$$\left|1 + \tfrac{z}{s^2}\right| \leq 1 \quad \Longrightarrow \quad -2s^2 \leq z \leq 0 \tag{3.22}$$

Thus, the interval of absolute stability for first order schemes based on the stability function (3.17) is given by

$$[-\gamma_s, 0] \quad \text{with} \quad \gamma_s = 2s^2 \tag{3.23}$$

For example, the stability intervals of $P_3(z)$ and $P_6(z)$ can be computed as

$$\begin{aligned} [-\gamma_3, 0] &= [-18, 0] \quad \text{for} \quad P_3(z) \\ [-\gamma_6, 0] &= [-72, 0] \quad \text{for} \quad P_6(z) \end{aligned} \tag{3.24}$$

Overall, the main feature is that the stability interval increases quadratically with the number of stages s and is actually scaled as $s^2\tau_{\max}$, which is crucial to achieve large efficiency gains compared to EE scheme. On closer inspection, the RKC methods can be considered as unconditionally stable, since $s^2\tau_{\max}$ is allowed to become arbitrarily large for sufficiently large s while the stability is guaranteed.

Second Order Polynomials The development of higher order RKC schemes with regard to designing the desired accuracy as well as finding intuitive formulas is much more challenging. The existence and uniqueness of optimal stability polynomials with maximal real negative stability interval for arbitrary p and s are theoretically guaranteed (cf. [286]), however, no analytical expression is known for such polynomials of order $p \geq 2$. Nonetheless, numerical approximations of polynomials (also referred to as *nearly optimal polynomials*) can be used to achieve optimal bounds γ_s which depend quadratically on s according to

$$\gamma_s \approx C_p s^2 \quad \text{for} \quad s \to \infty \quad (C_2 \approx 0.82,\, C_3 \approx 0.49,\, C_4 \approx 0.34) \tag{3.25}$$

For $p = 2$ there are two approximate polynomials in analytical form [280] for arbitrary s, which were derived by Bakker and van der Houwen & Sommeijer. In general, the Bakker-Chebyshev polynomial is preferred because of its better robustness with respect to numerical errors. For instance, Bakker's stability function for second order RKC methods is defined by

$$B_s(z) = \frac{2}{3} + \frac{1}{3s^2} + \left(\frac{1}{3} - \frac{1}{3s^2}\right) T_s\left(1 + \frac{3z}{s^2 - 1}\right) \tag{3.26}$$

which represents approximately 80% of the optimal stability region. By way of illustration, second order accuracy is verified by

$$\begin{aligned} B_3(z) &= 1 + z + \frac{1}{2}z^2 + \frac{1}{16}z^3 \\ B_4(z) &= 1 + z + \frac{1}{2}z^2 + \frac{2}{25}z^3 + \frac{1}{250}z^4 \\ B_5(z) &= 1 + z + \frac{1}{2}z^2 + \frac{7}{80}z^3 + \frac{1}{160}z^4 + \frac{1}{6400}z^5 \end{aligned} \tag{3.27}$$

In an analogous manner as above one can examine $|B_s(z)| \leq 1$ which finally leads to $\frac{2}{3}(s^2 - 1) \leq z \leq 0$. Therefore, the interval of absolute stability for (3.26) can be specified as

$$[-\gamma_s, 0] \quad \text{with} \quad \gamma_s \approx \frac{2}{3}\left(s^2 - 1\right) \tag{3.28}$$

We mention that "≈" in (3.28) depends on the degree of s. For an even degree γ_s is equal to $\frac{2}{3}(s^2 - 1)$, whereas for an odd degree γ_s is slightly larger. In contrast to the first order polynomials the stability intervals are reduced, but yield a higher temporal accuracy. For example, the interval of absolute stability for $B_3(z)$ and $B_6(z)$ is given by

$$\begin{aligned} [-\gamma_3, 0] &= [-16/3, 0] \quad \text{for} \quad B_3(z) \\ [-\gamma_6, 0] &= [-70/3, 0] \quad \text{for} \quad B_6(z) \end{aligned} \tag{3.29}$$

Higher Order Polynomials In the past, various strategies have been proposed for approximating optimal stability polynomials which can generally be classified into three main approaches. Besides the RKC methods mentioned, there are the DUMKA methods and the ROCK methods. In particular, the latter two can be used to construct consistency orders of $p > 2$. The DUMKA methods are based on the zeros of the optimal stability polynomials, which can be computed iteratively. Conversely, the ROCK methods use the orthogonality of Chebyshev polynomials and are obtained by combining the approaches of RKC and DUMKA. For more detailed information, see e.g. [1] and the references therein. Apart from that, higher order methods can also be achieved using extrapolation techniques [174, 175].

3.2.2 Construction of Explicit Runge-Kutta-Chebyshev Methods

Given an optimal stability polynomial, the second step is now to construct a corresponding RKC method. Schemes based on the optimal stability polynomials (3.17) and (3.26) are known as first and second order RKC methods, respectively. In particular, two main strategies have been realised for the construction. First, by factorisation of Euler steps, and second, by exploiting the three-term recurrence relation of the Chebyshev polynomials. In the following we describe the RKC methods formulated as factorised or recursive scheme.

Methods by Factorisation The basic idea for such RKC methods goes back to Yuan' Chzao-Din, Saul'ev, Franklin and Guillou & Lago in the years around 1960. The approach is based on a sequence of EE steps $\Psi_{\tau_1}, \Psi_{\tau_2}, \ldots, \Psi_{\tau_s}$ with corresponding time step sizes $\tau_1, \tau_2, \ldots, \tau_s$ which defines a one-step method as the composition

$$\boldsymbol{u}_1 = \left(\Psi_{\tau_s} \circ \cdots \circ \Psi_{\tau_1}\right)\left(\boldsymbol{u}^0\right) \tag{3.30}$$

Applying this factorisation gives the stability function $R_s(z) = \prod_{i=1}^{s}(1 + \tau_i z)$. As is known, the shifted Chebyshev polynomials are optimal so that the optimal sequence $\tau_1, \ldots, \tau_s$ is given by

$$\tau_i = -\frac{1}{z_i}, \quad i = 1, \ldots, s \tag{3.31}$$

where z_i are the zeros of $T_s(z)$, since

$$T_s(z) = \prod_{i=1}^{s}(z - z_i) = \prod_{i=1}^{s}\left(1 - \frac{z}{z_i}\right) \tag{3.32}$$

Thus, the resulting s-stage numerical scheme, called the *factorised* method, reads

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_j &= \boldsymbol{y}_{j-1} + \tau_j \tau_{\max} \boldsymbol{f}\left(t_n + c_{j-1}\tau_{\max}, \boldsymbol{y}_{j-1}\right), \quad 1 \le j \le s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.33}$$

with solution $\boldsymbol{y}_j$ at stage j being denoted the intermediate solution of the method. Here, $\tau_{\max}$ is the upper stability condition for the integration step $\tau_{\max} \le \frac{\gamma_s}{\rho(J)}$ with the spectral radius

ρ of J. Obviously, the factorised method corresponds to an explicit s-stage RK method with

$$c_0 = 0, \quad c_j = \sum_{i=1}^{j} \tau_i \quad (1 \leq j \leq s), \quad c_s = 1 \tag{3.34}$$

so that the stages (3.33) of this method are a sequence of EE steps with time step sizes

$$\widetilde{\tau}_i = \left| \frac{\tau_{\max}}{z_i} \right| \tag{3.35}$$

and the corresponding super time step $\Delta\widetilde{\tau}_s = \widetilde{\tau}_1 + \cdots + \widetilde{\tau}_s$.

As already mentioned, the order of the time step sizes $\widetilde{\tau}_i$ is extremely important with respect to the internal stability. From a theoretical point of view, the factorised method is stable at the end of each super time step due to (3.32). Otherwise, the condition $|(1 + \widetilde{\tau}_i \lambda_k)| > 1$ reveals that half of the stages are unstable (internal instability), e.g. for certain eigenvalues λ_k of J. Furthermore, the first zero of $T_s(z)$, which is much smaller in absolute value than the others, constructs a very large Euler step. The latter two facts lead to practical problems such as numerical rounding errors which can cause a large accumulation of errors within one cycle $\Delta\widetilde{\tau}_s$. A strategy to improve the internal stability and to overcome highly inaccurate results is based on a special order in which small and large time steps are combined, see [99]. However, a proper order of the time step sizes depends on the stages used and can still lead to unstable approximations.

Note that the factorised method (3.33) cannot be used as second order method because the Bakker-Chebyshev polynomials (3.26) possess complex zeros. To achieve second order accuracy, e.g. the Richardson extrapolation is applied as proposed in [198]. Another second order method is possible using the numerical scheme of Lebedev [160] which is constructed in a similar way to the factorised method described above. The Lebedev method approximates the optimal stability polynomials by Zolotarev polynomials, while the internal stability is again achieved using a special order of the stages. In addition, higher order factorised methods [196, 197] have been recently derived, which possess a high internal stability while maintaining efficiency.

Methods by Recurrence Instead of using a factorisation formulation, van der Houwen & Sommeijer [281] developed a family of methods with special recurrence relations around 1980. Based on the *three-term recurrence relation*

$$T_0(z) = 1, \quad T_1(z) = z, \quad T_j(z) = 2zT_{j-1}(z) - T_{j-2}(z), \quad 2 \leq j \leq s \tag{3.36}$$

they constructed a simple first order s-stage RKC scheme that generates the stability function (3.17), which is given by

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \frac{\tau}{s^2} \boldsymbol{f}_0 \\ \boldsymbol{y}_j &= 2\boldsymbol{y}_{j-1} - \boldsymbol{y}_{j-2} + \frac{2\tau}{s^2} \boldsymbol{f}_{j-1}, \quad 2 \leq j \leq s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.37}$$

with $\boldsymbol{f}_j = \boldsymbol{f}(t_n + c_j\tau, \boldsymbol{y}_j)$, the time step size $\tau = t_{n+1} - t_n$ and the intermediate solutions $\boldsymbol{y}_j$. Apart from the computation of $\boldsymbol{y}_1$, the solutions $\boldsymbol{y}_j$ at each intermediate stage obviously depend on the two previous stages. The increment parameters c_j are defined as

$$c_0 = 0, \quad c_1 = \frac{1}{s^2}, \quad c_j = 2c_{j-1} - c_{j-2} + \frac{2}{s^2} \quad (2 \leq j \leq s) \tag{3.38}$$

so that the standard RK form reads

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_j &= \boldsymbol{u}^n + \tau \sum_{l=0}^{j-1} a_{jl} \boldsymbol{f}(t_n + c_l\tau, \boldsymbol{y}_l), \quad 1 \leq j \leq s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.39}$$

where the coefficients a_{jl} are identified via (3.38) and satisfying $c_l = \sum_{l=0}^{j-1} a_{jl}$. It should be noted that $0 = c_0 < c_1 < \cdots < c_{s-1} < c_s = 1$ and thus all (intermediate) stages at points $t_n + c_j\tau$ lie within the current integration step.

The derivation of the stability function for each internal stage can be verified by applying the scheme (3.37) to the Dahlquist test problem $u' = \lambda u$ via

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \frac{\tau}{s^2}\lambda \boldsymbol{y}_0 = \left(1 + \frac{\tau\lambda}{s^2}\right) \boldsymbol{y}_0 = T_1\left(1 + \frac{\tau\lambda}{s^2}\right) \boldsymbol{u}^n \\ \boldsymbol{y}_2 &= 2\boldsymbol{y}_1 - \boldsymbol{y}_0 + \frac{2\tau}{s^2}\lambda \boldsymbol{y}_1 = \left(1 + \frac{4\tau\lambda}{s^2} + \frac{2(\tau\lambda)^2}{s^4}\right) \boldsymbol{y}_0 = T_2\left(1 + \frac{\tau\lambda}{s^2}\right) \boldsymbol{u}^n \\ \boldsymbol{y}_j &= 2\boldsymbol{y}_{j-1} - \boldsymbol{y}_{j-2} + \frac{2\tau}{s^2}\lambda \boldsymbol{y}_{j-1} = T_j\left(1 + \frac{\tau\lambda}{s^2}\right) \boldsymbol{u}^n, \quad 3 \leq j \leq s \end{aligned} \tag{3.40}$$

so that after a super time step $\boldsymbol{u}^{n+1} = \boldsymbol{y}_s = T_s(1 + \frac{z}{s^2})\boldsymbol{u}^n$ with $z = \tau\lambda$ is obtained.

The main advantage of the three-term Chebyshev recursion is based on its internal stability, which is of crucial importance for the practical application. According to [281], it can be shown that the scheme is stable at each integration step. This means that the recursive RKC method, in contrast to the factorised method, only uses stable time integration steps. Besides preserving internal stability, convergence properties were also analysed. The results of the convergence analysis can be found in [287]. We will come back to this issue later.

3.2.3 Damped Stability Function

Although the introduced RKC schemes are of beneficial use, practical problems arise when using the mentioned stability polynomials (3.17) and (3.26). To explain the potential complications, let us exemplarily consider the stability function and the stability region of the first order polynomial $P_3(z)$ in Figure 3.1. On closer inspection, the stability function contains interior points $z \in (-\gamma_s, 0)$ with $|R_s(z)| = 1$, so that at these points the stability function touches the stability constraint at $y = 1$ or $y = -1$. Similarly, the same holds for

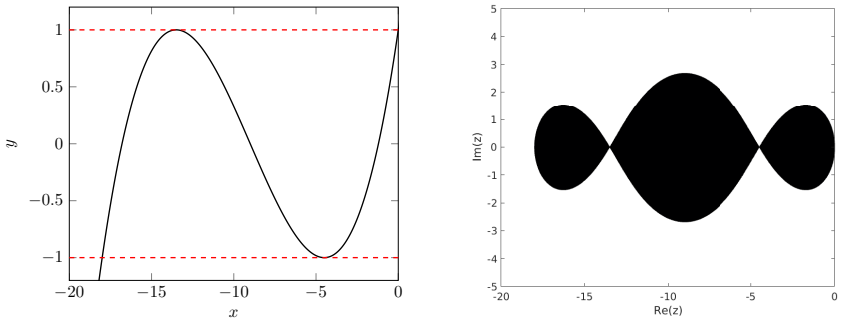


Figure 3.1: First order stability polynomial $R_3(z)$ of degree $s = 3$. **Left:** Stability function. **Right:** Stability region.

the stability region which contracts to a point on the real axis when $|R_s(z)| = 1$. This issue is generally very restrictive in applications where a small imaginary perturbation on z due to $|R_s(z)| > 1$ might cause instability.

To solve this problem, a modification of the stability polynomials by adding an additional small damping is useful. In doing so, the stability requirement $|R_s(z)| \leq 1$, $z \in [-\gamma_s, 0]$ is replaced by $|R_s(z)| \leq \nu < 1$, $z \in [-\gamma_{s,\nu}, -\delta_\nu]$ where δ_ν being a small positive parameter depending on ν. On this basis, the stability function $|R_s(z)|$ becomes bounded slightly below 1 without losing its order of accuracy. Obviously, the proposed technique leads to a slightly smaller stability interval than the undamped version, but the gain in stability is primarily of central importance. In addition, the damping can also be used for problems with complex-valued eigenvalues with a small imaginary part. It should be emphasised that the use of a damping technique is generally preferable, especially for model problems that contain a real eigenvalue spectrum. This follows from the fact that without a suitable damping, the higher frequency components might be preserved such that these frequencies are not damped and may thus lead to oscillations. As a result, an additional damping makes the method more robust with respect to high frequencies.

The application of the damping technique requires the introduction of a damping parameter

$$w_0 = 1 + \frac{\epsilon}{s^2} \tag{3.41}$$

with a small positive value ϵ. To derive the damped stability functions, the following ansatz based on the general structures (3.17) and (3.26) via

$$R_s(z) = a_s + b_s T_s(\omega_0 + \omega_1 z) \tag{3.42}$$

is used, where the parameters a_s, b_s, ω_0, ω_1 are determined in such a way that the requirements for first and second order accuracy are fulfilled.

First Order Damped Polynomials In order to construct first order damped polynomials the consistency conditions must be naturally satisfied. As is known, the stability function $R_s(z) = \beta_0 + \beta_1 z + \beta_2 z^2 + \cdots + \beta_s z^s$ of a first order accurate time stepping scheme must fulfil $R_s(z) = 1 + z + \mathcal{O}(z^2)$. This requirement implies $\beta_0 = 1$ and $\beta_1 = 1$ which corresponds

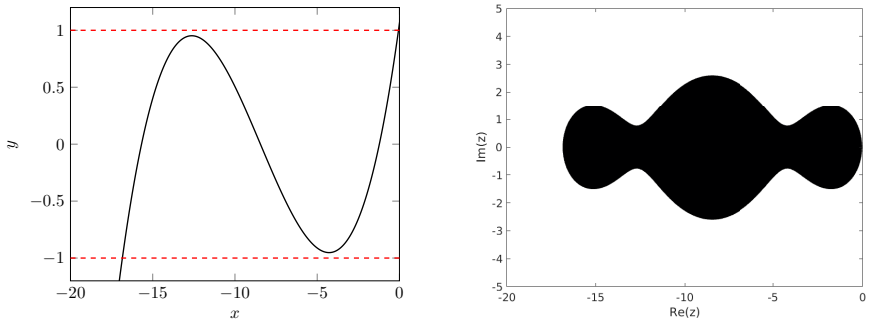


Figure 3.2: First order damped stability polynomial $R_3(z)$ of degree $s = 3$. **Left:** Stability function with damping. **Right:** Stability region with damping.

to the conditions $R_s(0) = 1$ and $R'_s(0) = 1$, respectively. On this basis, the parameters of the damped version are chosen such that

$$R_s(0) = a_s + b_s T_s(\omega_0) = 1, \quad R'_s(0) = \omega_1 b_s T'_s(\omega_0) = 1 \tag{3.43}$$

implies first order consistency. With $a_s = 0$ one obtains the expression for b_s by

$$R_s(0) = b_s T_s(\omega_0) = 1 \quad \Longrightarrow \quad b_s = \frac{1}{T_s(\omega_0)} \tag{3.44}$$

and it follows that

$$R'_s(0) = \omega_1 b_s T'_s(\omega_0) = 1 \quad \Longrightarrow \quad \omega_1 = \frac{1}{b_s T'_s(\omega_0)} \quad \Longrightarrow \quad \omega_1 = \frac{T_s(\omega_0)}{T'_s(\omega_0)} \tag{3.45}$$

Finally, the first order damped stability function can be written in the format

$$R_s(z) = a_s + b_s T_s(\omega_0 + \omega_1 z) = \frac{T_s(\omega_0 + \omega_1 z)}{T_s(\omega_0)} \tag{3.46}$$

Furthermore, it can be shown that

$$\gamma_s \approx \left(2 - \frac{4}{3}\epsilon\right) s^2 \tag{3.47}$$

whereby the boundary of absolute stability is slightly reduced compared to the undamped version (3.23). In practice, $\epsilon = 0.05$ is often used which leads to $\gamma_s \approx 1.93 s^2$. For completeness, the damped stability function and the damped stability region are illustrated in Figure 3.2.

Second Order Damped Polynomials In an analogous manner it is assumed that

$$B_s(z) = a_s + b_s T_s(\omega_0 + \omega_1 z) \tag{3.48}$$

and to ensure second order accuracy, the following must hold:

$$B_s(0) = a_s + b_s T_s(\omega_0) = 1, \quad B'_s(0) = \omega_1 b_s T'_s(\omega_0) = 1, \quad B''_s(0) = \omega_1^2 b_s T''_s(\omega_0) = 1 \tag{3.49}$$

By solving the system depending on the three equations and unknowns, the stability function reads as

$$B_s(z) = 1 + \frac{T_s''(\omega_0)}{\left(T_s'(\omega_0)\right)^2}\left(T_s(\omega_0 + \omega_1 z) - T_s(\omega_0)\right) \tag{3.50}$$

with the parameters

$$a_s = 1 - b_s T_s(\omega_0), \quad b_s = \frac{T_s''(\omega_0)}{\left(T_s'(\omega_0)\right)^2}, \quad \omega_1 = \frac{T_s'(\omega_0)}{T_s''(\omega_0)} \tag{3.51}$$

The boundary of the damped stability interval is then given by

$$\gamma_s \approx \tfrac{2}{3}\left(s^2 - 1\right)\left(1 - \tfrac{2}{15}\epsilon\right) \tag{3.52}$$

whereby the damping coefficient is often suggested as $\epsilon = 2/13$.

3.2.4 Explicit Stabilised Runge-Kutta-Chebyshev Methods

As described earlier, the original RKC methods are based on the three-term recursion. Logically, the Chebyshev recursion has to be adapted when the damped stability polynomials are used. By imposing the three-term Chebyshev recursion and using the property $R_j(0) = 1$ it follows that R_j satisfies

$$\begin{aligned} &R_0(z) = 1, \quad R_1(z) = 1 + \widetilde{\mu}_1 z \\ &R_j(z) = (1 - \mu_j - \nu_j) + \mu_j R_{j-1}(z) + \nu_j R_{j-2}(z) + \widetilde{\mu}_j R_{j-1}(z) z + \widetilde{\gamma}_j z, \quad 2 \le j \le s \end{aligned} \tag{3.53}$$

with associated parameters

$$\widetilde{\mu}_1 = b_1\omega_1, \quad \mu_j = \frac{2b_j\omega_0}{b_{j-1}}, \quad \nu_j = \frac{-b_j}{b_{j-2}}, \quad \widetilde{\mu}_j = \frac{2b_j\omega_1}{b_{j-1}}, \quad \widetilde{\gamma}_j = -a_{j-1}\widetilde{\mu}_j, \quad 2 \le j \le s \tag{3.54}$$

Using the relations (3.53), the RKC integration formula for the underlying model problem (3.14) can be derived by associating R_j with the intermediate approximation $\boldsymbol{y}_j$ and z as the function evaluation which results in

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \widetilde{\mu}_1 \tau \boldsymbol{f}_0 \\ \boldsymbol{y}_j &= (1 - \mu_j - \nu_j)\boldsymbol{y}_0 + \mu_j \boldsymbol{y}_{j-1} + \nu_j \boldsymbol{y}_{j-2} + \widetilde{\mu}_j \tau \boldsymbol{f}_{j-1} + \widetilde{\gamma}_j \tau \boldsymbol{f}_0, \quad 2 \le j \le s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.55}$$

The latter s-stage RKC scheme obviously belongs to the explicit RK class (3.39). It should be noted that a different set of coefficients is defined for each value of s. Finally, the *first order s-stage RKC scheme* is realised on the basis of $\mu_j + \nu_j = 1$ and $a_j = 0$ via

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \widetilde{\mu}_1 \tau \boldsymbol{f}_0 \\ \boldsymbol{y}_j &= \mu_j \boldsymbol{y}_{j-1} + \nu_j \boldsymbol{y}_{j-2} + \widetilde{\mu}_j \tau \boldsymbol{f}_{j-1}, \quad 2 \le j \le s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.56}$$

with the integration parameters

$$\widetilde{\mu}_1 = \frac{\omega_1}{\omega_0}, \quad \mu_j = \frac{2b_j\omega_0}{b_{j-1}}, \quad \nu_j = \frac{-b_j}{b_{j-2}}, \quad \widetilde{\mu}_j = \frac{2b_j\omega_1}{b_{j-1}}, \quad 2 \leq j \leq s \tag{3.57}$$

The numerical stability of the scheme can be guaranteed if

$$\tau \leq \left(1 - \frac{2}{3}\epsilon\right) s^2 \tau_{\max} \tag{3.58}$$

where $\tau_{\max}$ is the theoretical stability limit for the original EE scheme. In contrast, the *second order s-stage RKC scheme* is given by

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \widetilde{\mu}_1 \tau \boldsymbol{f}_0 \\ \boldsymbol{y}_j &= (1 - \mu_j - \nu_j)\boldsymbol{y}_0 + \mu_j \boldsymbol{y}_{j-1} + \nu_j \boldsymbol{y}_{j-2} + \widetilde{\mu}_j \tau \boldsymbol{f}_{j-1} + \widetilde{\gamma}_j \tau \boldsymbol{f}_0, \quad 2 \leq j \leq s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.59}$$

with the parameters

$$\begin{aligned} &\widetilde{\mu}_1 = b_1\omega_1 \\ &\mu_j = \frac{2b_j\omega_0}{b_{j-1}}, \quad \nu_j = \frac{-b_j}{b_{j-2}}, \quad \widetilde{\mu}_j = \frac{2b_j\omega_1}{b_{j-1}}, \quad \widetilde{\gamma}_j = -\left(1 - b_{j-1}T_{j-1}(\omega_0)\right)\widetilde{\mu}_j, \quad 2 \leq j \leq s \end{aligned} \tag{3.60}$$

as well as

$$a_0 = 1 - b_0, \quad a_1 = 1 - b_1\omega_0, \quad b_0 = b_1 = b_2 \tag{3.61}$$

with stability being ensured when

$$\tau \leq \frac{1}{3}\left(s^2 - 1\right)\left(1 - \frac{2}{15}\epsilon\right)\tau_{\max} \tag{3.62}$$

The intermediate solutions $\boldsymbol{y}_j$ depend only on the two previously computed intermediate solutions and $\boldsymbol{y}_0$, independent of the number of stages s. The remaining time increment parameters c_j for the first and second order schemes are defined by

$$c_0 = 0, \quad c_j = \frac{T_s(\omega_0)}{T_s'(\omega_0)}\frac{T_j'(\omega_0)}{T_j(\omega_0)} \quad (1 \leq j \leq s-1), \quad c_s = 1 \tag{3.63}$$

and

$$c_0 = 0, \quad c_1 = \frac{c_2}{T_2'(\omega_0)}, \quad c_j = \frac{T_s'(\omega_0)}{T_s''(\omega_0)}\frac{T_j''(\omega_0)}{T_j'(\omega_0)} \quad (2 \leq j \leq s-1), \quad c_s = 1 \tag{3.64}$$

respectively. Moreover, the increment parameters (3.63) and (3.64) fulfil the condition $0 = c_0 < c_1 < \cdots < c_{s-1} < c_s = 1$. For more details we refer the reader to [281, 286, 287].

3.2.5 Internal Stability and Convergence Properties

Finally, we focus on important properties such as internal stability and convergence properties, see e.g. [287]. Since the RKC methods are in multi-stage form and include a large number of stages s, it is necessary to take into account the error propagation over these stages within a single integration step. This examination is often referred to as the *internal (numerical) stability analysis*. In addition, the internal stability is also important for the convergence of these methods.

As is known, the RKC methods can be specified in the RK form

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_j &= \boldsymbol{u}^n + \tau \sum_{l=0}^{j-1} a_{jl} \boldsymbol{f}(t_n + c_l \tau, \boldsymbol{y}_l), \quad 1 \leq j \leq s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.65}$$

The perturbed version of (3.65) is then represented as

$$\begin{aligned} \widetilde{\boldsymbol{y}}_0 &= \widetilde{\boldsymbol{u}}^n \\ \widetilde{\boldsymbol{y}}_j &= \widetilde{\boldsymbol{u}}^n + \tau \sum_{l=0}^{j-1} a_{jl} \boldsymbol{f}(t_n + c_l \tau, \widetilde{\boldsymbol{y}}_l) + \boldsymbol{r}_j, \quad 1 \leq j \leq s \\ \widetilde{\boldsymbol{u}}^{n+1} &= \widetilde{\boldsymbol{y}}_s \end{aligned} \tag{3.66}$$

where $\widetilde{\boldsymbol{u}}^n$ denotes a perturbation of $\boldsymbol{u}^n$ and $\boldsymbol{r}_j$ a local perturbation at stage j that represents round-off errors. Let $\boldsymbol{e}^n = \widetilde{\boldsymbol{u}}^n - \boldsymbol{u}^n$ and $\boldsymbol{d}_j = \widetilde{\boldsymbol{y}}^n - \boldsymbol{y}^n$ denote the errors associated with this perturbation. By applying the RKC method to the linear system (3.1) the error scheme is in the general form

$$\boldsymbol{d}_j = R_j(\tau L)\boldsymbol{e}^n + \sum_{k=1}^{j} Q_{jk}(\tau L)\boldsymbol{r}_k, \quad 1 \leq j \leq s \tag{3.67}$$

with the absolute stability polynomials R_j and the so-called *internal stability polynomials* Q_{jk}. In particular, the internal stability polynomials determine the propagation of all internal perturbations over the stages within a single integration step. Furthermore, it holds that $\boldsymbol{e}^{n+1} = \boldsymbol{d}_s$ and one obtains in the final stage the error

$$\boldsymbol{e}^{n+1} = R_s(\tau L)\boldsymbol{e}^n + \sum_{k=1}^{s} Q_{sk}(\tau L)\boldsymbol{r}_k \tag{3.68}$$

Let L be symmetric and negative semi-definite, the error for the linear stability results in

$$\left\|\boldsymbol{e}^{n+1}\right\|_2 \leq \max_{z=\tau\lambda} \left|R_s(z)\right| \left\|\boldsymbol{e}^n\right\|_2 + \sum_{k=1}^{s} \max_{z=\tau\lambda} \left|Q_{sk}(z)\right| \left\|\boldsymbol{r}_k\right\|_2 \tag{3.69}$$

where λ denotes the eigenvalues of L. If the absolute stability condition

$$\tau\rho(L) \leq \gamma_s \tag{3.70}$$

is fulfilled, it follows that $\|R_s(z)\|_2 \leq 1$, and then the usual stability from step to step for the propagation $\boldsymbol{e}^n$ is considered. This, however, does not guarantee internal stability in

general and depends directly on the numerical scheme used. An example of this is illustrated in [286] when using the *diagonal method*. Notwithstanding, the use of the recursive RKC method ensures internal stability. Let us emphasise again that the assumption (3.70) should be interpreted as a condition on s rather than a restriction on τ.

Using the numerical scheme (3.55), the internal stability polynomials take the form

$$Q_{sk}(z) = \frac{b_s}{b_k} U_{s-k}(\omega_0 + \omega_1 z), \quad 1 \leq k \leq s \tag{3.71}$$

where $U_i(z)$ being the i-th Chebyshev polynomial of the second kind. If then b_j is chosen as (3.44) or (3.51), (3.61) and $z \in [-\gamma_s, 0]$, it can be shown that

$$\left\| Q_{sk}(z) \right\|_2 \leq \frac{b_s}{b_k}(s-k+1)(1+C\epsilon) \tag{3.72}$$

with the damping parameter ϵ and a constant C of moderate size independent of s. Consequently, for a symmetric and negative semi-definite matrix L with $\tau\rho(L) \leq \gamma_s$ the error bound is determined by

$$\left\| \boldsymbol{e}^{n+1} \right\|_2 \leq \|\boldsymbol{e}^n\|_2 + \sum_{k=1}^{s} \frac{b_s}{b_k}(s-k+1)(1+C\epsilon)\|\boldsymbol{r}_k\|_2 \tag{3.73}$$

and the following theorem holds for the first and second order RKC methods:

Theorem 3.3 ([287]). *Suppose that τ and s are chosen such that the stability time step size restriction $\tau\rho(L) \leq \gamma_s$ is satisfied. Then the following error bound is valid*

$$\left\| \boldsymbol{e}^{n+1} \right\|_2 \leq \|\boldsymbol{e}^n\|_2 + \widetilde{C} \sum_{k=1}^{s} (s-k+1)\|\boldsymbol{r}_k\|_2 \leq \|\boldsymbol{e}^n\|_2 + \tfrac{1}{2} s(s+1)\widetilde{C} \max_k \|\boldsymbol{r}_k\|_2 \tag{3.74}$$

where $\widetilde{C}$ is a constant of moderate size independent of L, τ and s.

Thus, the accumulation of internal perturbations is independent of the spectrum of L if $\tau\rho(L) \leq \gamma_s$ is chosen. Apart from that, the estimate shows that the perturbations grow at most quadratically with s, which is harmless in practice. In contrast to the diagonal or original (not rearranged) factorised method, the recursive RKC scheme is therefore predestined to use a large number of stages. The latter internal stability analysis is given more detailed in [287].

The internal stability is also of practical importance for the convergence properties and the accuracy of the method. Let $\boldsymbol{e}^n = \boldsymbol{u}_h(t_n) - \boldsymbol{u}^n$ be the fully discrete error with respect to the exact PDE solution $\boldsymbol{u}_h(t_n)$ restricted on a grid point with the underlying spatial mesh size h, and $\boldsymbol{\sigma}_h(t)$ is the local spatial truncation error, then the global error bound for the first order undamped RKC method can be specified in the following theorem:

Theorem 3.4 ([287]). *Assume $\boldsymbol{u}_h \in C^2[0, t_F]$ and $\tau\rho(L) \leq \gamma_s$. Then the global errors of the first order undamped RKC scheme satisfy*

$$\|\boldsymbol{e}^n\|_2 \leq C \left(\tau \max_{0 \leq t \leq t_F} \left\| \ddot{\boldsymbol{u}}_h(t) \right\|_2 + \max_{0 \leq t \leq t_F} \left\| \boldsymbol{\sigma}_h(t) \right\|_2 \right), \quad n = 1, 2, \ldots : n\tau \leq t_F \tag{3.75}$$

with a constant C of moderate size independent of L, τ and s.

Consequently, first order temporal convergence is achieved and it can be seen that the global error decreases linearly with τ. For the second order undamped RKC method the following error bound holds:

Theorem 3.5 ([287]). *Assume $\boldsymbol{u}_h \in C^3[0, t_F]$ and $\tau\rho(L) \leq \gamma_s$. Then the global errors of the second order undamped RKC scheme (for $n = 1, 2, \ldots : n\tau \leq t_F$) satisfy*

$$\|\boldsymbol{e}^n\|_2 \leq C \left(\frac{\tau}{s^3} \max_{0\leq t\leq t_F} \|\ddot{\boldsymbol{u}}_h(t)\|_2 + \tau^2 \max_{0\leq t\leq t_F} \|\ddot{\boldsymbol{u}}_h(t)\|_2 + \max_{0\leq t\leq t_F} \|\boldsymbol{\sigma}_h(t)\|_2 \right) \tag{3.76}$$

with a constant C of moderate size independent of L, τ and s.

The error bound (3.76) shows that the second order undamped RKC scheme has almost a second order convergence in time, where the term $\mathcal{O}(\frac{\tau}{s^3})$ corresponds to the fact that the first stage of the scheme possesses only first order consistency. However, a temporal order of two is observed for large s. For further details we refer again to [287]. Note that the convergence properties also hold for the damped RKC schemes and that the results based on internal stability and convergence are transferable to nonlinear parabolic problems.

In conclusion, the class of RKC schemes has several advantages. The methods possess extended stability regions and are easy to implement due to their explicit formulation. Amazingly, the size of a super integration step grows quadratically with the number of stages s. In addition, their explicit nature makes them well-suited for parallel computing such as GPUs. Another advantage of the recursive RKC scheme (3.55) is that maximal three intermediate solutions are required in order to compute the current intermediate solution $\boldsymbol{y}_j$ independent of the number of stages s, which therefore results in low memory storage costs. By exploiting the recursion relations of the Chebyshev polynomials the internal stability within the intermediate stages is ensured which is of crucial importance for practical application. Apart from that, the use of damped stability polynomials makes the method more robust with respect to higher frequencies. A further positive property is that the stabilised explicit RKC methods are simple and well applicable to nonlinear parabolic problems.

Unfortunately, these techniques also have some disadvantages. For example, designing higher order RKC schemes being challenging, but also the development of the formulas and the incorporated parameters may not be intuitive. It should also be noted that the RKC methods are not very suitable for extremely stiff (parabolic) problems as well as for problems with significant advection.

Lastly, we emphasise that the idea of maximising the real stability boundary can also be used to maximise the imaginary stability boundary and thus to develop hyperbolic RKC methods for integrating hyperbolic problems (or convection dominated problems) that have a Jacobian matrix with imaginary eigenvalues. For more details see [142, 254, 271, 279, 280].

3.3 Explicit Stabilised Runge-Kutta-Legendre Methods

According to the latter concept, an alternative approach [182, 183] has been constructed building on the recursion relation associated with the Legendre polynomials which is called the *Runge-Kutta-Legendre (RKL)* method. The proposed shifted Legendre polynomials are

used as the basis for the construction of robust STS schemes without losing important features of this class of methods. The main advantage of using Legendre polynomials over Chebyshev polynomials is that their absolute value is bounded by one if their argument lies in the range $(-1, 1)$. Therefore, the corresponding stability function satisfies $|R_s(z)| < 1$ for each stage of the multi-stage RK scheme, so that the RKL method is naturally stabilised. This makes the method very robust and at the same time no damping is required, but the natural stabilisation leads to a slightly smaller time step size limit. Let us briefly describe the RKL scheme below.

As is known, the Legendre polynomials $P_j(x)$ satisfy $|P_j(x)| < 1$ for $x \in (-1, 1)$. Of course, these polynomials also obey a three-term recursion which is given by

$$P_0(x) = 1, \quad P_1(x) = x, \quad P_j(x) = \left(\tfrac{2j-1}{j}\right) xP_{j-1}(x) - \left(\tfrac{j-1}{j}\right) P_{j-2}(x), \quad 2 \le j \le s \tag{3.77}$$

For the development of first and second order RKL methods the same derivation as for the RKC schemes can be used. As already stated, the general ansatz for the stability polynomial of an s-stage RKL scheme is based on $R_s(z) = a_s + b_s P_s(\omega_0 + \omega_1 z)$. Building on the boundedness of the Legendre polynomials, no damping is needed and the damping parameter is set to $\omega_0 = 1$ for all RKL schemes. When deriving RKL schemes again consistency conditions must be ensured. For first order consistency $R_s(0) = 1$ and $R_s'(0) = 1$ must hold, which consequently results in $a_s = 0$, $b_s = 1$ and $\omega_1 = \frac{2}{s^2+s}$. Thus, the stability function is determined as

$$R_s(z) := P_s\left(1 + \frac{2}{s^2+s} z\right) \tag{3.78}$$

with the corresponding stability polynomial for each s being a shifted Legendre polynomial. Furthermore, the first order stability polynomials P_j satisfy the recursion relation

$$\begin{aligned} P_j\left(1 + \frac{2}{s^2+s} z\right) = &\left(\frac{2j-1}{j}\right) P_{j-1}\left(1 + \frac{2}{s^2+s} z\right) + \left(\frac{1-j}{j}\right) P_{j-2}\left(1 + \frac{2}{s^2+s} z\right) \\ &+ \left(\frac{2j-1}{j}\right) \left(\frac{2}{s^2+s} z\right) P_{j-1}\left(1 + \frac{2}{s^2+s} z\right) \end{aligned} \tag{3.79}$$

Based on the latter recursion the *first order s-stage RKL scheme* is given by

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \widetilde{\mu}_1 \tau \boldsymbol{f}_0 \\ \boldsymbol{y}_j &= \mu_j \boldsymbol{y}_{j-1} + \nu_j \boldsymbol{y}_{j-2} + \widetilde{\mu}_j \tau \boldsymbol{f}_{j-1}, \quad 2 \le j \le s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.80}$$

with the integration parameters

$$\widetilde{\mu}_j = \frac{2j-1}{j}\omega_1 = \frac{2j-1}{j}\frac{2}{s^2+s}, \quad \mu_j = \frac{2j-1}{j}, \quad \nu_j = \frac{1-j}{j} \tag{3.81}$$

To ensure stability, the condition $|R_s(z)| \le 1$ must be fulfilled. This implies that

$$\left|P_s(1 + \omega_1 z)\right| \le 1 \quad \Longrightarrow \quad |1 - \omega_1 \tau \lambda_{\max}| \le 1 \tag{3.82}$$

and $\tau \leq \frac{2}{\omega_1 \lambda_{\max}}$, so that the scheme remains stable if

$$\tau \leq \left(\frac{s^2 + s}{2} \right) \tau_{\max} \tag{3.83}$$

The corresponding time increment parameters c_j are defined by the integration coefficients μ_j, ν_j and $\widetilde{\mu}_j$ via

$$c_0 = 0, \quad c_1 = \widetilde{\mu}_1, \quad c_j = \mu_j c_{j-1} + \nu_j c_{j-2} + \widetilde{\mu}_j \quad (2 \leq j \leq s) \tag{3.84}$$

To achieve a second order accurate RKL scheme, the leading terms of the stability polynomial must exactly match with the expansion of the exponential in the exact solution that is equivalent to satisfy the conditions $R_s(0) = 1$, $R_s'(0) = 1$ and $R_s''(0) = 1$. Without damping and thus setting $\omega_0 = 1$, the coefficients of the stability polynomial are determined by

$$\begin{aligned} a_s &= 1 - b_s, \quad b_s = \frac{P_s''(1)}{\left(P_s'(1)\right)^2} = \frac{s^2 + s - 2}{2s(s+1)}, \quad (2 \leq j \leq s) \\ \omega_1 &= \frac{P_s'(1)}{P_s''(1)} = \frac{4}{s^2 + s - 2} \end{aligned} \tag{3.85}$$

For $s < 2$ the free parameters are set to $b_0 = b_1 = b_2 = \frac{1}{3}$. Substituting this into the three-term Legendre recurrence relation (3.77) yields

$$\begin{aligned} a_j + b_j P_j(1 + \omega_1 z) = &\left(1 - \left(\frac{2j-1}{j} \right) \frac{b_j}{b_{j-1}} - \left(\frac{1-j}{j} \right) \frac{b_j}{b_{j-2}} \right) \\ &+ \left(\frac{2j-1}{j} \right) \frac{b_j}{b_{j-1}} \left(a_{j-1} + b_{j-1} P_{j-1}(1 + \omega_1 z) \right) \\ &+ \left(\frac{1-j}{j} \right) \frac{b_j}{b_{j-2}} \left(a_{j-2} + b_{j-2} P_{j-2}(1 + \omega_1 z) \right) \\ &+ \left(\frac{2j-1}{j} \right) \frac{b_j}{b_{j-1}} \omega_1 z \left(a_{j-1} + b_{j-1} P_{j-1}(1 + \omega_1 z) \right) \\ &- a_{j-1} \left(\frac{2j-1}{j} \right) \frac{b_j}{b_{j-1}} \omega_1 z \end{aligned} \tag{3.86}$$

Finally, the *second order s-stage RKL scheme* can be written as

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \widetilde{\mu}_1 \tau \boldsymbol{f}_0 \\ \boldsymbol{y}_j &= (1 - \mu_j - \nu_j) \boldsymbol{y}_0 + \mu_j \boldsymbol{y}_{j-1} + \nu_j \boldsymbol{y}_{j-2} + \widetilde{\mu}_j \tau \boldsymbol{f}_{j-1} + \widetilde{\gamma}_j \tau \boldsymbol{f}_0, \quad 2 \leq j \leq s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.87}$$

with the integration parameters

$$\begin{aligned}
\widetilde{\mu}_1 &= b_1\omega_1 = \frac{4}{3(s^2+s-2)}, \quad \mu_j = \frac{2j-1}{j}\frac{b_j}{b_{j-1}} = \frac{(2j-1)(j+2)(j-1)^2}{j(j-2)(j+1)^2} \\
\nu_j &= -\frac{j-1}{j}\frac{b_j}{b_{j-2}} = -\frac{(j-1)^3(j^2-4)}{j^3(j+1)(j-3)} \\
\widetilde{\mu}_j &= \mu_j\omega_1 = \frac{(2j-1)(j+2)(j-1)^2}{j(j-2)(j+1)^2}\frac{4}{s^2+s-2}, \quad \widetilde{\gamma}_j = -a_{j-1}\widetilde{\mu}_j = (b_{j-1}-1)\widetilde{\mu}_j
\end{aligned} \tag{3.88}$$

To ensure stability, the corresponding maximum time step size has to be chosen as

$$\tau \leq \left(\frac{s^2+s-2}{4}\right)\tau_{\max} \tag{3.89}$$

Analogous to the first order scheme the remaining time increment parameters c_j are defined by μ_j, ν_j, $\widetilde{\mu}_j$ and $\widetilde{\gamma}_j$ as

$$c_0 = 0, \quad c_1 = \widetilde{\mu}_1, \quad c_j = \mu_j c_{j-1} + \nu_j c_{j-2} + \widetilde{\mu}_j + \widetilde{\gamma}_j \quad (2 \leq j \leq s) \tag{3.90}$$

Overall, the comparison of (3.58) and (3.62) with (3.83) and (3.89) clarifies that the first and second order RKL methods have a slightly smaller time step size limit compared to their RKC counterparts. The substantial advantage, however, is the better stability properties which are useful for practical applications. Theoretical investigations such as the monotone stability and the von Neumann stability analysis are presented in [182,183]. We also emphasise that although the second order RKL method is theoretically stable, in practice only odd values of s should be used as higher frequencies are not damped for even s, see [182]. However, the second order RKC schemes can cause a similarly undesirable property.

The RKC and RKL schemes have been used successfully for parabolic problems in the past. Since both variants are competitive, neither method is specifically preferred. Nevertheless, the RKL methods have recently been considered more often because of their desired stability properties. A comparison of the various STS schemes, especially in connection with nonlinear diffusion, can be found in [2, 3, 34, 183, 277].

In the further course of the work we denote all schemes that belong to the class of multi-stage RK methods with extended stability regions, such as RKC and RKL, as the class of *fast explicit methods*.

3.4 Fast Explicit Diffusion

The introduced fast explicit methods RKC and RKL can be considered as methods in which the extended stability region is constructed in an explicit manner. A similar (but indirect) approach has been developed by Grewenig [107, 108], the so-called *fast explicit diffusion (FED)* method, which is well-known in image processing. In contrast to the class of RKC schemes, the use of FED is based on the decomposition of a box filter that can be factorised into a cycle of explicit linear diffusion steps. In doing so, the FED method uses stable and unstable time step sizes that significantly violate the upper stability bound of an EE scheme.

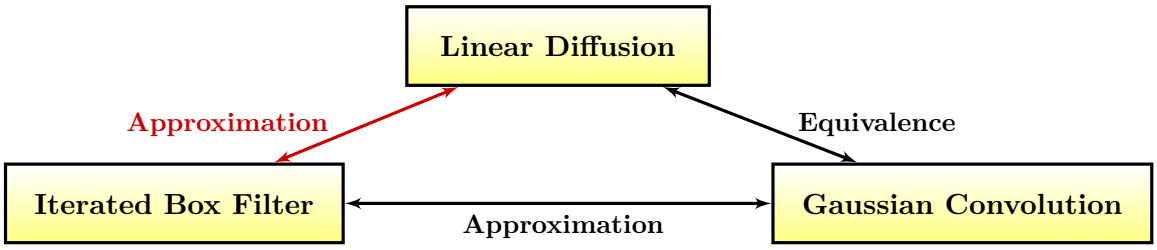


Figure 3.3: Relationship between linear diffusion, Gaussian convolution and iterated box filter. From a theoretical point of view the linear diffusion process is equivalent to the Gaussian convolution. The Gaussian convolution, in turn, can be approximated by iterated box filtering. Thus, the iterative application of the box filter approximated linear diffusion.

In other words, FED is built on a factorisation of Euler steps and is therefore categorised as a factorised method. To overcome some of the drawbacks of the basic FED method, the advanced version called *fast explicit diffusion Runge-Kutta (FEDRK)* can be used, which is based on a recurrence relation of the box filters. As a result, the FEDRK scheme ensures internal stability, so the scheme is of practical use exactly like RKC and RKL. Furthermore, unlike the RKC method, no additional damping is required which makes FEDRK ideal for problems with higher frequencies.

In particular, the FEDRK scheme is of an explicit nature, well-suited for parallel computing and highly efficient, and can also easily be applied to inhomogeneous, isotropic or anisotropic and multi-dimensional diffusion problems. In the following let us provide an insight into the FED and FEDRK method. For a more detailed analysis from a theoretical point of view we refer the reader to [107, 294]. Before going into the details, we first point out the core idea of the FED technique.

Beyond Fast Explicit Diffusion The basic concept of the FED technique can be explained as follows. It is well-known that homogeneous linear diffusion is equivalent to the Gaussian convolution, since the Gaussian kernel is a fundamental solution to this problem. In simple terms, the solution of the homogeneous linear diffusion applied to a signal is equivalent to applying a Gaussian convolution to that signal. On the other hand, it is known from signal filtering theory (central limit theorem) that a box filter that is applied multiple times to a signal approximates a Gaussian convolution with this signal. In addition, it can be shown that a box filter can be factorised into a cycle of explicit linear diffusion steps. Based on this fact, the FED technique provides an alternative way of applying such a filter by using iterations of explicit diffusion steps. The described relationship between linear diffusion, Gaussian convolution and iterated box filter is also illustrated in Figure 3.3.

3.4.1 General Background

We are now going to describe the abovementioned equivalences in more detail. The crucial point within these relationships is based on the equivalence between one-dimensional linear symmetric filters and explicit diffusion schemes with varying time step sizes. For the derivation, a factorisation of linear symmetric filter kernels is used, which can be represented as the sum of discrete derivatives such as finite differences.

Equivalence Between Linear Diffusion and Gaussian Convolution Let us consider the one-dimensional homogeneous linear diffusion equation

$$\partial_t u(x,t) = \operatorname{div}\left(\nabla u(x,t)\right) = \partial_{xx} u(x,t), \quad (x,t) \in \mathbb{R} \times (0,\infty) \tag{3.91}$$

with initial data given by $u(x,0) = f(x)$. The fundamental solution to this process reads

$$u(x,t) = \begin{cases} f(x), & t = 0 \\ \left(G_{\sqrt{2t}} * f\right)(x), & t > 0 \end{cases} \tag{3.92}$$

where G_σ is the Gaussian function defined as

$$G_\sigma(x) := (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left(-\frac{|x|^2}{2\sigma^2}\right) \tag{3.93}$$

with the standard deviation $\sigma > 0$ and the symbol $*$ denotes the continuous convolution

$$(g * h)(x) = \int_{\mathbb{R}} g(x-y)h(y)\,\mathrm{d}y \tag{3.94}$$

In fact, the solution (3.92) means that the linear diffusion applied to $f(x)$ with stopping time t_F is equivalent to a convolution with a Gaussian of variance $\sigma^2 = 2t_F$. For higher dimensional problems the fundamental solution is the product of the fundamental solutions in each variable.

Approximation of Gaussian Convolution by Iterated Filter As stated above, the homogeneous diffusion process is equivalent to a Gaussian convolution applied to the original signal data. For this reason, the Gaussian convolution is of fundamental importance for linear parabolic problems and thus for numerous application. For practical use the continuous convolution can also be expressed in a discrete setting.

Let $\boldsymbol{f} = (f_i)_{i\in\mathbb{Z}}$ and $\boldsymbol{g} = (g_i)_{i\in\mathbb{Z}}$ be discrete real-valued one-dimensional signals given on an equidistant grid with mesh size $h > 0$. The discrete convolution of the two signals is given by

$$(\boldsymbol{f} *_h \boldsymbol{g})_i := \sum_{k\in\mathbb{Z}} f_k g_{i+k} \tag{3.95}$$

However, the latter discrete convolution can also be redefined using a discrete filter L^h_{2n+1} of finite length $(2n+1)h$ with $n \in \mathbb{N}$ via

$$\left(L^h_{2n+1}(\boldsymbol{f})\right)_i := \sum_{k=-n}^{n} w_k f_{i+k} \tag{3.96}$$

where $w_k \in \mathbb{R}$ are the weights of the convolution kernel. For example, in the case of a Gaussian kernel, the discrete kernel $w_k = G_\sigma(x_k)$ is obtained by sampling at the points $x_k = kh$ which are the midpoints of the corresponding intervals $[(k-\frac{1}{2})h, (k+\frac{1}{2})h]$. It should be noted that when diffusion processes are considered the underlying kernels are symmetric and the weights are assumed to be $w_{-k} = w_k$ for $k \geq 1$.

Instead of discretising the Gaussian function and computing the corresponding discrete convolution, one can also use arbitrary filter kernels whose weights are nonnegative and sum up to 1, i.e.

$$\sum_{k=-n}^{n} w_k = 1 \tag{3.97}$$

More precisely, the Gaussian kernel can be approximated using nonnegative filter kernels with weights (3.97). This property follows from the central limit theorem, in which the filter kernels are interpreted as probability density functions. As a result, the Gaussian convolution in the discrete setting can be replaced by iterated filtering. There are several ways to approximate the Gaussian, e.g. binomial kernel, maximum variance kernel, box kernel or extended box kernel, for a more specific definition see [107].

Equivalence Between Symmetric Discrete Filter and Discrete Derivatives Before the relation of symmetric filter kernels and explicit homogeneous diffusion schemes can be illustrated, let us clarify that every discrete linear, symmetric one-dimensional filter L_{2n+1}^h can be written as a weighted sum of discrete even order derivatives such as

$$L_{2n+1}^h := \sum_{m=0}^{n} \alpha_m^{(n)} \Delta_h^m \tag{3.98}$$

with real-valued coefficients $\alpha_m^{(n)}$ and the discrete one-dimensional Laplacian defined as

$$(\Delta_h \boldsymbol{f})_i := \frac{f_{i+1} - 2f_i + f_{i-1}}{h^2} \tag{3.99}$$

For $m = 0$, the operator Δ_h^m is declared as the identity operator, otherwise for $m > 0$ as the m-times composition of the discrete Laplacian which corresponds to a central finite difference approximation for the derivative of order $2m$. The closed-form expression for Δ_h^m with respect to a discrete signal $\boldsymbol{f}$ is specified in the following proposition:

Proposition 3.1 ([107]). *Let $\boldsymbol{f}$ be a discrete signal and $m \geq 1$. Then the m-times composition of the discrete Laplacian Δ_h fulfils*

$$(\Delta_h^m \boldsymbol{f})_i = \frac{1}{h^{2m}} \sum_{k=-m}^{m} (-1)^{m+k} \binom{2m}{m+k} f_{i+k} \tag{3.100}$$

In addition, it can be shown that the coefficients $\alpha_m^{(n)}$ in (3.98) are unique and that there exists an explicit connection between the filter weights w_k and $\alpha_m^{(n)}$ so that the following theorem can be formulated:

Theorem 3.6 ([107]). *Let L_{2n+1}^h be an arbitrary discrete linear, symmetric one-dimensional filter. Then the representation*

$$L_{2n+1}^h = \sum_{m=0}^{n} \alpha_m^{(n)} \Delta_h^m \tag{3.101}$$

expressed by a weighted sum of discrete even order derivative approximations is unique. The

corresponding coefficients are given by

$$\alpha_m^{(n)} = h^{2m} \sum_{k=m}^{n} \left(\binom{k+m}{2m} + \left(1 - \delta_{(k+m),0}\right) \binom{k+m-1}{2m} \right) w_k \tag{3.102}$$

with the Kronecker delta $\delta_{i,j}$.

In this context, let us give an example for the computation of a signal convoluted via the original definition (3.96) and the equivalent representation (3.101) using the box filter B_{2n+1}^h.

Example 3.1. *Let us consider the box filter* B_{2n+1}^h *with length* $(2n+1)h$. *The uniform weights of* B_{2n+1}^h *are defined by*

$$w_k = \frac{1}{2n+1}, \quad \sum_{k=-n}^{n} w_k = 1, \quad k = -n, \ldots, 0, \ldots, n \tag{3.103}$$

In this example, the signal is given by

$$\boldsymbol{f} = (f_i) = (0, 0, 1, 2, 3, 2, 1, 0, 0) \tag{3.104}$$

The convoluted signal (3.96) *based on the box filter with* $h = 1$, $n = 1$ *and* $w_k = \frac{1}{3}$ *yields*

$$\left(B_3^1(\boldsymbol{f})\right)_i = \sum_{k=-1}^{1} w_k f_{i+k} = \left(0, \tfrac{1}{3}, 1, 2, \tfrac{7}{3}, 2, 1, \tfrac{1}{3}, 0\right) \tag{3.105}$$

Analogously, for $n = 2$ *and* $w_k = \frac{1}{5}$ *it follows that*

$$\left(B_5^1(\boldsymbol{f})\right)_i = \sum_{k=-2}^{2} w_k f_{i+k} = \left(\tfrac{1}{5}, \tfrac{3}{5}, \tfrac{6}{5}, \tfrac{8}{5}, \tfrac{9}{5}, \tfrac{8}{5}, \tfrac{6}{5}, \tfrac{3}{5}, \tfrac{1}{5}\right) \tag{3.106}$$

From Theorem 3.6 it is known that the box filter take the form

$$B_{2n+1}^h = \sum_{m=0}^{n} \alpha_m^{(n)} \Delta_h^m \tag{3.107}$$

where the corresponding coefficients are computed via (3.102) *as*

$$\alpha_0^{(n)} = 1, \quad \alpha_m^{(n)} = \frac{h^{2m}}{2m+1} \binom{n+m}{2m} \quad (\textit{for } m > 0) \tag{3.108}$$

Consequently, the convoluted signal computed with (3.100) *and* $\Delta_1^0 = I$ *for* $n = 1$ *leads to*

$$\begin{aligned} \left(B_3^1(\boldsymbol{f})\right)_i &= \sum_{m=0}^{1} \alpha_m^{(1)} \left(\Delta_1^m \boldsymbol{f}\right)_i = \alpha_0^1 (\boldsymbol{f})_i + \alpha_1^1 \left(\Delta_1^1 \boldsymbol{f}\right)_i = (\boldsymbol{f})_i + \tfrac{1}{3} \left(\Delta_1^1 \boldsymbol{f}\right)_i \\ &= f_i + \tfrac{1}{3} \left(f_{i-1} - 2f_i + f_{i+1}\right) = \left(0, \tfrac{1}{3}, 1, 2, \tfrac{7}{3}, 2, 1, \tfrac{1}{3}, 0\right) \end{aligned} \tag{3.109}$$

and for $n = 2$ *to*

$$\begin{aligned}\left(B_5^1(\boldsymbol{f})\right)_i &= \sum_{m=0}^{2} \alpha_m^{(2)} \left(\Delta_1^m \boldsymbol{f}\right)_i \\ &= \alpha_0^2 \left(\boldsymbol{f}\right)_i + \alpha_1^2 \left(\Delta_1^1 \boldsymbol{f}\right)_i + \alpha_2^2 \left(\Delta_1^2 \boldsymbol{f}\right)_i \\ &= \left(\boldsymbol{f}\right)_i + \left(\Delta_1^1 \boldsymbol{f}\right)_i + \tfrac{1}{5} \left(\Delta_1^2 \boldsymbol{f}\right)_i \\ &= f_i + \left(f_{i-1} - 2f_i + f_{i+1}\right) + \tfrac{1}{5} \left(f_{i-2} - 4f_{i-1} + 6f_i - 4f_{i+1} + f_{i+2}\right) \\ &= \left(\tfrac{1}{5}, \tfrac{3}{5}, \tfrac{6}{5}, \tfrac{8}{5}, \tfrac{9}{5}, \tfrac{8}{5}, \tfrac{6}{5}, \tfrac{3}{5}, \tfrac{1}{5}\right) \end{aligned} \tag{3.110}$$

which gives the same results as using the original definition.

Based on the fact that a discrete linear, symmetric filter L_{2n+1}^h can be represented as a series expansion of a weighted sum of discrete even order derivative approximations, the equivalence between symmetric one-dimensional filter kernels and explicit homogeneous diffusion schemes with varying time step sizes can ultimately be derived.

Diffusion Interpretation of Symmetric Filters The starting point for demonstrating the equivalence is based on the abovementioned filter representation (3.101). The discrete one-dimensional Laplace operator Δ_h contained therein can be replaced by the variable $-z$, i.e. $\Delta_h = -z$, which here reflects the spectrum of this operator, so that the following polynomial $P_L^n(z) \in \mathbb{C}$ is obtained:

$$L_{2n+1}^h = \sum_{m=0}^{n} \alpha_m^{(n)} \Delta_h^m = \sum_{m=0}^{n} \alpha_m^{(n)} (-z)^m =: P_L^n(z) \tag{3.111}$$

According to the fundamental theorem of algebra, $P_L^n(z)$ has exactly n zeros $z_0, \ldots, z_{n-1} \in \mathbb{C}$ and can be written as

$$P_L^n(z) = c \prod_{m=0}^{n-1} (z_m - z) \tag{3.112}$$

where $c \in \mathbb{R}$ is a normalisation factor. Assuming that the weights w_k of the filter L_{2n+1}^h are nonnegative, this also holds for the coefficients $\alpha_m^{(n)}$ represented by (3.102). In the case of $P_L^n(0) = \alpha_0^{(n)} > 0$, this implies

$$P_L^n(0) = c \prod_{m=0}^{n-1} z_m = \alpha_0^{(n)} > 0 \tag{3.113}$$

which yields $z_m \neq 0$ for all m and therefore the normalisation factor has to fulfil the following simple representation:

$$c = \alpha_0^{(n)} \left(\prod_{m=0}^{n-1} z_m \right)^{-1} \tag{3.114}$$

If it is further assumed that the weights of L^h_{2n+1} satisfy the symmetry condition $w_{-k} = w_k$ and sum up to 1, it holds that

$$\alpha_0^{(n)} = \sum_{k=0}^{n} \left(\binom{k}{0} + (1 - \delta_{k,0}) \binom{k-1}{0} \right) w_k = w_0 + 2 \sum_{k=1}^{n} w_k = 1 \tag{3.115}$$

Due to $\alpha_0^{(n)} = 1$, the polynomial $P_L^n(z)$ can be rewritten as follows:

$$P_L^n(z) = c \prod_{m=0}^{n-1} (z_m - z) = \left(\prod_{m=0}^{n-1} z_m \right)^{-1} \prod_{m=0}^{n-1} (z_m - z) = \prod_{m=0}^{n-1} \left(1 - \frac{z}{z_m} \right) \tag{3.116}$$

Finally, the replacement of the variable $-z$ via the discrete Laplacian Δ_h leads to a composition of operators

$$L^h_{2n+1} = \prod_{m=0}^{n-1} (I + z_m^{-1} \Delta_h) \tag{3.117}$$

with the zeros z_m of the polynomial $P_L^n(z)$ and the identity operator I, i.e. $I\boldsymbol{f} = \boldsymbol{f}$.

Remark 3.1. *Considering the one-dimensional linear diffusion equation $\partial_t u(x,t) = \partial_{xx} u(x,t)$. The application of the standard spatial and temporal discretisation to the PDE gives*

$$u_i^{k+1} = u_i^k + \tau \frac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{h^2} = (I + \tau \Delta_h) u_i^k \tag{3.118}$$

In this case, the representation (3.117) *can be seen as a composition of n explicit diffusion steps with varying time step sizes z_m^{-1}.*

Obviously, the linear filter representation (3.117) corresponds to a series or a so-called *cycle* of explicit homogeneous diffusion steps. In other words, the filter L^h_{2n+1} can be decomposed into n explicit diffusion steps with varying time step sizes z_m^{-1}, $m = 0, \ldots, n-1$. The associated *cycle time* Θ_n, which is also interpreted as the stopping time of the underlying diffusion process, matches with

$$\Theta_n = \sum_{m=0}^{n-1} z_m^{-1} = \alpha_1^{(n)} \tag{3.119}$$

and is thus determined by

$$\alpha_1^{(n)} = h^2 \sum_{k=1}^{n} \left(\binom{k+1}{2} + \binom{k}{2} \right) w_k = h^2 \sum_{k=1}^{n} k^2 w_k \tag{3.120}$$

We summarise these results established by Grewenig in the following main theorem:

Theorem 3.7 ([107])**.** *Let L^h_{2n+1} be an arbitrary discrete linear, symmetric one-dimensional filter kernel with weights that fulfil*

$$\sum_{k=-n}^{n} w_k = 1 \tag{3.121}$$

Then L^h_{2n+1} is equivalent to a cycle of n explicit one-dimensional linear homogeneous diffusion steps

$$L^h_{2n+1} = \prod_{m=0}^{n-1} (I + \tau_m \Delta_h) \tag{3.122}$$

with the varying time step sizes

$$\tau_m = z_m^{-1} \in \mathbb{C} \tag{3.123}$$

where $z_m \in \mathbb{C}\backslash\{0\}$ are the zeros of the polynomial $P^n_L(z)$. The cycle time Θ_n is given by

$$\Theta_n = h^2 \sum_{k=1}^{n} k^2 w_k \tag{3.124}$$

The latter theorem is the key to connecting symmetric filters and explicit diffusion schemes. In particular, it allows a symmetric filter L^h_{2n+1} to be expressed in the form of n explicit diffusion steps. Before presenting the final FED scheme, let us briefly indicate examples of filter factorisations.

Filter Factorisations So far, it is known that any iterated filter kernel whose coefficients are nonnegative and sum up to 1 approximates the Gaussian convolution. In fact, this is rather general and therefore filter kernels must be analysed for practical use, see [107]. In the work mentioned, four discrete filters have been evaluated by investigating their quality and performance aspects. More precisely, the former relates to the approximation quality of a Gaussian by iterated application of a filter, and the latter is based on the corresponding time step sizes, number of iterations per cycle and the cycle times of the filter used. An overview of the discrete filters investigated and their properties for the use within the FED technique is given in Table 3.1. As can be seen, no filter fits perfectly for both aspects examined. However, on closer inspection at the table indicates that the best compromise between efficiency and approximation quality is achieved by the use of the box filter. In contrast, the iterated binomial kernel provides a very good approximation of the Gaussian, but is rather inefficient due to its small time step sizes. Otherwise, the maximum variance kernel only requires a small number of iterations per cycle at the expense of a very poor approximation quality.

Table 3.1: Comparison of three symmetric discrete filters for the approximation of the Gaussian with nonnegative weights which sum up to 1, adopted from [294]. We mention that the extended box filter is also investigated in [107].

Kernel	**Binomial**	**Maximum Variance**	**Box**
Time step sizes τ_i	$\frac{h^2}{4}$	$\frac{h^2}{2} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n}\right)}$	$\frac{h^2}{2} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n+2}\right)}$
Cycle time Θ_n	$\frac{h^2}{4} n$	$\frac{h^2}{2} n^2$	$\frac{h^2}{6}(n^2+n)$
Performance	Poor: $\mathcal{O}(n)$	Very good: $\mathcal{O}(n^2)$	Good: $\mathcal{O}(n^2)$
Quality	Very good	Poor	Good

Box Filter The discrete box filter, denoted by B^h_{2n+1}, is identified as the best kernel in the FED-setup because the filter provides both reasonable approximation quality and still good performance. On this basis, we present some technical aspects of the box filter.

For the representation of B^h_{2n+1} in the form (3.111) the coefficients are computed based on the uniform box filter weights $w_k = \frac{1}{2n+1}$ by means of (3.102) via

$$\alpha_m^{(n)} = \frac{h^{2m}}{2n+1} \sum_{k=m}^{n} \left(\binom{k+m}{2m} + \binom{k+m-1}{2m} \right) = \frac{h^{2m}}{2n+1} \binom{n+m}{2m} \tag{3.125}$$

Finally, the polynomial $P^n_B(z)$ with respect to the box filter B^h_{2n+1} is given by

$$P^n_B(z) = \sum_{m=0}^{n} \frac{h^{2m}}{2n+1} \binom{n+m}{2m} (-z)^m \tag{3.126}$$

The latter can also be represented using the Chebyshev polynomials $T_{2n+1}(z)$ of the first kind for $z > 0$ as

$$P^n_B(z) = (-1)^n \frac{2}{2n+1} \frac{T_{2n+1}\left(\frac{h\sqrt{z}}{2}\right)}{h\sqrt{z}} \tag{3.127}$$

For a detailed derivation see [107, 294]. The reformulated polynomial (3.127) also exists for $z \to 0$ and is equal to 1. In addition, it holds that $|P^n_B(z)| < 1$ for $z \in (0, \frac{4}{h^2}]$, and therefore no additional damping is required. The corresponding zeros $z_0, \dots, z_{n-1}$ of $P^n_B(z)$ coincide with the n positive zeros of the Chebyshev polynomial $T_{2n+1}(\frac{h\sqrt{z}}{2})$ and are determined by

$$z_m = \frac{4}{h^2} \cos^2\left(\pi \frac{2m+1}{4n+2}\right), \quad m = 0, \dots, n-1 \tag{3.128}$$

so that the time step sizes of the explicit one-dimensional linear homogeneous diffusion scheme result in

$$\tau_m = \frac{1}{z_m} = \frac{h^2}{4} \frac{1}{\cos^2\left(\pi \frac{2m+1}{4n+2}\right)}, \quad m = 0, \dots, n-1 \tag{3.129}$$

The associated cycle time based on n diffusion steps reads

$$\Theta_n = h^2 \sum_{k=1}^{n} k^2 w_k = \frac{h^2}{2n+1} \sum_{k=1}^{n} k^2 = \frac{h^2}{6}(n^2 + n) \tag{3.130}$$

Obviously, the stopping time grows quadratically in n and is $\frac{n^2+n}{3}$ times higher than the stopping time of a stable EE scheme with $\tau_{\max} = \frac{h^2}{2}$. The latter consideration is a special case of Theorem 3.7 and yields the following conclusion:

Theorem 3.8 ([108]). *A discrete one-dimensional box filter B^h_{2n+1} is equivalent to a cycle with n explicit linear diffusion steps:*

$$B^h_{2n+1} = \prod_{i=0}^{n-1} (I + \tau_i \Delta_h) \tag{3.131}$$

with the varying time step sizes

$$\tau_i = \tau_{\max} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n+2}\right)}, \quad i = 0, \ldots, n-1 \tag{3.132}$$

and the corresponding diffusion time of one cycle

$$\Theta_n = \tau_{\max}\left(\frac{n^2+n}{3}\right) \tag{3.133}$$

where $\tau_{\max} = \frac{h^2}{2}$ *is the theoretical upper bound for a stable one-dimensional EE scheme.*

It should be noted that the polynomial (3.127) represents the stability polynomial of the underlying FED scheme, which can also be formulated by a Chebyshev polynomial of the second kind via

$$P_B^n(z) = \frac{1}{2n+1} U_{2n}\left(\sqrt{1 - \frac{h^2}{4}z}\right) \tag{3.134}$$

with $z \in [0, \frac{h^2}{4}]$. The corresponding stability region is then given by the set

$$\mathcal{S}_B^n = \left\{ z \in \mathbb{C} : \, |P_B^n(-z)| \leq 1 \right\} = \left\{ z \in \mathbb{C} : \, \left| U_{2n}\left(\sqrt{1 + \frac{h^2}{4}z}\right) \right| \leq 2n+1 \right\} \tag{3.135}$$

In order to match the well-known consistency property

$$P_B^n(-z) = 1 + z + \sum_{k=2}^{n} \beta_k \frac{z^k}{k!} \tag{3.136}$$

a normalisation by means of the division using the cycle time $\Theta_n = \frac{h^2}{6}(n^2+n)$ is required, so that the stability function is finally obtained as follows:

$$R_B^n(-z) := P_B^n\left(\frac{-z}{\Theta_n}\right) \tag{3.137}$$

As a result, the stability region is redefined by

$$\mathcal{S}_B^n = \left\{ z \in \mathbb{C} : \, |R_B^n(-z)| \leq 1 \right\} = \left\{ z \in \mathbb{C} : \, \left| U_{2n}\left(\sqrt{1 + \frac{3}{2(n^2+n)}z}\right) \right| \leq 2n+1 \right\} \tag{3.138}$$

In particular, first order accuracy holds, since

$$\begin{aligned} R_B^2(-z) &= 1 + z + \frac{1}{5}z^2 \\ R_B^3(-z) &= 1 + z + \frac{1}{4}z^2 + \frac{1}{56}z^3 \\ R_B^4(-z) &= 1 + z + \frac{27}{100}z^2 + \frac{27}{1000}z^3 + \frac{9}{10000}z^4 \end{aligned} \tag{3.139}$$

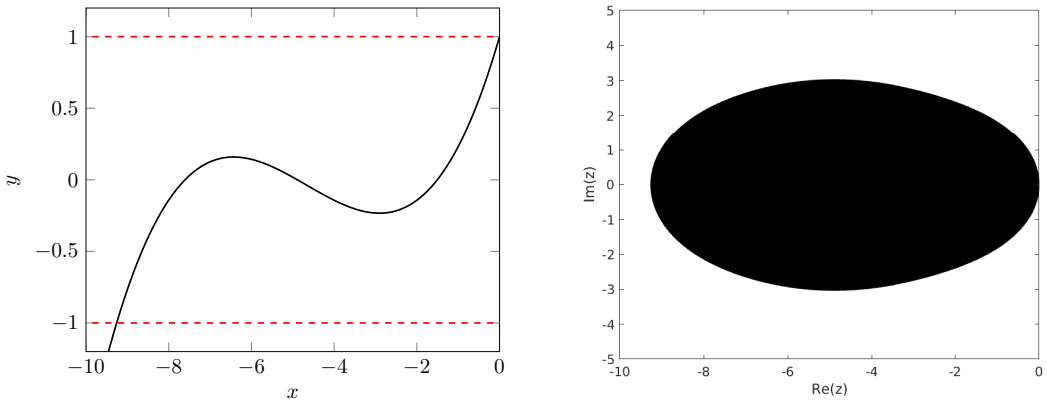


Figure 3.4: Stability polynomial $R_B^3(-z)$ of degree $n = 3$ corresponding to the box filter. **Left:** Stability function. **Right:** Stability region.

For the sake of completeness, the stability function and the stability region of the polynomial $R_B^3(-z)$ are illustrated in Figure 3.4. As already indicated, the stability polynomial based on the box filter ensures $|R_B^n(-z)| < 1$ for $z \in (0, \gamma_n)$ and thus no additional damping is necessary. Obviously, problems such as hyperbolic-parabolic equations with dominant diffusion and possible complex-valued eigenvalues with a small imaginary part can also be dealt with.

In summary, the Gaussian convolution is equivalent to homogeneous diffusion, whereby the latter can easily be solved using the EE scheme. Otherwise, the box filter factorisation is an efficient and accurate ansatz to approximate the Gaussian convolution in the form of explicit diffusion steps with varying time step sizes. Consequently, the EE scheme is replaced with the box filter factorisation to obtain an acceleration technique that is still a simple explicit method, but avoids small time step sizes. The resulting scheme is called the FED method.

3.4.2 Fast Explicit Diffusion Scheme for Homogeneous Diffusion

Let us consider the ODE system, which is obtained by spatial discretisation of the one-dimensional homogeneous heat equation equipped with homogeneous Neumann boundary conditions, in the form of

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad t \in (0, t_F],\ \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{3.140}$$

The underlying matrix L is assumed to be symmetric and negative semi-definite. The numerical approximation of (3.140) with the EE scheme gives

$$\boldsymbol{u}^{k+1} = (I + \tau L)\boldsymbol{u}^k \tag{3.141}$$

According to the well-known Gershgorin's circle theorem [100], the eigenvalues of L lie in the interval $[-\frac{4}{h^2}, 0]$ so that the EE scheme is guaranteed to be stable for

$$\tau \leq \frac{h^2}{2} =: \tau_{\max} \tag{3.142}$$

From the theoretical observations described in this section, the EE scheme can be replaced by a box filter factorisation with varying time step sizes

$$\tau_i = \tau_{\max} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n+2}\right)}, \quad i = 0, \dots, n-1 \tag{3.143}$$

in order to obtain a numerical approximation using an FED cycle:

$$\boldsymbol{u}^{k+1} = \left(\prod_{i=0}^{n-1} (I + \tau_i L) \right) \boldsymbol{u}^k \tag{3.144}$$

The latter cycle in factorised form reads as

$$\boldsymbol{u}^{k,i+1} = (I + \tau_i L)\, \boldsymbol{u}^{k,i}, \quad i = 0, \dots, n-1 \tag{3.145}$$

with $\boldsymbol{u}^{k,0} := \boldsymbol{u}^k$ and setting $\boldsymbol{u}^{k+1} := \boldsymbol{u}^{k,n}$ after a complete cycle. The FED scheme is stable in the Euclidean norm because the vector $\boldsymbol{u}^{k+1}$ satisfies

$$\begin{aligned} \left\| \boldsymbol{u}^{k+1} \right\|_2 &= \left\| \left(\prod_{i=0}^{n-1} (I + \tau_i L) \right) \boldsymbol{u}^k \right\|_2 \leq \left\| \left(\prod_{i=0}^{n-1} (I + \tau_i L) \right) \right\|_2 \left\| \boldsymbol{u}^k \right\|_2 \\ &= \max_{z \in \left[0, \frac{4}{h^2}\right]} \left| \prod_{i=0}^{n-1} (1 - \tau_i z) \right| \left\| \boldsymbol{u}^k \right\|_2 \leq \left\| \boldsymbol{u}^k \right\|_2 \end{aligned} \tag{3.146}$$

where the box filter B_{2n+1}^h is given by the stability polynomial (3.127) that fulfils the condition $|R_B^n(z)| \leq 1$ for all $z \in [0, \frac{4}{h^2}]$. In particular, FED achieves a considerable acceleration compared to the original EE scheme, based on the fact that some of the time steps are significantly large, i.e. $\tau_i \gg \tau_{\max}$. Note that the FED method may violate the stability in the infinity norm, which is presented later in Example 3.2.

3.4.3 Internal Stability

Obviously, the FED scheme (3.144) belongs to the group of factorised methods, which generally do not preserve the internal stability. In this context, let us briefly indicate theoretical and numerical stability aspects.

Theoretical Internal Stability As known from (3.146) the FED scheme is stable in the Euclidean norm after a complete cycle with n time steps. At this point the examination of the theoretical internal stability

$$R_B^{[k,n]}(z) := \prod_{i=0}^{k-1} (1 - \tau_i z), \quad k = 1, \dots, n \tag{3.147}$$

is of particular interest, in which the latter must be bounded in absolute value by 1. Remarkably, it can be shown that this property is fulfilled when the time step sizes τ_i are used in their natural order, sorted from small to large.

Numerical Internal Stability Although the internal stability is theoretically fulfilled, the FED scheme is nevertheless highly sensitive to numerical rounding errors due to $|(1+\tau_i\lambda_k)| \gg 1$ when using large time step sizes. Consequently, a rearrangement of the natural sequence τ_i is necessary in practice. This can be done e.g. by the so-called κ-cycles technique (originally proposed by Gentzsch [99]), Leja ordering or Lebedev-Finogenov ordering. However, the rearrangement causes another problem because the use of a rearranged sequence τ_i may produce unstable intermediate results. More precisely, this leads to a significant loss of efficiency when nonlinear diffusion problems or linear diffusion models with time-dependent boundary conditions and source terms are considered, for which internal model updates should therefore only be performed after a full FED cycle.

Before describing the stabilised FEDRK method that can be used to overcome the internal instability problem, we discuss the consistency and convergence properties of the FED scheme.

3.4.4 Consistency and Convergence

In an analogous manner to the RKC methods, important properties such as the consistency and the convergence of the linear FED scheme are analysed. Although, consistency has been discussed earlier, we return to this property, especially with respect to the FED scheme used with a total of M cycles, but also for the analysis of higher order schemes. The exact solution of (3.140) is given by

$$\boldsymbol{u}(t) = e^{tL}\boldsymbol{u}^0 = \left(\sum_{i=0}^{\infty} \frac{(tL)^i}{i!}\right)\boldsymbol{u}^0 = \left(I + tL + \frac{t^2L^2}{2} + \dots\right)\boldsymbol{u}^0 \tag{3.148}$$

In order to investigate the consistency order of the FED scheme, it must be analysed how consistent the computed approximation for $t_k \to 0$ is compared with the latter expansion. Let t_k be an equidistant time grid

$$\left\{ t_k = k\tfrac{t_F}{M} \,\middle|\, 0 \le k \le M \right\} \subset [0, t_F] \tag{3.149}$$

with $M \geq 1$. The parameter M denotes the number of outer cycles and should not be confused with the number n of inner steps. More precisely, we emphasise that the accuracy of the FED method can only be improved when outer cycles are employed. Obviously, increasing the number of cycles M, whereby n becomes smaller, improves the approximation quality of this scheme which is shown in this subsection. To be consistent, the FED method should approximate the matrix exponential

$$\exp\left(\frac{t_F}{M}L\right) = \sum_{i=0}^{\infty} \frac{(t_F)^i}{M^i i!}L^i \tag{3.150}$$

in some degree. In particular, for first order consistency the condition

$$\exp\left(\frac{t_F}{M}L\right) \approx I + \frac{t_F}{M}L \tag{3.151}$$

must be fulfilled, while for second order the condition reads

$$\exp\left(\frac{t_F}{M}L\right) \approx I + \frac{t_F}{M}L + \frac{(t_F)^2}{2M^2}L^2 \tag{3.152}$$

Based on the polynomial representation (3.126), one FED cycle (3.144) can be written as a matrix polynomial in L as follows:

$$\boldsymbol{u}^{k+1} = \left(\sum_{m=0}^{n} \frac{h^{2m}}{2m+1} \binom{n+m}{2m} L^m \right) \boldsymbol{u}^k \tag{3.153}$$

Consequently, the series expansion of one FED cycle with n inner time steps yields

$$I + \frac{h^2}{3} \binom{n+1}{2} L + \sum_{m=2}^{n} \frac{h^{2m}}{2m+1} \binom{n+m}{2m} L^m \tag{3.154}$$

so that the condition

$$\frac{h^2}{3} \binom{n+1}{2} = \frac{t_F}{M} \tag{3.155}$$

implies first order consistency. Although this condition cannot be fulfilled for arbitrary $t_F > 0$ an incorporated adjustment factor $q \in (0, 1]$ can overcome this problem. This means, first the smallest integer $\widetilde{n}$ that satisfies

$$\Theta_{\widetilde{n}} = \frac{h^2}{3} \binom{\widetilde{n}+1}{2} \geq \frac{t_F}{M} \tag{3.156}$$

is determined by

$$\widetilde{n} = \left\lceil \sqrt{\frac{2}{h^2} \frac{3t_F}{M} + \frac{1}{4}} - \frac{1}{2} \right\rceil \tag{3.157}$$

then the adjustment factor can be defined as

$$q := \frac{t_F}{M \Theta_{\widetilde{n}}} \leq 1 \tag{3.158}$$

As a result, the time step sizes τ_i are scaled via $\widetilde{\tau}_i := \tau_i q, i = 0, \ldots, \widetilde{n} - 1$, so that this finally leads to

$$\prod_{i=0}^{\widetilde{n}-1} (I + \widetilde{\tau}_i L) = I + \frac{t_F}{M} L + \sum_{m=2}^{\widetilde{n}} \frac{(qh^2)^m}{2m+1} \binom{\widetilde{n}+m}{2m} L^m \tag{3.159}$$

which demonstrates that the FED scheme is of first order consistency. Obviously, the introduced scaling does not restrict the stability property, since

$$\left[-q \frac{4}{h^2}, 0 \right] \subseteq \left[-\frac{4}{h^2}, 0 \right] \tag{3.160}$$

due to $q \leq 1$. It should be stressed that the FED scheme, however, cannot satisfy the second order consistency. To show convergence and to derive a closed-form expression of the error bound, we refer to [107]. The error bound estimation of the FED method is given in the following theorem:

Theorem 3.9 ([107]). *Let the function $\boldsymbol{u} : (0, \infty) \to \mathbb{R}^N$ be the exact solution of* (3.140) *with initial data $\boldsymbol{u}^0 = \boldsymbol{u}(0)$, $t_F > 0$ and $M \in \mathbb{N}$ the number of FED cycles. If the cycle*

length $\widetilde{n} \in \mathbb{N}$ *and* $q \in (0,1]$ *are given according to* (3.157) *and* (3.158)*, respectively, then the numerical solution after* M *FED cycles,*

$$\boldsymbol{u}^M := \left(\prod_{i=0}^{\widetilde{n}-1} (I + q\tau_i L) \right)^M \boldsymbol{u}^0 \tag{3.161}$$

fulfils

$$\left\| \boldsymbol{u}(t_F) - \boldsymbol{u}^M \right\|_2 \leq \frac{32}{h^4} \frac{t_F^2}{M} \left\| \boldsymbol{u}^0 \right\|_2 \tag{3.162}$$

From the theorem it is observable that the FED scheme converges to the exact solution if the number of cycles is increased, since

$$\lim_{M \to \infty} \left\| \boldsymbol{u}(t_F) - \boldsymbol{u}^M \right\|_2 = 0 \tag{3.163}$$

In addition, the global error decreases linearly with M and thus the convergence order is one with respect to the number of cycles.

3.4.5 Higher Order Fast Explicit Diffusion

As mentioned above, the FED method cannot guarantee second order accuracy. However, it can be shown that this problem is solved if FED is combined with the well-known Richardson extrapolation technique introduced in [228, 229]. In doing so, the basic idea is to consider the underlying method used over different time increments and to combine them in a suitable manner. We first briefly describe the extrapolation procedure based on the EE method.

The EE scheme $\boldsymbol{u}^{k+1} = (I + \tau L)\boldsymbol{u}^k$ implies first order accuracy in time. In contrast to this calculation method, the numerical solution at time t_{k+1} can alternatively be computed over two different time increments

$$\boldsymbol{u}_1^{k+1} = (I + \tau L)\boldsymbol{u}^k \tag{3.164}$$

$$\boldsymbol{u}_2^{k+1} = \left(I + \tfrac{\tau}{2} L\right)\left(I + \tfrac{\tau}{2} L\right)\boldsymbol{u}^k \tag{3.165}$$

which are finally combined via

$$\boldsymbol{u}^{k+1} = 2\boldsymbol{u}_2^{k+1} - \boldsymbol{u}_1^{k+1} \tag{3.166}$$

The latter *extrapolated* result has second order accuracy which can be shown by comparing the Taylor series of the low order numerical solutions. The corresponding series of $(I + \tau L)$ in (3.164) and $(I + \frac{\tau}{2}L)^2$ in (3.165) result in

$$\boldsymbol{u}_1(t+\tau) = \left(I + \tau L + \tau^2 L^2\right)\boldsymbol{u}(t) + \mathcal{O}\left(\tau^3\right) \tag{3.167}$$

$$\boldsymbol{u}_2(t+\tau) = \left(I + \tau L + \tfrac{3}{4}\tau^2 L^2\right)\boldsymbol{u}(t) + \mathcal{O}\left(\tau^3\right) \tag{3.168}$$

otherwise the series to the analytical solution is given by

$$\boldsymbol{u}(t+\tau) = \left(I + \tau L + \tfrac{1}{2}\tau^2 L^2\right)\boldsymbol{u}(t) + \mathcal{O}\left(\tau^3\right) \tag{3.169}$$

Obviously, (3.167) and (3.168) do not have the consistency order two, but using the ansatz $2 \cdot (3.168) - (3.167)$, like the scheme (3.166), a second order accurate method is obtained. Further details concerning higher order schemes are given, e.g. in [105].

On the basis of this extrapolation concept, a second order FED method can finally be constructed. More precisely, the scaled time step sizes $\widetilde{\tau}_i := \tau_i q$ with the cycle length $\widetilde{n}$ are chosen such that

$$\sum_{i=0}^{\widetilde{n}-1} \widetilde{\tau}_i = \tau \tag{3.170}$$

where τ is the cycle time. In an analogous manner a cycle with the stopping time

$$\sum_{i=0}^{\widetilde{n}-1} \frac{\widetilde{\tau}_i}{2} = \frac{\tau}{2} \tag{3.171}$$

is defined. Consequently, the extrapolated second order FED method reads

$$\begin{aligned} \boldsymbol{u}_1^{k+1} &= \left(\prod_{i=0}^{\widetilde{n}-1} (I + \widetilde{\tau}_i L) \right) \boldsymbol{u}^k, \quad \boldsymbol{u}_2^{k+1} = \left(\prod_{i=0}^{\widetilde{n}-1} \left(I + \tfrac{\widetilde{\tau}_i}{2} L \right) \right)^2 \boldsymbol{u}^k \\ \boldsymbol{u}^{k+1} &= 2\boldsymbol{u}_2^{k+1} - \boldsymbol{u}_1^{k+1} \end{aligned} \tag{3.172}$$

The corresponding stability analysis in the Euclidean norm is shown in [107]. Overall, the FED extrapolation scheme requires a higher computational effort than the second order versions of RKC and RKL, since the approximation is the combination of two separately computed results.

3.4.6 Extension to Arbitrary Diffusion Problems

The FED method introduced in the one-dimensional setting can easily be applied to nonlinear, anisotropic and multi-dimensional problems. Assume that a fully discrete nonlinear diffusion problem with symmetric and negative semi-definite matrix $L(\boldsymbol{u})$, which depends on the time-dependent data $\boldsymbol{u}(t)$, is given in explicit form

$$\boldsymbol{u}^{k+1} = \left(I + \tau L(\boldsymbol{u}^k) \right) \boldsymbol{u}^k \tag{3.173}$$

In this case, a worst case a priori estimate with respect to the time step size restriction is required obtained by

$$\tau \leq \frac{2}{\rho\left(L\left(\boldsymbol{u}\left(t \right) \right) \right)} =: \tau_{\max} \tag{3.174}$$

Let us denote the largest eigenvalue in magnitude with $\lambda_{\max}$. As is known, the stability of the FED scheme in the Euclidean norm is only satisfied for $\lambda_{\max} \leq \frac{4}{h^2}$. In order to ensure stability also for the case $\lambda_{\max} > \frac{4}{h^2}$, the multiplication by a suitable factor $c < 1$ is employed similar to the adjustment factor q used for the consistency property. The factor is defined as

$$c := \frac{4}{h^2 \lambda_{\max}} \tag{3.175}$$

so that $\lambda_{\max} c = \frac{4}{h^2}$, and the time step sizes are determined via

$$\widehat{\tau}_i := c\tau_i = \frac{2}{\lambda_{\max}} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n+2}\right)} = \tau_{\max} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n+2}\right)}, \quad i = 0, 1, \ldots, n-1 \tag{3.176}$$

Thus, an FED cycle for the nonlinear problem reads then

$$\begin{aligned} \boldsymbol{u}^{k,0} &= \boldsymbol{u}^k \\ \boldsymbol{u}^{k,i+1} &= \left(I + \widehat{\tau}_i L(\boldsymbol{u}^k)\right) \boldsymbol{u}^{k,i}, \quad i = 0, \ldots, n-1 \\ \boldsymbol{u}^{k+1} &= \boldsymbol{u}^{k,n} \end{aligned} \tag{3.177}$$

We stress once again that the matrix $L(\boldsymbol{u}^k)$ should be kept constant within a whole cycle, as the FED scheme does not preserve internal stability from a numerical point of view. Fortunately, this problem can be solved by using a recurrence relation for the box filters, as already explained in connection with RKC and RKL.

3.4.7 Fast Explicit Diffusion Runge-Kutta

The FED method described is essentially based on the decomposition of a box filter and causes numerical rounding errors in practice. To improve the numerical internal stability, a different representation for the box filters building on a recursion formula can be used. The resulting scheme, which is referred to as the FEDRK method, is thus related to the class of RKC schemes introduced.

Box Filter Recursion In an analogous manner to the known three-term recurrence relation with regard to special functions such as Chebyshev or Legendre polynomials, there exists also the derivation of a box filter recursion relation. Let $n \geq 2$, a given one-dimensional filtered signal $\boldsymbol{f}$ using a box filter of length $(2n-1)h$, i.e.

$$\left(B^h_{2n-1}(\boldsymbol{f})\right)_i := \frac{1}{2n-1} \sum_{k=-n+1}^{n-1} f_{i+k} \tag{3.178}$$

which is additionally convolved with the kernel mask $(1/2, 0, 1/2)$ yields

$$\begin{aligned} \widetilde{f}_i &= \frac{1}{2n-1} \sum_{k=-n+1}^{n-1} \frac{1}{2} \left(f_{i+k+1} + f_{i+k-1}\right) = \frac{1}{4n-2} \left(\sum_{k=-n+1}^{n-1} f_{i+k+1} + \sum_{k=-n+1}^{n-1} f_{i+k-1} \right) \\ &= \frac{1}{4n-2} \left(\sum_{k=-n+2}^{n} f_{i+k} + \sum_{k=-n}^{n-2} f_{i+k} \right) =: \left(R^h_{2n+1}(\boldsymbol{f})\right)_i \end{aligned} \tag{3.179}$$

The latter can be expressed as

$$\begin{aligned} \left(R^h_{2n+1}(\boldsymbol{f})\right)_i &= \frac{1}{4n-2} \sum_{k=-n}^{n} f_{i+k} + \frac{1}{4n-2} \sum_{k=-n+2}^{n-2} f_{i+k} \\ &= \frac{2n+1}{4n-2} \left(B^h_{2n+1}(\boldsymbol{f})\right)_i + \frac{2n-3}{4n-2} \left(B^h_{2n-3}(\boldsymbol{f})\right)_i \end{aligned} \tag{3.180}$$

and subsequently rearranged into

$$B^h_{2n+1} = \frac{4n-2}{2n+1} R^h_{2n+1} - \frac{2n-3}{4n-2} B^h_{2n-3} \tag{3.181}$$

The convolution with $(1/2, 0, 1/2)$ corresponds to a one-dimensional linear diffusion step with time step size $\frac{h^2}{2}$, so that one finally obtains for $n \geq 2$:

$$B^h_{2n+1} = \alpha_n \left(I + \tfrac{h^2}{2}\Delta_h\right) B^h_{2n-1} + (1-\alpha_n) B^h_{2n-3} \tag{3.182}$$

with $\alpha_n = \frac{4n-2}{2n+1}$. Consequently, the recurrence formula (3.182) means that a box filter of length $(2n+1)h$ can be decomposed into a sum of two box filters with lengths $(2n-1)h$ and $(2n-3)h$. At this point it should be noted that the recursion relation can also be derived directly using the stability polynomial in the form of the Chebyshev polynomials of the second kind (3.134) with $z \in [0, \frac{4}{h^2}]$, so that the recursion formula reads as

$$\begin{aligned} P^n_B(z) &= \frac{4n-2}{2n+1}\left(1 - \tfrac{h^2}{2}z\right) P^{n-1}_B(z) - \frac{2n-3}{2n+1} P^{n-2}_B(z) \\ &= \alpha_n P^{n-1}_B(z) - \alpha_n \tfrac{h^2}{2} z P^{n-1}_B(z) + (1-\alpha_n) P^{n-2}_B(z) \end{aligned} \tag{3.183}$$

As explained earlier, by replacing z via $-z$ and using an additional normalisation to an FED cycle with length n by means of $\Theta_s = \frac{h^2}{6}(n^2+n)$, the recursion relation is finally given by

$$R^n_B(-z) = \alpha_n R^{n-1}_B(-z) + (1-\alpha_n) R^{n-2}_B(-z) + \alpha_n \tfrac{3}{n^2+n} z R^{n-1}_B(-z) \tag{3.184}$$

Recursive Fast Explicit Diffusion Based on the derived recursion relation (3.184), the original FED scheme of length s with varying time step sizes

$$\boldsymbol{u}^{n+1} = \left(\prod_{i=0}^{s-1} (I + \tau_i L)\right) \boldsymbol{u}^n \tag{3.185}$$

can be replaced by the equivalent formulation of a *first order s-stage FEDRK scheme*

$$\begin{aligned} \boldsymbol{y}_0 &= \boldsymbol{u}^n \\ \boldsymbol{y}_1 &= \boldsymbol{y}_0 + \widetilde{\mu}_1 \tau \boldsymbol{f}_0 \\ \boldsymbol{y}_j &= \mu_j \boldsymbol{y}_{j-1} + \nu_j \boldsymbol{y}_{j-2} + \widetilde{\mu}_j \tau \boldsymbol{f}_{j-1}, \quad 2 \leq j \leq s \\ \boldsymbol{u}^{n+1} &= \boldsymbol{y}_s \end{aligned} \tag{3.186}$$

with $\boldsymbol{f}_j = \boldsymbol{f}(t_n + c_j\tau, \boldsymbol{y}_j)$, the time step size $\tau = t_{n+1} - t_n$, the intermediate solutions $\boldsymbol{y}_j$ and the integration parameters

$$\widetilde{\mu}_j = \alpha_j \frac{3}{s^2+s}, \quad \mu_j = \alpha_j, \quad \nu_j = 1 - \alpha_j \tag{3.187}$$

The associated increment parameters c_j are defined as

$$c_0 = 0, \quad c_1 = \alpha_1 \frac{3}{s^2+s}, \quad c_j = \alpha_i c_{i-1} + (1-\alpha_i) c_{i-2} + \alpha_i \frac{3}{s^2+s} \quad (2 \leq j \leq s) \tag{3.188}$$

which satisfy the condition $0 = c_0 < c_1 < \cdots < c_{s-1} < c_s = 1$. Compared to (3.185) the FEDRK method requires the same number of explicit diffusion steps and is also well-suited for parallel computing. However, two intermediate results for $j \geq 2$ must be stored in each iteration. Conversely, the recursive scheme has two main advantages: first, no rearrangement of time step sizes is required, and second, the internal stability is preserved. The internal stability preservation follows from the relation to the class of RK schemes.

Theorem 3.10 ([107]). *If the time step size $\tau > 0$ satisfies*

$$\tau \leq \tau_{\max} \left(\frac{s^2 + s}{3} \right) \tag{3.189}$$

then each integration step of the FEDRK scheme (3.186) *is stable in the Euclidean norm.*

As a result of preserving the internal stability, it is now allowed to perform inner updates, e.g. when dealing with nonlinear problems. Nevertheless, increasing the number of stages s does not result in highly accurate approximations. In particular, increasing the number of outer cycles M, so that s becomes smaller, generally improves the accuracy of the scheme. Of course, FEDRK can also be used to build a second order method based on the Richardson extrapolation technique.

We note that the FEDRK method is also called the *fast semi-iterative* scheme [118], in which the direct use of (3.182) gives the k-th outer iteration by

$$\begin{aligned} \boldsymbol{u}^{k,i+1} &= \alpha_i \left(I + \widetilde{\tau} L \left(\boldsymbol{u}^{k,i} \right) \right) \boldsymbol{u}^{k,i} + (1 - \alpha_i)\, \boldsymbol{u}^{k,i-1}, \quad i = 0, \ldots, s-1 \\ \widetilde{\tau} &:= \frac{3}{s^2 + s} \tau, \quad \alpha_i = \frac{4i + 2}{2i + 3}, \quad \boldsymbol{u}^{k,-1} := \boldsymbol{u}^{k,0} \end{aligned} \tag{3.190}$$

and $\boldsymbol{u}^{k,0} = \boldsymbol{u}^k$ as well as $\boldsymbol{u}^{k+1} := \boldsymbol{u}^{k,s}$.

After this comprehensive description of the RKC, RKL and FEDRK methods, a summary of the algorithms and a performance evaluation are given in the next two subsections.

3.5 Implementation of Fast Explicit Methods

As we have already clarified in the description of FED, only an increase in the number of outer cycles M improves the accuracy of the numerical solution. This characteristic is naturally also true for the RKC and RKL schemes. Thus, for a given stopping time t_F and a desired number M of cycles, the corresponding number of stages s can be computed from their maximum stable time step size with $\tau = \frac{t_F}{M}$.

For the *first order RKC scheme* the corresponding number of stages s required for the stability (3.58) can be computed via

$$s = \left\lceil \sqrt{\frac{1}{\left(1 - \frac{2}{3}\epsilon\right) \tau_{\max}} \frac{t_F}{M}} \right\rceil \tag{3.191}$$

and for the *second order RKC scheme* using (3.62) as

$$s = \left\lceil \sqrt{1 + \frac{3}{\left(1 - \frac{2}{15}\epsilon\right)\tau_{\max}} \frac{t_F}{M}} \right\rceil \tag{3.192}$$

Here, the ceiling function $\lceil x \rceil$ denotes the smallest integer $n \in \mathbb{Z}$ with $n \geq x$. In an analogous manner, the corresponding number of stages s is obtained for the *first order RKL scheme* by (3.83) with

$$s = \left\lceil \frac{1}{2} \left(\sqrt{1 + \frac{8}{\tau_{\max}} \frac{t_F}{M}} - 1 \right) \right\rceil \tag{3.193}$$

as well as for the *second order RKL scheme* by means of (3.89) as

$$s = \left\lceil \frac{1}{2} \left(\sqrt{9 + \frac{16}{\tau_{\max}} \frac{t_F}{M}} - 1 \right) \right\rceil \tag{3.194}$$

Finally, for the FEDRK method the corresponding number of stages s is computed from its stability condition (3.189) using

$$s = \left\lceil \frac{1}{2} \left(\sqrt{1 + \frac{12}{\tau_{\max}} \frac{t_F}{M}} - 1 \right) \right\rceil \tag{3.195}$$

In total, a comparison of the stability limits or the number of stages s shows that the computational efficiency of the RKC method is marginally better than that of the RKL method. In this regard, FEDRK achieves the most inefficient performance. Nevertheless, both RKL and FEDRK have better stability properties and are therefore much more attractive in practice. The corresponding algorithms for RKC, RKL and FEDRK are illustrated in the Figures 3.5-3.7. Let us emphasise again that the calculation of an estimate for $\tau_{\max}$ poses no problem in practice due to the existence of the Gershgorin's circle theorem [100]. This theorem provides a straightforward rule for estimating the largest eigenvalue $\lambda_{\max}$ in magnitude, and thus for estimating $\tau_{\max}$.

If the underlying model problem is linear, we prefer to apply the recursive form (3.190) since from a computational point of view it is cheaper to compute

$$\begin{aligned} \boldsymbol{u}^{k,i+1} &= \alpha_i \left(I + \widetilde{\tau} L\right) \boldsymbol{u}^{k,i} + \left(1 - \alpha_i\right) \boldsymbol{u}^{k,i-1} \\ &= \alpha_i A \boldsymbol{u}^{k,i} + \left(1 - \alpha_i\right) \boldsymbol{u}^{k,i-1}, \quad i = 0, \ldots, s-1 \end{aligned} \tag{3.196}$$

with $A = I + \widetilde{\tau} L$. On this basis, the first order s-stage RKC scheme (3.56) has the form (3.196) if the parameters are used via

$$\widetilde{\tau} := \widetilde{\mu}_1 \tau = \frac{\omega_1}{\omega_0} \tau \quad \text{and} \quad \alpha_i = \mu_i = \frac{2 b_i \omega_0}{b_{i-1}} \tag{3.197}$$

Analogously, for the first order s-stage RKL scheme (3.80) the parameters are given by

$$\widetilde{\tau} := \frac{2}{s^2 + s} \tau \quad \text{and} \quad \alpha_i = \mu_i = \frac{2i - 1}{i} \tag{3.198}$$

Algorithm 3.1 Explicit stabilised Runge-Kutta-Chebyshev method

Input: Stopping time t_F; number M of RKC cycles; step size limit $\tau_{\max}$; initial state $\boldsymbol{u}^0$

Output: $\boldsymbol{u}(t_F)$

1.) Compute minimum number of stages s according to the formula (3.191) or (3.192)

2.) Compute integration parameters (3.57) and (3.63) or (3.60) and (3.64)

3.) Apply numerical scheme using (3.56) or (3.59)

Figure 3.5: Algorithm for computing first and second order s-stage RKC method.

Algorithm 3.2 Explicit stabilised Runge-Kutta-Legendre method

Input: Stopping time t_F; number M of RKL cycles; step size limit $\tau_{\max}$; initial state $\boldsymbol{u}^0$

Output: $\boldsymbol{u}(t_F)$

1.) Compute minimum number of stages s according to the formula (3.193) or (3.194)

2.) Compute integration parameters (3.81) and (3.84) or (3.88) and (3.90)

3.) Apply numerical scheme using (3.80) or (3.87)

Figure 3.6: Algorithm for computing first and second order s-stage RKL method.

Algorithm 3.3 Fast explicit diffusion Runge-Kutta method

Input: Stopping time t_F; number M of FEDRK cycles; step size limit $\tau_{\max}$; initial state $\boldsymbol{u}^0$

Output: $\boldsymbol{u}(t_F)$

1.) Compute minimum number of stages s according to the formula (3.195)

2.) Compute integration parameters (3.187) and (3.188)

3.) Apply numerical scheme using (3.186) or in combination with (3.172)

Figure 3.7: Algorithm for computing first and second order s-stage FEDRK method. The second order scheme is constructed by an extrapolation method.

3.6 Numerical Experiments on Fast Explicit Methods

To complete this chapter, we finally give a numerical comparison of RKC, RKL and FEDRK using two experiments. More precisely, we investigate the approximation accuracy and the corresponding computational costs based on linear and nonlinear PDE-based image diffusion. First, however, the L_∞-stability is examined.

3.6.1 L_∞-Stability

As is known, all fast explicit methods are stable in the Euclidean norm. The latter property is particularly important for linear and nonlinear diffusion filters (equipped with homogeneous Neumann boundary conditions), since the average grey level of the original image is preserved during image evolution. In contrast, stability properties related to the infinity norm $\|\boldsymbol{u}\|_\infty = \max_i |u_i|$, denoted by L_∞, are important to satisfy the discrete maximum-minimum principle

$$\min_i u_i^k \leq u_j^{k+1} \leq \max_i u_i^k \tag{3.199}$$

for each j, where $\boldsymbol{u}^k$ and $\boldsymbol{u}^{k+1}$ are the numerical solutions at time t_k and t_{k+1}, respectively. Using a specific example from Grewenig [107], we show that all fast explicit methods may violate the L_∞-stability of the form $\|\boldsymbol{u}(t_F)\|_\infty \leq 1$.

Example 3.2. *Let us consider the one-dimensional example based on the symmetric and negative semi-definite matrix $L \in \mathbb{R}^{6\times 6}$ given by*

$$L = \begin{pmatrix} -0.8564 & 0.8564 & 0 & 0 & 0 & 0 \\ 0.8564 & -1.8357 & 0.9793 & 0 & 0 & 0 \\ 0 & 0.9793 & -1.1773 & 0.1980 & 0 & 0 \\ 0 & 0 & 0.1980 & -0.3196 & 0.1216 & 0 \\ 0 & 0 & 0 & 0.1216 & -0.9721 & 0.8506 \\ 0 & 0 & 0 & 0 & 0.8506 & -0.8506 \end{pmatrix} \tag{3.200}$$

Furthermore, we set $t_F = 28/3$, $M = 1$, $\tau_{\max} = \frac{1}{2}$ and consider the initial condition $\boldsymbol{u}^0 = (1,1,1,-1,1,1)^\top$ which fulfils $\|\boldsymbol{u}^0\|_\infty = 1$. Solving this problem with the EE scheme leads to

$$\|\boldsymbol{u}(t_F)\|_\infty = \left\|(0.6857, 0.6780, 0.6673, 0.6130, 0.6756, 0.6811)^\top\right\|_\infty \approx 0.6857 \leq 1 \tag{3.201}$$

so that the L_∞-stability is ensured. In contrast, applying the first order fast explicit schemes produces the following results:

(i) RKC with $\epsilon = 0.05$ and minimum number of stages $s = 5$

$$\|\boldsymbol{u}(t_F)\|_\infty = \left\|(0.222, 0.412, 0.611, 1.937, 0.558, 0.259)^\top\right\|_\infty \approx 1.937 > 1 \tag{3.202}$$

(ii) RKL with minimum number of stages $s = 6$

$$\|\boldsymbol{u}(t_F)\|_\infty = \left\|(0.363, 0.542, 0.734, 1.286, 0.563, 0.511)^\top\right\|_\infty \approx 1.286 > 1 \tag{3.203}$$

(iii) FEDRK with minimum number of stages $s = 7$

$$\left\|\boldsymbol{u}(t_F)\right\|_\infty = \left\|(0.509, 0.594, 0.698, 1.006, 0.604, 0.587)^\top\right\|_\infty \approx 1.006 > 1 \tag{3.204}$$

Consequently, this example demonstrates that all methods may violate the L_∞-stability. In other words, these methods may not maintain positivity.

The latter experiment highlights a disadvantage of the fast explicit methods, as they can cause solution artefacts when using larger time steps for problems with higher frequency structures. As a consequence, high frequencies are not damped and may lead to oscillations. In this case, we emphasise that a time step size which is chosen to be really smaller than the stability limit often yields stability in the L_∞ sense. Otherwise, this problem can also be solved by increasing the number of outer cycles M, since the methods posses natural damping, so that overshoots and undershoots are usually smoothed out in the evolution process. In addition, the experiment also indicates that the FEDRK method appears to be the most robust method.

Let us now compare the methods in terms of the computational effort and the accuracy of the computed solution. For the comparison we use the *mean squared error (MSE)* defined by

$$\mathrm{MSE}(\boldsymbol{u}_1, \boldsymbol{u}_2) := \frac{1}{N}\sum_{i=1}^{N}\left(u_{1,i} - u_{2,i}\right)^2 \tag{3.205}$$

where $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are two images with N pixels.

3.6.2 Linear Image Diffusion

First, we consider the two-dimensional linear diffusion filtering

$$\partial_t u(\boldsymbol{x}, t) = \mathrm{div}(\nabla u(\boldsymbol{x}, t)) = \Delta u(\boldsymbol{x}, t), \quad \boldsymbol{x} \in \mathbb{R}^2,\, t \in (0, t_F] \tag{3.206}$$

with the Laplace operator Δ and where $u(x, y, t) : \mathbb{R}^2 \times (0, t_F] \to \mathbb{R}$ is a continuous image at time t. The two-dimensional initial data (original image) is given by $f(x, y)$ and we set $u(x, y, 0) = f(x, y)$. In image processing f is a digital image, with the pixel (x_i, y_j) representing a grid point on the rectangular grid and $f(x_i, y_j) = f_{i,j}$ the corresponding grey value at that point in the image domain. The application of the continuous linear diffusion filter (3.206) iteratively computes a filtered version $u(x, y, t)$ of $f(x, y)$ at diffusion time t with suitable boundary conditions. In computer science this is also referred to as the evolution process of an image. In general, homogeneous Neumann boundary conditions $\partial_n u = 0$ are chosen so that there is no flux of grey values across the boundaries and thus the average grey value of the image can be preserved (depending on the numerical method used). For a more detailed insight, see e.g. [293] and the references therein.

For the sake of simplicity, we suppose that the grid size is set to $\Delta x = \Delta y = 1$ as is common practice in image processing. The approximation of the spatial partial derivatives u_{xx} and u_{yy} in (3.206) by central differences leads to

$$\frac{\mathrm{d}u_{i,j}}{\mathrm{d}t} = u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} \tag{3.207}$$

and we end up with the semi-discrete system

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad t \in (0, t_F],\, \boldsymbol{u}(0) = \boldsymbol{f} \tag{3.208}$$

Here, $\boldsymbol{u}(t)$ is an N-dimensional vector that represents the unknown values in the image domain, and the Laplacian matrix $L \in \mathbb{R}^{N\times N}$ is symmetric and negative semi-definite. Subsequently, the fully discrete system according to the EE method is given by $\boldsymbol{u}^{k+1} = (I + \tau L)\boldsymbol{u}^k$.

For the linear diffusion experiment the *Lena* test image is selected as initial data, visualised in Figure 3.8. Furthermore, we set the stopping time to $t_F = 100$ seconds and apply the mentioned first and second order fast explicit methods to obtain the filtered image. In order to compare the filtered results $\boldsymbol{u}$ generated by RKC, RKL and FEDRK, the computed solution $\widetilde{\boldsymbol{u}}$ of the EE method (cf. Figure 3.8) is used as a reference solution for the MSE measurement, i.e. $\mathrm{MSE}(\widetilde{\boldsymbol{u}}, \boldsymbol{u})$. The corresponding performance of the methods used is shown in Figure 3.9. It can be seen that in order to improve the accuracy of the fast explicit methods, the number of cycles M must be increased which is directly connected with higher computational costs. The first order versions of FEDRK and RKL provide the best performance. In contrast, the RKC scheme provides worse results, but the second order damped scheme outperforms its first order damped counterpart. As expected, the second order FEDRK scheme achieves the worst performance, because the extrapolation procedure is computationally intensive.

3.6.3 Nonlinear Isotropic Image Diffusion

Lastly, we investigate the performance of the fast explicit methods using nonlinear diffusion filtering introduced by Perona and Malik [217]. In order to avoid smoothing of edges (which are important characteristics of an image) in the diffusion process, they suggested to apply a diffusivity controlled using the image gradient ∇u of the evolving image. The corresponding nonlinear diffusion equation is given by

$$\partial_t u(\boldsymbol{x}, t) = \operatorname{div}\Big(g\Big(|\nabla u(\boldsymbol{x}, t)|^2\Big)\nabla u(\boldsymbol{x}, t)\Big), \quad \boldsymbol{x} \in \mathbb{R}^2,\, t \in (0, t_F],\, u(\boldsymbol{x}, 0) = f(\boldsymbol{x}) \tag{3.209}$$

with a diffusivity function such as

$$g\Big(|\nabla u|^2\Big) := \frac{1}{1 + \frac{|\nabla u|^2}{\kappa^2}}, \quad \kappa > 0 \tag{3.210}$$

Figure 3.8: Linear diffusion filtering. **Left:** *Lena* test image (512×512). **Right:** Filtered image using the EE scheme with stopping time $t_F = 100$ and $\tau_{\max} = 1/4$.

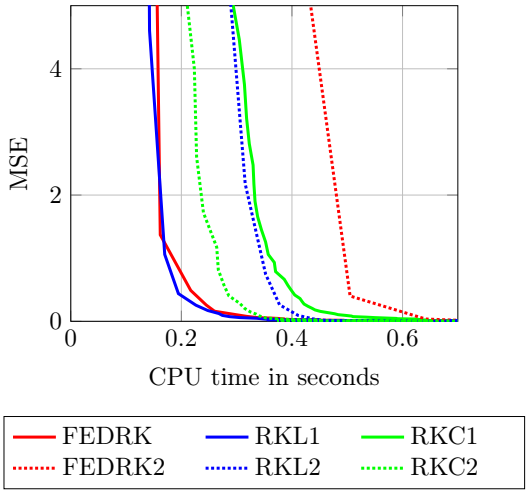


Figure 3.9: Linear diffusion filtering based on the *Lena* test image (512×512), cf. Figure 3.8: comparison of the MSE and the corresponding CPU time for the fast explicit methods at stopping time $t_F = 100$ and $\tau_{\max} = 1/4$. The reference solution is computed with the EE method. The best performances achieve the first order versions of FEDRK and RKL. In contrast, the second order damped RKC scheme (with $\epsilon = 2/13$) outperforms its first order damped counterpart (with $\epsilon = 0.05$). The second order FEDRK scheme clearly provides the worst results because it is based on a computationally intensive extrapolation approach.

Here, κ is a contrast parameter, where the Perona-Malik model is of forward parabolic type for $|\nabla u|^2 \leq \kappa$ and of backward parabolic type for $|\nabla u|^2 > \kappa$, for further details see e.g. [293]. To overcome the ill-posedness, a regularisation in the form $g(|\nabla u_\sigma|^2)$ can be used proposed by [56] which is a Gaussian-smoothed version of (3.210). The resulting regularised nonlinear diffusion process take the form

$$\partial_t u = \operatorname{div}\left(g\left(|\nabla u_\sigma|^2\right)\nabla u\right) \tag{3.211}$$

which is a well-posedness problem for $\sigma > 0$. The implementation of a spatial discretisation as proposed in [293] onto (3.211) yields the semi-discrete nonlinear system

$$\dot{\boldsymbol{u}}(t) = L(\boldsymbol{u}(t))\boldsymbol{u}(t), \quad t \in (0, t_F],\ \boldsymbol{u}(0) = \boldsymbol{f} \tag{3.212}$$

with symmetric and negative semi-definite matrix $L(\boldsymbol{u}(t))$, which depends on the evolving image at time t. Consequently, using the EE scheme, the fully discrete version reads $\boldsymbol{u}^{k+1} = (I + \tau L(\boldsymbol{u}^k))\boldsymbol{u}^k$, which is numerically stable for $\tau \leq \frac{1}{4}$.

Finally, we compare the performance of all the methods at hand a noisy version of the *Lena* test image, see Figure 3.10. For this experiment the parameters are set to $t_F = 100$, $\kappa = 3/2$ and $\sigma = 1/2$. Moreover, the computed solution of the EE method (cf. Figure 3.10) is used again as the reference solution for the MSE assessment. The performances of RKC, RKL and FEDRK based on first and second order consistency are illustrated in Figure 3.11 and show the same results as before. Once again, the first order versions of FEDRK and RKL are clearly the superior methods.

Figure 3.10: Nonlinear isotropic diffusion filtering. **Left:** Noisy *Lena* test image (512×512). **Right:** Filtered image using the EE scheme with stopping time $t_F = 100$ and parameters $\kappa = 3/2$, $\sigma = 1/2$, $\tau_{\max} = \frac{1}{4}$.

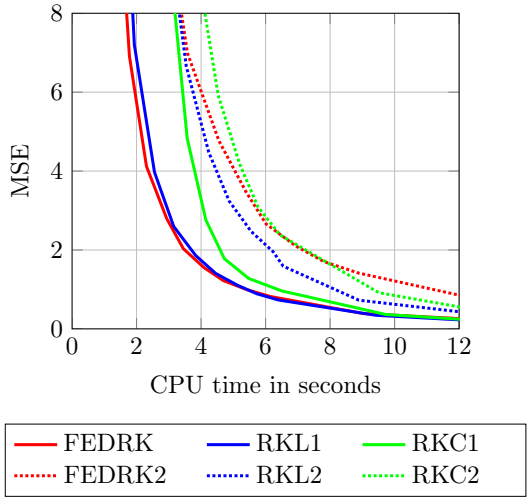


Figure 3.11: Nonlinear isotropic diffusion filtering based on a noisy version of the *Lena* test image (512×512), cf. Figure 3.10: comparison of the MSE and the corresponding CPU time for the fast explicit methods at stopping time $t_F = 100$ and parameters $\kappa = 3/2$, $\sigma = 1/2$, $\tau_{\max} = \frac{1}{4}$. The reference solution is computed with the EE method. Once again, the first order versions of FEDRK and RKL are the superior methods.

Overall, the first order variants of FEDRK and RKL achieve the best results in terms of accuracy and computational costs, where both methods providing nearly competitive efficiency. In contrast, the first order RKC scheme is significantly less efficient. As shown, the second order versions are not beneficial for the experiments considered.

3.7 Summary

Fast explicit methods have favorable stability properties, which makes them ideal for solving stiff parabolic PDEs efficiently. The formulation of fast explicit methods is generally based on factorisation, recurrence relation and a combination of both methods for developing

higher order schemes. Although the stability polynomial is independent of the numerical scheme used, it can no longer be guaranteed that the intermediate stages of the factorised methods are stable, since some τ_i are larger than $\tau_{\max}$. Rearranging the time step sizes τ_i can be helpful, but when linear problems with time-dependent model updates and nonlinear parabolic operators are considered, it can lead to an unstable scheme. In contrast, the recursive methods ensure internal stability and provide a greater robustness in the scheme.

We emphasise again that the stability and convergence results for the fast explicit methods presented here only hold for symmetric matrices in the Euclidean norm. In cases where the operators are nonlinear and nonsymmetric, the condition on the amplification factor $R(z)$ obviously remains the same, but z can now range over the complex plane instead of being restricted to the negative real axis. To preserve stability, the overall amplitude of the amplification factor should be bounded by unity. This is the reason that the fast explicit methods are based on (natural) damped stability polynomials. As a result, the solvers remain well applicable to linear and nonlinear problems with nonsymmetric matrices. However, the following requirements must be satisfied: first, the spectrum of the system matrix (or Jacobian matrix) is located in a long narrow strip along the negative axis of the complex plane, and second, the underlying matrix is close to be normal. This situation typically arises when parabolic or hyperbolic-parabolic PDEs with dominant diffusion are considered.

Besides the comprehensive theoretical overview, the numerical experiments presented in this section show, to the best of our knowledge, for the first time a performance comparison of all three fast explicit methods in the literature. Even if the solvers, especially FEDRK and RKL, are equally competitive, the conclusion of these experiments should not be underestimated. As stated at the beginning of this chapter, the numerical solvers are generally only known in their special research areas such as computational astrophysics or image processing. These knowledge gaps can be filled within the scope of the present work. In particular, we have analysed and discussed in detail the differences in their theoretical derivations, but also their differences from a numerical point of view in relation to L_∞-stability as well as in their use for linear and nonlinear diffusion problems, for example in image processing applications. Although both FEDRK and RKL are considered as superior and competitive schemes, we find that there is a notable difference between the two methods. Apart from their theoretical derivations, it should be emphasised that RKL possess a second order method in which the approximations can be computed efficiently. In contrast, the FEDRK method appears to be more robust with respect to higher frequencies, as shown in Example 3.2, which is advantageous in practice.

Although fast explicit methods are a powerful tool for solving (dominant) parabolic heat or diffusion equations in a highly efficient manner, we emphasise that these methods lose their superiority for highly stiff problems, since in this case the finest grid width leads to an extremely small upper stability bound.

Chapter 4

Model Order Reduction for Linear Dynamical Systems

As already mentioned in this thesis, various physical processes such as the linear heat equation can be described by a linear time-invariant input-output system that arise by spatial discretisation of the underlying mathematical PDE, in a semi-discrete form

$$\begin{aligned} \dot{\boldsymbol{u}}(t) &= L\boldsymbol{u}(t) + K\boldsymbol{w}(t), \quad \boldsymbol{u}(0) = \boldsymbol{u}^0 \\ \boldsymbol{y}(t) &= C\boldsymbol{u}(t) \end{aligned} \tag{4.1}$$

with sparse state matrix $L \in \mathbb{R}^{n\times n}$, input matrix $K \in \mathbb{R}^{n\times p}$ and output matrix $C \in \mathbb{R}^{q\times n}$. Here $\boldsymbol{u}(t) \in \mathbb{R}^n$ denotes the state vector, $\boldsymbol{w}(t) \in \mathbb{R}^p$ the inputs, $\boldsymbol{y}(t) \in \mathbb{R}^q$ the outputs and $\boldsymbol{u}^0$ the initial state of the model. The time-dependent vector $\boldsymbol{w}(t)$ represents e.g. the boundary conditions and the sources/sinks that control the dynamical model. In systems theory, the system (4.1) is called as *multi-input-multi-output (MIMO)* system. Otherwise, for a *single-input-single-output (SISO)* system with $p = q = 1$, the matrices K and C become vectors $\boldsymbol{k}$ and $\boldsymbol{c}^\top$ and the vectors $\boldsymbol{w}$ and $\boldsymbol{y}$ become scalars w and y.

When using time integration methods, the introduced explicit and implicit methods have to handle a large sparse system matrix L, whereby the computational costs are directly linked to the number of grid points that result from the spatial discretisation. In this context, *model order reduction (MOR)* methods (also known as *reduced order model* methods) can be used to approximate the original high dimensional linear and time-invariant first order ODE system (4.1), where the time invariance refers to the fact that the corresponding matrix is constant in time, by a very low r-dimensional dynamical system

$$\begin{aligned} \dot{\boldsymbol{u}}_r(t) &= L_r\boldsymbol{u}_r(t) + K_r\boldsymbol{w}(t), \quad \boldsymbol{u}_r(0) = \boldsymbol{u}^{r,0} \\ \boldsymbol{y}_r(t) &= C_r\boldsymbol{u}_r(t) \end{aligned} \tag{4.2}$$

with $r \ll n$ and $L_r \in \mathbb{R}^{r\times r}, K_r \in \mathbb{R}^{r\times p}, C_r \in \mathbb{R}^{q\times r}, \boldsymbol{u}_r(t) \in \mathbb{R}^r, \boldsymbol{y}_r(t) \in \mathbb{R}^q$ so that the main characteristics of the original ODE system are preserved. In fact, this means that MOR is a technique that is employed as a preprocessing step to reduce the computational effort for numerical simulations of models like (4.1). The general procedure of MOR is visualised in Figure 4.1 and can be considered as a useful tool to obtain efficient numerical simulations while ensuring the desired accuracy. It should be noted that within the formulation (4.2) only the number of the state variables is reduced, whereas the number of inputs and outputs remains the same.

In this thesis, we focus on MOR methods that are based on projections. Existing projection-based MOR techniques can be classified into *singular value decomposition (SVD)-based*

methods and *Krylov-based methods* and are very frequently used in different research fields such as control system theory, circuit design, fluid mechanics, thermal flow problems or computational biology. The mentioned classification is not always consistent in the literature, for a general overview see e.g. [8, 10, 11, 25, 29, 35, 193, 248].

The projection-based MOR methods are derived via a projection of the original model onto a low-dimensional subspace which reduce the full system using projection matrices and therefore rely on efficient numerical linear algebra techniques. Let us mention that projection techniques are characterised by the way in which the projection matrices are constructed. Common and popular MOR approaches are modal coordinate reduction [45, 76, 93, 222], balanced truncation [159, 176, 188, 270], proper orthogonal decomposition [43, 59, 61, 133, 147, 165, 220] and Krylov-based model order reduction [9, 19, 92, 94–96, 109]. A more detailed comparison of some model reduction techniques can be found in [40, 44, 86, 152]. It should also be noted that the existing methods are not strictly limited to the categories mentioned. There exist a variety of MOR techniques by combining the advantages of different techniques, see e.g. [112, 144, 157, 218, 233, 299].

Before we give an overview of the common MOR methods, let us briefly introduce the mathematical fundamentals that are based on the concept of projection operators, as well as some general background information on linear dynamical systems in a first step.

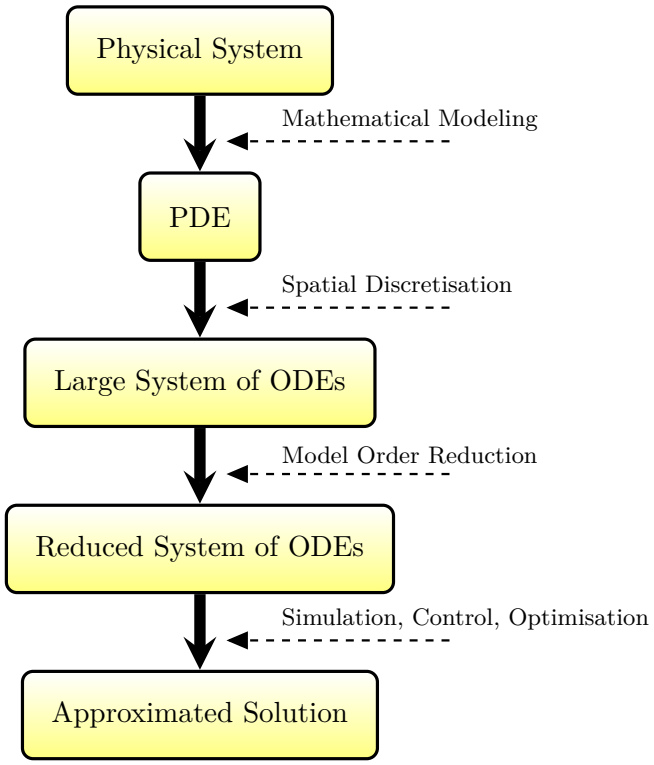


Figure 4.1: Motivation of the MOR framework.

4.1 Linear Dynamical Systems

A dynamical system is a system that is based firstly on state variables that completely describe the state of the system, and secondly on a dynamical rule or law that specifies the system changes over time. The intrinsic behaviour of a dynamical system is defined based on the dynamic and the initial state. Linear dynamical systems are characterised by a linear mapping, meaning that the system can be written in matrix form (4.1) with the matrix coefficients being constant. In systems theory, a linear dynamical system therefore describes an input-output behaviour relying on the relations between the input $\boldsymbol{w}(t)$ and initial conditions $\boldsymbol{u}(0)$ and the corresponding output response $\boldsymbol{y}(t)$. We recommend e.g. [93,143,234] for a more detailed description to the fundamentals in systems theory. Let us give a very brief insight.

4.1.1 Model Accuracy Measurement

As already mentioned, the main challenge for MOR methods is to construct a reduced order model (4.2) in such a way that the behaviour of the original system (4.1) is approximated. In order to describe how good a reduced system *approximates* the original system, a measure for quantifying the accuracy is required. This can be examined by defining the (time domain) error signal $E(t)$ as the deviation between two responses $\boldsymbol{y}(t)$ and $\boldsymbol{y}_r(t)$ from the original and the reduced model, respectively, in the form

$$E(t) = \|\boldsymbol{y}(t) - \boldsymbol{y}_r(t)\| \tag{4.3}$$

with a suitable vector norm. Consequently, the approximation concept is generally based on minimising the errors between the original and approximated responses. In particular, the behaviour of the response depends on the given initial condition $\boldsymbol{u}^0$ and the input $\boldsymbol{w}(t)$, as can be seen from the exact integration of (4.1) via

$$\boldsymbol{y}(t) = C\left(e^{Lt}\boldsymbol{u}^0 + \int_0^t e^{L(t-z)}K\boldsymbol{w}(z)\,\mathrm{d}z\right) \tag{4.4}$$

If $\boldsymbol{w}(t) = \boldsymbol{0}$ is assumed, the output is called the *zero-input response* and is defined exclusively on the basis of the initial condition $\boldsymbol{u}^0$. Otherwise, if $\boldsymbol{u}^0 = \boldsymbol{0}$, the output is called as the *zero-state response* and is defined solely by the input $\boldsymbol{w}(t)$. Thus, the response (4.4) of a linear dynamical system can be decomposed into the zero-state response and the zero-input response. On closer examination, by the use of the coordinate transformation $\widetilde{\boldsymbol{u}}(t) = \boldsymbol{u}(t) - \boldsymbol{u}^0$ the original system can be translated into a transformed system with zero initial conditions $\widetilde{\boldsymbol{u}}(0) = \boldsymbol{0}$. This observation suggests that the response of a system in general only depends on the zero-state response, so that we assume zero initial conditions in the further course. Overall, for a correct measurement (4.3) the difference between the responses should be compared at the same time instances and for the same input $\boldsymbol{w}(t)$. We stress that the above error signal is defined in the *time domain*, but such a characterisation can also be considered in the *frequency domain*. In other words, the error can be measured based on the frequency responses $\boldsymbol{Y}(s)$ and $\boldsymbol{Y}_r(s)$ of the original and the reduced system, respectively. As part of systems theory, the frequency response is defined by the transfer function of the system.

4.1.2 Transfer Function

The use of MOR methods is closely related to the *transfer function* $\boldsymbol{H}(s)$ of the underlying dynamical system (4.1). The transfer function represents the *input-output behaviour* depending on the input $\boldsymbol{w}(t)$ and the output $\boldsymbol{y}(t)$ in the frequency domain. Since the transfer function is defined in the frequency domain and describes a function of the frequency of the input signal, this terminology is traditionally linked to the *Laplace transform*. The Laplace transform $F(s)$ of a function $f(t)$ is defined by

$$F(s) = \int_0^\infty f(t)e^{-st}\,\mathrm{d}t \tag{4.5}$$

where $s \in \mathbb{C}$ is the frequency parameter. An alternative notion for the Laplace transform is $\mathcal{L}\{f(t)\}$. Applying the Laplace transform to the system (4.1) gives

$$s\boldsymbol{U}(s) - \boldsymbol{u}^0 = L\boldsymbol{U}(s) + K\boldsymbol{W}(s) \tag{4.6}$$

The assumption of zero initial conditions finally leads to

$$\begin{aligned} \boldsymbol{U}(s) &= (sI - L)^{-1} K\boldsymbol{W}(s) \\ \boldsymbol{Y}(s) &= C\,(sI - L)^{-1} K\boldsymbol{W}(s) \end{aligned} \tag{4.7}$$

where the inverse $(sI - L)^{-1}$ exists for $s \neq \lambda_i$ and where λ_i denotes the eigenvalues of L. For $s = \lambda_i$ the characteristic polynomial is zero, which is often referred to as the *pole* of the system. Based on the representation (4.7), the transfer function is defined as

$$\boldsymbol{H}(s) = C\,(sI - L)^{-1} K \tag{4.8}$$

and describes the direct relation between the input $\boldsymbol{W}(s)$ and the output $\boldsymbol{Y}(s)$ of the original system in the frequency domain via

$$\boldsymbol{Y}(s) = \boldsymbol{H}(s)\boldsymbol{W}(s) \tag{4.9}$$

Note that for SISO systems the transfer function will be a vector and for MIMO systems it gives a matrix, more precisely a matrix of transfer functions. The transfer function $\boldsymbol{H}_r(s)$ of the reduced model is defined in an analogous manner. As is known, the model accuracy of a reduced order model is measured based on the difference of the frequency responses $\boldsymbol{Y}(s)$ and $\boldsymbol{Y}_r(s)$. As the responses depend on their transfer functions, the approximation quality in the frequency domain is defined by

$$E(s) = \left\| \boldsymbol{H}(s) - \boldsymbol{H}_r(s) \right\| \tag{4.10}$$

with a suitable norm. This consequently implies that the approximation quality of a reduced system is linked to the approximation of the transfer function of the original system. As a result, the appropriate approximation of the transfer function

$$\boldsymbol{H}(s) \approx \boldsymbol{H}_r(s) \tag{4.11}$$

is the main task of MOR methods.

4.2 Projective Model Order Reduction

Projection operators play an important role in numerical linear algebra and are also of great importance for the derivation and development of MOR approaches. In the following the mathematical basics of projection [237] are explained and how this technique can be used in connection with the solution of dynamical systems.

4.2.1 Projection Operators

A projection is defined as a linear mapping of a vector space $\mathbb{R}^n$ into itself, which can be represented by a multiplication with a *projector* P in the form

$$\boldsymbol{x}_p = P\boldsymbol{x} \tag{4.12}$$

An illustrative example for a projection is the shadow cast of a three-dimensional object onto a two-dimensional plane. In fact, the image (range) of a projection is either a subspace of $\mathcal{V} \subset \mathbb{R}^n$ or $\mathbb{R}^n$ itself. In this framework, a matrix $P \in \mathbb{R}^{n\times n}$ is defined as a projector onto a subspace $\mathcal{V} \subseteq \mathbb{R}^n$ if it fulfils:

$$P^2 = P \quad \text{and} \quad \operatorname{im}(P) = \mathcal{V} \tag{4.13}$$

From this definition it follows that a repeated projection provides the same outcome

$$\boldsymbol{x}_p = P\boldsymbol{x}_p \tag{4.14}$$

In more detail, the eigenvalues of P are either zero and one, so any vector $\boldsymbol{x} \in \mathbb{R}^n$ can be specified as

$$\boldsymbol{x} = P\boldsymbol{x} + (I - P)\boldsymbol{x} \tag{4.15}$$

Consequently, the vector space $\mathbb{R}^n$ can be decomposed into two subspaces as

$$\mathbb{R}^n = \operatorname{span}(\mathcal{V} \oplus \mathcal{W}) \tag{4.16}$$

with $\mathcal{V} = \operatorname{im}(P)$ and $\mathcal{W} = \operatorname{null}(P)$. In other words, a projector P separates $\mathbb{R}^n$ into two subspaces and projects $\boldsymbol{x}$ onto $\mathcal{V}$ and along $\mathcal{W}$.

Matrix Representation In order to obtain such desired projectors P in matrix representation two bases are required: a basis $V = [v_1, \ldots, v_r]$ for the subspace $\mathcal{V} = \operatorname{im}(P)$ and another basis $W = [w_1, \ldots, w_r]$ for the subspace $\mathcal{W} = \operatorname{null}(P)$. Let us explain the derivation of a projector P as in (4.12) using the abovementioned example of the shadow cast in $\mathbb{R}^3$.

A vector $\boldsymbol{x} \in \mathbb{R}^3$ shall be projected onto a two-dimensional plane spanned by the columns of the matrix $V \in \mathbb{R}^{3\times 2}$, meaning that $\boldsymbol{x}$ is projected along the direction $\boldsymbol{x}_d$. To this end, a matrix $W \in \mathbb{R}^{3\times 2}$ is defined whose columns are orthogonal to $\boldsymbol{x}_d$, i.e.

$$W^\top \boldsymbol{x}_d = \boldsymbol{0} \tag{4.17}$$

Since the vector $\boldsymbol{x}_p$ should lie in V, $\boldsymbol{x}_p$ can be represented as a linear combination of the

columns of V via

$$\boldsymbol{x}_p = V\boldsymbol{r} \tag{4.18}$$

where $\boldsymbol{r} \in \mathbb{R}^2$ is unknown. Based on the relations of $\boldsymbol{x}, \boldsymbol{x}_d$ and $\boldsymbol{x}_p$, it follows that

$$\boldsymbol{x}_d = \boldsymbol{x} - \boldsymbol{x}_p \tag{4.19}$$

and inserting (4.18) and (4.19) into (4.17) yields

$$W^\top \boldsymbol{x}_d = W^\top \boldsymbol{x} - W^\top \boldsymbol{x}_p = W^\top \boldsymbol{x} - W^\top V\boldsymbol{r} = \boldsymbol{0} \tag{4.20}$$

The latter can be solved for $\boldsymbol{r}$ as

$$W^\top \boldsymbol{x} = W^\top V\boldsymbol{r} \quad \Longleftrightarrow \quad \boldsymbol{r} = \left(W^\top V\right)^{-1} W^\top \boldsymbol{x} \tag{4.21}$$

assuming that V and W have full rank and that the inverse of $W^\top V$ exists. Finally, with (4.18) the projector can be specified by

$$\boldsymbol{x}_p = V\boldsymbol{r} = V\left(W^\top V\right)^{-1} W^\top \boldsymbol{x} = P\boldsymbol{x} \tag{4.22}$$

with $P = V(W^\top V)^{-1}W^\top$. Obviously, the derived P is a projector since

$$P^2 = V\left(W^\top V\right)^{-1} \underbrace{W^\top V\left(W^\top V\right)^{-1}}_{I} W^\top = V\left(W^\top V\right)^{-1} W^\top = P \tag{4.23}$$

The formulation (4.22) describes more precisely a projection onto the subspace $\mathrm{im}(V)$ along the direction of the orthogonal complement of $\mathrm{im}(W)$. Furthermore, it can be shown that the assumption of invertibility of $W^\top V$ is fulfilled if no vector of V is orthogonal to W. Let us also stress that the projector is independent of the choice of the basis matrices V and W for the subspaces $\mathcal{V}$ and $\mathcal{W}$, respectively. Obviously, this derivation can also apply to higher dimensional projections.

It should be noted that the projection relation $\boldsymbol{x}_p = V\boldsymbol{r}$ is of particular importance as it describes a reduction in the order, where

$$\boldsymbol{x}_r := \boldsymbol{r} = \left(W^\top V\right)^{-1} W^\top \boldsymbol{x} \tag{4.24}$$

represents the reduced vector $\boldsymbol{x}_r$ of $\boldsymbol{x}$ with respect to the basis V.

Orthogonal and Skew Projection As part of projection, two main types are considered, an important class of projectors being obtained when the subspace $\mathcal{V}$ and $\mathcal{W}$ are *orthogonal*. The projector P is said to be orthogonal in an algebraic formulation when $P = P^\top$, otherwise the projection is called *skew*. The special choice $W = V$ automatically leads to an orthogonal projection $P = V(V^\top V)^{-1}V^\top$. If the projection matrices are chosen to be biorthogonal, i.e. $W^\top V = I$, then the projector is represented via $P = VW^\top$. Orthogonal projectors do not necessarily have biorthogonal bases, meaning that orthogonal projection and biorthogonal bases denote different characterisations. The choice $W = V$ is defined as *Galerkin projection*, otherwise using a skew projection is called a *Petrov-Galerkin projection*.

4.2.2 Model Reduction by Projection

The basic concept of MOR is to approximate the high dimensional state space vector $\boldsymbol{u}(t) \in \mathbb{R}^n$ by a reduced vector $\boldsymbol{u}_r(t) \in \mathbb{R}^r$ of lower dimension $r \ll n$. For this purpose, projection-based methods approximate the state space vector $\boldsymbol{u}(t)$ by a reduced basis

$$\boldsymbol{u}(t) \approx V\boldsymbol{u}_r(t) \tag{4.25}$$

where $V \in \mathbb{R}^{n\times r}$ characterises the projection matrix. An exact realisation in (4.25) is usually not fulfilled so that the difference is denoted by the error $\boldsymbol{e}(t) = \boldsymbol{u}(t) - V\boldsymbol{u}_r(t)$. Inserting the latter into the original system (4.1) results in

$$V\dot{\boldsymbol{u}}_r(t) = LV\boldsymbol{u}_r(t) + K\boldsymbol{w}(t) + \boldsymbol{\epsilon}(t) \tag{4.26}$$

where the residual $\boldsymbol{\epsilon}(t) = L\boldsymbol{e}(t) - \dot{\boldsymbol{e}}(t)$ contains the error terms caused by the approximation. The system is generally overdetermined since it has n equations but only r unknowns resulting from $\boldsymbol{u}_r(t)$. To obtain a unique solution, the system is projected onto the r-dimensional subspace of $\mathrm{im}(V)$ using a projector matrix $P = V(W^\top V)^{-1}W^\top$ in the form of

$$\begin{aligned} V\left(W^\top V\right)^{-1} W^\top V\dot{\boldsymbol{u}}_r(t) = V\left(W^\top V\right)^{-1} W^\top LV\boldsymbol{u}_r(t) \\ + V\left(W^\top V\right)^{-1} W^\top K\boldsymbol{w}(t) + V\left(W^\top V\right)^{-1} W^\top\boldsymbol{\epsilon}(t) \end{aligned} \tag{4.27}$$

The projected system can be solved exactly for any $\boldsymbol{\epsilon}(t)$ (generally not known), where the projection is chosen such that the projection of the residual $\boldsymbol{\epsilon}(t)$ becomes zero, i.e. $V(W^\top V)^{-1}W^\top\boldsymbol{\epsilon}(t) = \boldsymbol{0}$. This condition is known as the *Petrov-Galerkin condition* and should not be confused with the error $\boldsymbol{e}(t)$ because $V(W^\top V)^{-1}W^\top\boldsymbol{e}(t) \neq \boldsymbol{0}$. Assuming that the columns of V are linearly independent, then the matrix V in (4.27) can be omitted and the reduced model with r equations and r unknowns, can be solved exactly for $\boldsymbol{u}_r(t)$ as

$$\dot{\boldsymbol{u}}_r(t) = \left(W^\top V\right)^{-1} W^\top LV\boldsymbol{u}_r(t) + \left(W^\top V\right)^{-1} W^\top K\boldsymbol{w}(t) \tag{4.28}$$

Finally, by setting $L_r := (W^\top V)^{-1}W^\top LV$ and $K_r := (W^\top V)^{-1}W^\top K$ the reduced system can be rewritten in a more compact form

$$\dot{\boldsymbol{u}}_r(t) = L_r\boldsymbol{u}_r(t) + K_r\boldsymbol{w}(t) \tag{4.29}$$

If the bases V and W are biorthogonal, the reduced system (4.29) is defined via $L_r := W^\top LV$ and $K_r := W^\top K$. Obviously, solving the reduced model (4.28) always fulfils the Petrov-Galerkin condition, since

$$\begin{aligned} & \dot{\boldsymbol{u}}(t) - L\boldsymbol{u}(t) - K\boldsymbol{w}(t) = \boldsymbol{0} \\ \Longrightarrow\quad & P\left(\dot{\boldsymbol{u}}(t) - L\boldsymbol{u}(t) - K\boldsymbol{w}(t)\right) = \boldsymbol{0} \\ \Longrightarrow\quad & P\left(V\dot{\boldsymbol{u}}_r(t) - LV\boldsymbol{u}_r(t) - K\boldsymbol{w}(t)\right) - P\boldsymbol{\epsilon}(t) = \boldsymbol{0} \\ \Longrightarrow\quad & P\boldsymbol{\epsilon}(t) = \boldsymbol{0} \end{aligned} \tag{4.30}$$

Overall, we have described the basic MOR procedure for linear dynamical systems which is based on the projection of the original high dimensional model into a reduced low-dimensional

model. For this reason, the main task of a projection-based MOR method is to construct appropriate bases V and W.

4.2.3 Stability of Projection-Based Model Order Reduction

Another important aspect that should not be neglected is the stability of the reduced system (4.29). Therefore, when dealing with projection-based MOR methods, it is advantageous if the stability is preserved under projection. In particular, this means if L is a stable system matrix, the reduced matrix L_r should also be stable. In this context, we recall that a matrix is said to be *stable* if every eigenvalue of L has a negative real part. In engineering, a stable matrix is often referred to as a *Hurwitz matrix*. At this point it should be stressed that the stability property above should not be confused with the numerical stability of the underlying numerical scheme. It is clear that explicit methods remain conditionally stable when reduced order models are solved numerically.

Projection-based MOR techniques are generally not stability preserving during the reduction process, even if the original system is stable. In fact, the stability preservation can only be ensured if truncation[1] methods are used or in other special cases. In order to avoid this problem and to preserve stability, for example, certain postprocessing procedures can be applied, which normally cause higher computational costs or/and accuracy losses.

Fortunately, the stability of the reduced order model can be guaranteed if a one-sided projection method, i.e. $W = V$, coupled with a negative semi-definite system matrix L is applied. This is obviously a stronger requirement than just a stable system matrix, since a Hurwitz matrix may not be negative (semi-)definite. At this point we would like to point out again that model problems with negative semi-definite matrices are of particular interest in this thesis. To give sufficient conditions for the preservation of stability during the reduction, we recall the logarithmic norm as already introduced in Section 2.3.2.

The logarithmic norm of a matrix $L \in \mathbb{C}^{n\times n}$, also called *numerical abscissa* or *matrix measure*, is defined as

$$\mu_p(L) = \lim_{h\to 0^+} \frac{\|I + hL\|_p - 1}{h} \tag{4.31}$$

For the logarithmic norm holds

$$\alpha(L) \leq \mu_p(L) \tag{4.32}$$

where α is referred to as the *spectral abscissa* and is defined by

$$\alpha(L) = \max_i \{\mathrm{Re}(\lambda_i)\} \tag{4.33}$$

The fundamental result for stability analysis purposes is that the numerical abscissa satisfies

$$\left\|e^{tL}\right\|_p \leq e^{t\mu_p(L)}, \quad \forall t \geq 0 \tag{4.34}$$

As a result, the logarithmic norm implies the important property that the state $\boldsymbol{u}(t)$ of the

[1] Truncation methods are characterised by a transformation of the original model into an equivalent system, which can subsequently be truncated in a suitable manner so that the reduced system inherits stability. Typical representatives are modal coordinate reduction and balanced truncation methods.

homogeneous system $\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t)$ fulfils for any arbitrary initial state

$$\left\|\boldsymbol{u}(t)\right\|_p \leq \left\|e^{tL}\right\|_p \left\|\boldsymbol{u}(0)\right\|_p \leq e^{t\mu_p(L)} \left\|\boldsymbol{u}(0)\right\|_p, \quad \forall t \geq 0 \tag{4.35}$$

This in turn means that the system is stable if $\mu_p(L) \leq 0$, or equivalently $e^{t\mu_p(L)} \leq M$ with finite constant M independent of t. For real matrices, the matrix measure for the three most common norms is given by

$$\begin{aligned}
\mu_1(L) &= \max_j \left(a_{jj} + \sum_{i \neq j} |a_{ij}| \right) \\
\mu_2(L) &= \max_i \lambda_i \left(\frac{L + L^\top}{2} \right) \\
\mu_\infty(L) &= \max_i \left(a_{ii} + \sum_{j \neq i} |a_{ij}| \right)
\end{aligned} \tag{4.36}$$

In particular, it follows that

$$\mu_2(A) \leq 0 \quad \Longleftrightarrow \quad \langle \boldsymbol{x}, A\boldsymbol{x} \rangle_2 \leq 0, \quad \forall \boldsymbol{x} \in \mathbb{R}^n \tag{4.37}$$

Based on the concept of matrix measure, the sufficient condition for stability preservation in the Euclidean norm can be derived as follows:

Theorem 4.1 ([251]). *Given a continuous linear time-invariant system* (4.1) *with* $\mu_2(L) \leq 0$*, then the reduced system* (4.2) *is stable if an orthonormal one-sided projection matrix* $V \in \mathbb{R}^{n \times r}$ *is used, i.e.* $W = V$*.*

Proof. Since V is orthonormal, it holds that $\|V\|_2 = 1$. Using the logarithmic norm one can easily obtain

$$\begin{aligned}
\mu_2(L_r) &= \lim_{h \to 0^+} \frac{\|I_r + hL_r\|_2 - 1}{h} \\
&= \lim_{h \to 0^+} \frac{\|I_r + hV^\top LV\|_2 - 1}{h} \\
&\leq \lim_{h \to 0^+} \frac{\|V^\top\|_2 \|I + hL\|_2 \|V\|_2 - 1}{h} \\
&= \lim_{h \to 0^+} \frac{\|I + hL\|_2 - 1}{h} = \mu_2(L) \leq 0
\end{aligned} \tag{4.38}$$

□

Consequently, the stability preservation with respect to the definiteness of L can be characterised by the following lemma:

Lemma 4.1 ([251, 259]). *Given a continuous linear time-invariant system* (4.1) *with real negative semi-definite matrix* L*, then the reduced system* (4.2) *is stable if an orthonormal one-sided projection matrix* $V \in \mathbb{R}^{n \times r}$ *is used, i.e.* $W = V$*.*

Proof. The stability preservation results from (4.36) via

$$\begin{aligned}\mu_2(L_r) &= \max_i \lambda_i \left(\frac{L_r + L_r^\top}{2} \right) = \max_i \lambda_i \left(\frac{V^\top L V + \left(V^\top L V \right)^\top}{2} \right) \\ &= \max_i \lambda_i \left(\frac{V^\top L V + V^\top L^\top V}{2} \right) = \max_i \lambda_i \left(\frac{V^\top (L + L^\top) V}{2} \right) \\ &\leq \max_i \lambda_i \left(\frac{L + L^\top}{2} \right) = \mu_2(L) \leq 0 \end{aligned} \tag{4.39}$$

The condition $\mu_2(L) \leq 0$, or equivalently $L + L^\top \prec 0$, corresponds to the negative semi-definiteness of a matrix. □

This sufficient condition was also presented in connection with the passivity property of systems [95, 194]. Let us emphasise that the Theorem 4.1 is of crucial importance due to its generality, meaning that the stability preservation holds independently of the MOR method used to build the projection matrix V. Notably, the stability preservation is only guaranteed when using a one-sided projection method. The stability properties also apply to more general models such as $E\dot{\boldsymbol{u}}(t)$ or second order systems, see e.g. [201, 251].

4.2.4 Main Requirements for Model Order Reduction Methods

As already described, the main task of applying MOR is to reduce the computational costs of expensive numerical simulations. However, there are a number of requirements that must be satisfied, when using MOR techniques to extract a reduced order model from the original one. For this reason, we summarise the most important aspects and desirable characteristics.

Accuracy: The reduced model should capture the most dominant dynamics and provide an adequately accurate model for the original system, meaning that the associated error between the input-output behaviour in time (4.3) or frequency domain (4.10) should be small with respect to a certain norm. In this framework, the existence of error bounds is desirable in order to assess the suitability of reduced models.

System properties preservation: The reduced model should preserve the main physical properties of the original system, e.g. stability, conservation and boundedness.

Compactness: The number of state variables of the reduced model, specified by the order r, should be significantly small compared to the original model in order to exhibit cheap online costs for the simulation of the resulting reduced model.

Numerical efficiency: The MOR technique should be numerically efficient in order to avoid large offline costs for the computation of the reduced model. The construction process of the reduced basis should be simple and numerically as robust as possible independently of the order r, and cause low memory storage costs.

Automation: The model reduction algorithm should in some way be automatic, meaning that the algorithm offers minimal user intervention and is thus nearly free of design parameters.

In the following the four most common methods for constructing suitable projection matrices are presented.

4.3 Modal Coordinate Reduction

The simplest MOR technique, known as the *modal coordinate reduction (MCR)* method (or *modal truncation*), was introduced in the 1960s by Davison [76] and is based on the eigendecomposition of the underlying state matrix. The basic concept of MCR is to transform the original model from physical coordinates in physical space into *modal coordinates* in *modal space* using the eigenvector matrix and to reduce the transformed system such that the *dominant eigenvalues and eigenvectors* are retained in the reduced model. We mention that at the same time the approach was also described in structural dynamics by Guyan [115]. It is known from structural mechanics that the actual response of the elastic string vibration can be approximated quite well by considering only the few first (dominant) harmonics. From a mathematical point of view, the vibration *frequencies* and *modes* correspond to the *eigenvalues* and the *eigenfunctions*, respectively. As a result of the fact that large-scale systems arise both structural dynamics and systems theory, many methods have been developed independently.

On closer examination, the MCR concept using an eigendecomposition ansatz is closely related to the matrix decomposition methods or the kernel-based methods, see Sections 2.3.1 or 5.7. The kernel-based methods build on the analytical solution of the underlying PDE, with the kernel representing the fundamental solution for a linear partial differential operator. The fundamental solution can be expressed by a series of its eigenfunctions and eigenvalues, which means that the analytical solution is finally approximated by the set of dominant eigenfunctions which have the most contribution in such a series. However, analytical solutions of PDEs are generally only available for some simple geometries. For a practical example based on the heat equation, see e.g. [101].

To tackle this issue, the spatial discretisation of the PDE is first performed so that the eigendecomposition ansatz can then be applied to the semi-discretised model. More precisely, the original system in physical coordinates is transformed into modal coordinates, in which those modes are subsequently removed that have less important contributions to the system responses. In general, the MCR technique is highly beneficial when only a few modes have a significant influence on the system dynamics within the frequency range of interest. Let us describe the MCR technique using the semi-discrete SISO system

$$\begin{aligned} \dot{\boldsymbol{u}}(t) &= L\boldsymbol{u}(t) + \boldsymbol{k}w(t) \\ y(t) &= \boldsymbol{c}^\top \boldsymbol{u}(t) \end{aligned} \tag{4.40}$$

with a stable and diagonalisable matrix $L \in \mathbb{R}^{n\times n}$ as well as $\boldsymbol{k}, \boldsymbol{c}, \boldsymbol{u}(t) \in \mathbb{R}^n$ and $w(t), y(t) \in \mathbb{R}$. For a more detailed insight into MCR, we refer the reader to [45, 93, 222].

4.3.1 Modal Transformation

Since L being diagonalisable, there exists a matrix $\Phi \in \mathbb{R}^{n\times n}$ with eigenvectors of L and a diagonal matrix $\Lambda \in \mathbb{R}^{n\times n}$ with the corresponding eigenvalues λ_i so that $L = \Phi\Lambda\Phi^{-1}$.

Inserting the latter identity into (4.40), the subsequent multiplication of Φ^{-1} leads to

$$\begin{aligned} \Phi^{-1}\dot{\boldsymbol{u}}(t) &= \Lambda\Phi^{-1}\boldsymbol{u}(t) + \Phi^{-1}\boldsymbol{k}w(t) \\ y(t) &= \boldsymbol{c}^\top\boldsymbol{u}(t) \end{aligned} \tag{4.41}$$

By changing the basis

$$\Phi^{-1}\boldsymbol{u}(t) = \boldsymbol{z}(t) \quad \Longleftrightarrow \quad \boldsymbol{u}(t) = \Phi\boldsymbol{z}(t) \tag{4.42}$$

also called regular (*modal*) transformation, the physical space representation (4.40) can be rewritten in modal coordinates as

$$\begin{aligned} \dot{\boldsymbol{z}}(t) &= \Lambda\boldsymbol{z}(t) + \widetilde{\boldsymbol{k}}w(t) \\ y(t) &= \widetilde{\boldsymbol{c}}^\top\boldsymbol{z}(t) \end{aligned} \tag{4.43}$$

with $\widetilde{\boldsymbol{c}}^\top = \boldsymbol{c}^\top\Phi$ and $\widetilde{\boldsymbol{k}} = \Phi^{-1}\boldsymbol{k}$. Obviously, the ODE system (4.43) is decoupled into n differential equations in the modal coordinates

$$z_i'(t) = \lambda_i z_i(t) + \widetilde{k}_i w(t), \quad i = 1, \ldots, n \tag{4.44}$$

Applying the Laplace transform to (4.44) yields

$$z_i(s) = \frac{1}{s - \lambda_i}\widetilde{k}_i w(s), \quad i = 1, \ldots, n \tag{4.45}$$

so that the response is represented by the composition of decoupled paths as follows:

$$y(s) = \sum_{i=1}^{n} \widetilde{c}_i z_i(s) \tag{4.46}$$

Otherwise, the defined transfer function (4.8) is given in the simple form

$$\boldsymbol{H}(s) = \widetilde{\boldsymbol{c}}^\top \left(sI - \Lambda\right)^{-1} \widetilde{\boldsymbol{k}} = \sum_{i=1}^{n} \frac{\widetilde{c}_i\widetilde{k}_i}{s - \lambda_i} \tag{4.47}$$

Comparing (4.46) and (4.47) shows that each decoupled ODE with the eigenvalue λ_i and the modal coordinate z_i represents exactly one path within the transfer function.

Remark 4.1. *Note that the approach mentioned is only applicable if the underlying matrix L is orthogonal diagonalisable. Although the discrete Laplacian is generally not symmetric, the properties such as the orthogonality of the eigenvectors can be preserved for most applications. Thus, the "desire" for a symmetric (diagonalisable) matrix can be recovered using an appropriate scalar product. More precisely, the mutual orthogonality can be achieved if the matrix L can be factorised in two symmetric matrices $L = AB$, with A also positive definite. This factorisation is fulfilled for any matrix that has real eigenvalues and a complete set of eigenvectors, see [46]. This will be important for the practical application in Chapter 5.*

Dominant Modal Coordinates The representation (4.47) illustrates that a modal coordinate z_i can be removed without affecting the input-output-relation, if either $\widetilde{c}_i$ or $\widetilde{k}_i$ zero. In

addition, each eigenvalue λ_i itself represents a certain significance in the transfer behaviour. These observations are the basis for the simplest MOR method, meaning to remove weak parts z_i if $\widetilde{c}_i$ or $\widetilde{k}_i$ or both are small as well as to remove z_i if λ_i has less contributions to the system responses.

The main strategy is now to identify those states z_i such that their elimination has a minimal influence on the system's behaviour. The reduction process results in a reduced order model, whereby the approximation quality of the r-dimensional reduced system being given by the error bound

$$\|\boldsymbol{H}(s) - \boldsymbol{H}_r(s)\| \leq \sum_{j=r+1}^{n} \frac{|\widetilde{c}_j| \left|\widetilde{k}_j\right|}{|s - \lambda_j|} = \sum_{j=r+1}^{n} \frac{|\widetilde{c}_j| \left|\widetilde{k}_j\right|}{|\mathrm{Re}(\lambda_j)|} \tag{4.48}$$

The identification of the relevant eigenvalues and eigenvectors is the main task of the MCR method, as this determines the approximation quality of the reduced model. However, deciding which modal coordinates may be neglected during the reduction process is not a trivial task. A closer look at (4.48) indicates that the eigenvalues close to the imaginary axis have a strong influence on the system's behaviour. This means that low frequencies, which correspond to small eigenvalues, usually dominate the dynamics of the underlying physical system. Therefore, a widespread and classic approach is to order the eigenvalues with respect to the distance to the imaginary axis:

$$0 > \mathrm{Re}(\lambda_1) \geq \mathrm{Re}(\lambda_2) \geq \cdots \geq \mathrm{Re}(\lambda_n) \tag{4.49}$$

In practice, a dominance analysis based only on the position of the eigenvalues in relation to the imaginary axis can lead to misjudgements of the system dominance behaviour. For this reason, the contributions of the modal coordinates are often linked to the dominance measure proposed by Litz. A more detailed explanation and discussion can be found in [93].

4.3.2 Modal Truncation

Assuming that the modes are provided, the modal coordinates representation (4.43) can be rearranged so that the eigenvalues are sorted from high to low dominance:

$$\begin{aligned} \begin{pmatrix} \dot{\boldsymbol{z}}_1(t) \\ \dot{\boldsymbol{z}}_2(t) \end{pmatrix} &= \begin{pmatrix} \Lambda_1 & 0^{r\times(n-r)} \\ 0^{(n-r)\times r} & \Lambda_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{z}_1(t) \\ \boldsymbol{z}_2(t) \end{pmatrix} + \begin{pmatrix} \widetilde{\boldsymbol{k}}_1 \\ \widetilde{\boldsymbol{k}}_2 \end{pmatrix} w(t) \\ y(t) &= (\widetilde{\boldsymbol{c}}_1^\top \widetilde{\boldsymbol{c}}_2^\top) \begin{pmatrix} \boldsymbol{z}_1(t) \\ \boldsymbol{z}_2(t) \end{pmatrix} \end{aligned} \tag{4.50}$$

with $\Lambda_1 \in \mathbb{R}^{r\times r}, \Lambda_2 \in \mathbb{R}^{(n-r)\times(n-r)}$ and $\widetilde{\boldsymbol{k}}_1, \widetilde{\boldsymbol{c}}_1^\top, \boldsymbol{z}_1(t) \in \mathbb{R}^r$ as well as $\widetilde{\boldsymbol{k}}_2, \widetilde{\boldsymbol{c}}_2^\top, \boldsymbol{z}_2(t) \in \mathbb{R}^{(n-r)}$. Based on this ordered representation, the reduced model of order r is obtained by *truncating* the nondominant subsystem

$$\begin{aligned} \dot{\boldsymbol{z}}_1(t) &= \Lambda_1 \boldsymbol{z}_1(t) + \widetilde{\boldsymbol{k}}_1 w(t) \\ y(t) &= \widetilde{\boldsymbol{c}}_1^\top \boldsymbol{z}_1(t) \end{aligned} \tag{4.51}$$

This low dimensional system is much faster to solve than the original one, but choosing a suitable order r depends on the model problem. We note that the stability of the reduced system (4.51) is guaranteed if the original system itself is stable, since the eigenvalues of the reduced system are a subset of the eigenvalues of the original system.

Computational Aspects The application of the MCR method depends strongly on the number of the eigenvalues that are required to obtain a desirable approximation quality of the reduced model. Based on the fact that there exists efficient solvers to compute the eigenvalues and eigenvectors, the MCR method can also be used for large-scale systems. Nevertheless, the technique is only practicable for a suitable number of eigenvalues. As stated in (4.49), the complete eigendecomposition is actually not necessary, meaning that the smallest eigenvalues of sparse matrices are computed iteratively in an efficient way. Furthermore, eigenmodes associated with other specific, relevant frequency ranges can also be computed.

The MCR method is considered as an automatic method, since the choice of r depends in a certain way on the growth rate of the eigenvalues, cf. (4.48). However, a good value for r is strongly depending on the model problem. If, for instance, the heat equation is considered, the modes are related to the geometry of the set-up, the boundary conditions and the material parameters such as the thermal diffusivity of the medium. Therefore, several thousand eigenfunctions may be required for an appropriate representation of a temperature field of a more complex model problem.

Obviously, MCR is well applicable to MIMO systems and is not directly limited by the size of the input matrix. Nonetheless, MCR is usually limited in such a way that the higher the number of inputs, the higher the number of dominant modes. But as the number of dominant poles is higher, the reduced system is larger, which shows the indirect dependence. As a consequence, a higher number of dominant modes directly increase the order of the reduced system, which shows the indirect dependence to the input matrix.

4.3.3 Modal Coordinate Reduction as Projection

Finally, we illustrate that the MCR technique belongs to the class of projection-based model reduction methods. In other words, the reduced model (4.51) can also be obtained directly by the projection (4.28), i.e.

$$L_r = \left(W^\top V\right)^{-1} W^\top L V, \quad \boldsymbol{k}_r = \left(W^\top V\right)^{-1} W^\top \boldsymbol{k}, \quad \boldsymbol{c}^\top = c^\top V \tag{4.52}$$

In doing so, the projection matrices V and $W^\top$ are given by the right eigenvectors $\boldsymbol{a}_i$ and left eigenvectors $\boldsymbol{b}_i^\top$ for $i = 1, \ldots, r$ corresponding to the r dominant eigenvalues via

$$V = A_r = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_r], \quad W^\top = B_r^\top = \begin{bmatrix} \boldsymbol{b}_1^\top \\ \vdots \\ \boldsymbol{b}_r^\top \end{bmatrix} \tag{4.53}$$

where the eigenvectors fulfil

$$\begin{aligned} L\boldsymbol{a}_i = \lambda_i \boldsymbol{a}_i \quad &\Longleftrightarrow \quad L A_r = A_r \Lambda_r \\ \boldsymbol{b}_i^\top L = \lambda_i \boldsymbol{b}_i^\top \quad &\Longleftrightarrow \quad B_r^\top L = \Lambda_r B_r \end{aligned} \tag{4.54}$$

Obviously, the matrices are not necessary biorthogonal, meaning $W^\top V \neq I_r$. If the matrix L is symmetric, then $\boldsymbol{a}_i = \boldsymbol{b}_i$ and thus $V = W$. In an analogous manner, let the matrices V and $W^\top$ be given by the eigenvector matrix and its inverse

$$V = \Phi_r, \quad W^\top = \Phi_r^{-1} \tag{4.55}$$

then it holds that $\boldsymbol{\phi}_i^\top \boldsymbol{\phi}_j = \delta_{i,j}$ with Kronecker delta $\delta_{i,j}$. Consequently, the matrices are biorthogonal $W^\top V = I_r$ and this leads to $B_r^\top = \Phi_r^{-1}$. In total, the MCR method can be interpreted as a projection onto the r-dimensional subspace of the dominant right eigenvectors and orthogonal to the subspace of the left eigenvectors.

4.4 Balanced Truncation

A common method for MOR is the *balanced truncation (BT)* method (or *truncated balanced realisation*) which was introduced by Moore [188] in the 1980s. In an analogous manner to MCR, the concept of BT is a transformation to a *balanced representation* so that nondominant variables can be easily identified. Then the reduced model is obtained by truncating these less important states. The BT method preserves the stability of the original system [216] and also provides a global a priori error bound [84] which is based on the difference of the transfer functions between the full and the reduced model. Let us give a brief introduction of BT.

The basic idea of BT is to preserve those states for which the least energy to be controlled is required and simultaneously provide the most energy through observation. Otherwise, state variables that are difficult to stimulate and/or to observe have a less contribution to the input-output behaviour and will be neglected. For this reason, the important properties of a dynamical system, known as *controllability* and *observability*, play a crucial role. In particular, the controllability describes the relation between the input $\boldsymbol{w}(t)$ and the state $\boldsymbol{u}(t)$, which means that the system can be transferred from the initial state $\boldsymbol{u}(0) = \boldsymbol{0}$ to any arbitrary final state $\boldsymbol{u}(t_F) = \boldsymbol{u}_{t_F}$. On the other hand, the observability characterises the connection between the state $\boldsymbol{u}(t)$ and the output $\boldsymbol{y}(t)$ such that the initial state $\boldsymbol{u}(0) = \boldsymbol{u}^0$ can be uniquely determined solely from $\boldsymbol{y}(t)$ and the known input $\boldsymbol{w}(t)$. In order to adequately asses controllability and observability, a measurement is needed, whereby the *(generalised) energy* is normally used.

Observability With regard to observability, the question arises which output energy in the steady state results from a given initial state. Assuming that $\boldsymbol{u}(0) = \boldsymbol{u}^0$ is given and $\boldsymbol{w}(t) = \boldsymbol{0}$, then the solution of (4.1) reads as $\boldsymbol{y}(t) = Ce^{Lt}\boldsymbol{u}^0$. The output energy dependent on $\boldsymbol{u}^0$ using the $\mathcal{L}_2$-norm is defined as

$$\begin{aligned}\int_0^\infty \boldsymbol{y}^2(t)\,\mathrm{d}t = \int_0^\infty \boldsymbol{y}^\top(t)\boldsymbol{y}(t)\,\mathrm{d}t &= \int_0^\infty (\boldsymbol{u}^0)^\top e^{L^\top t} C^\top C e^{Lt}\boldsymbol{u}^0\,\mathrm{d}t \\ &= \left(\boldsymbol{u}^0\right)^\top \int_0^\infty e^{L^\top t} C^\top C e^{Lt}\,\mathrm{d}t\ \boldsymbol{u}^0 \\ &= \left(\boldsymbol{u}^0\right)^\top W_o \boldsymbol{u}^0 \end{aligned} \tag{4.56}$$

with $W_o = \int_0^\infty e^{L^\top t} C^\top C e^{Lt}\,\mathrm{d}t$, which is often referred to as the *observability Gramian*. The product (4.56) describes the energy that the state $\boldsymbol{u}^0$ provides by observing the output. Moreover, considering the eigenvalues and eigenvectors of W_o yield

$$W_o \boldsymbol{\phi}_{o,i} = \lambda_{o,i} \boldsymbol{\phi}_{o,i} \tag{4.57}$$

If $\boldsymbol{u}^0 = \boldsymbol{\phi}_{o,i}$ then

$$\mathcal{E}_o = \boldsymbol{\phi}_{o,i}^\top \lambda_{o,i} \boldsymbol{\phi}_{o,i} \tag{4.58}$$

Therefore, large eigenvalues (singular values) of the observability Gramian usually provide useful information as the corresponding eigenvectors point in the directions that generate the most energy. In simple terms, the eigenvectors corresponding to large eigenvalues of W_o are easy to observe since (4.58) is large.

Controllability In an analogous manner it should be evaluated, which minimum input energy is required to transfer the state variable from zero to the final state. This task can be formulated as a linear optimisation problem via

$$\begin{aligned} \text{minimise} \quad & J = \int_0^{t_F} \boldsymbol{w}^2(t)\,\mathrm{d}t \\ \text{subject to} \quad & \dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + K\boldsymbol{w}(t) \\ & \boldsymbol{u}(0) = \boldsymbol{0}, \quad \boldsymbol{u}(t_F) = \boldsymbol{u}_{t_F} \end{aligned} \tag{4.59}$$

where t_F is free and is thus part of the optimisation itself. To address this problem, the substitution $\delta = t_F - t$ can be used, so that by setting $t_F \to \infty$, the optimisation variable t_F is eliminated. Finally, this yields the optimisation problem

$$\begin{aligned} \text{minimise} \quad & J = \int_0^{\infty} \boldsymbol{w}^2(\delta)\,\mathrm{d}\delta \\ \text{subject to} \quad & \dot{\boldsymbol{u}}(\delta) = -L\boldsymbol{u}(\delta) - K\boldsymbol{w}(\delta) \\ & \boldsymbol{u}(0) = \boldsymbol{u}_{t_F}, \quad \lim_{\delta\to\infty} \boldsymbol{u}(\delta) = \boldsymbol{0} \end{aligned} \tag{4.60}$$

The solution to this problem with minimum energy results from the control law

$$\boldsymbol{w}^\star(t) = K^\top e^{-L^\top t} W_c^{-1} \boldsymbol{u}_{t_F} \tag{4.61}$$

while for the minimum energy it holds that

$$\mathcal{E}_c = \boldsymbol{u}_{t_F}^\top W_c^{-1} \boldsymbol{u}_{t_F} \tag{4.62}$$

The matrix W_c is called the *controllability Gramian*, which is explicitly given by

$$W_c = \int_0^{\infty} e^{Lt} K K^\top e^{L^\top t}\,\mathrm{d}t \tag{4.63}$$

In particular, the measure (4.62) describes the minimum energy that is needed to reach the final state. Considering the eigenvalues and eigenvectors of W_c gives

$$W_c \boldsymbol{\phi}_{c,i} = \lambda_{c,i} \boldsymbol{\phi}_{c,i} \tag{4.64}$$

so that $\boldsymbol{u}_{t_F} = \boldsymbol{\phi}_{c,i}$ implies

$$\mathcal{E}_c = \boldsymbol{\phi}_{c,i}^\top \frac{1}{\lambda_{c,i}} \boldsymbol{\phi}_{c,i} \tag{4.65}$$

In the case of controllability the largest eigenvalues also provide significant information, since the corresponding eigenvectors point in the direction that generates minimum energy. More precisely, the eigenvectors corresponding to large eigenvalues of W_c^{-1} are easy to reach as the energy (4.62) is small.

Lyapunov Equations The main task of the BT method is based on the computation of the controllability Gramian and the observability Gramian in order to transform the system in its balanced realisation form. In practice, the Gramians W_c and W_o can be found as the solutions of the two Lyapunov equations

$$LW_c + W_c L^\top + KK^\top = 0, \quad L^\top W_o + W_o L + C^\top C = 0 \tag{4.66}$$

under the assumption that the dynamical system is stable. Let us mention that the Lyapunov equation is a special case of the *Sylvester equation*. The relation of BT to the Lyapunov equation can also be presented, if W_c is used directly as the solution of (4.66) because

$$\begin{aligned} LW_c + W_c L^\top &= \int_0^\infty L e^{Lt} KK^\top e^{L^\top t} \,\mathrm{d}t + \int_0^\infty e^{Lt} KK^\top e^{L^\top t} L^\top \,\mathrm{d}t \\ &= \int_0^\infty \frac{\mathrm{d}}{\mathrm{d}t} \left(e^{Lt} KK^\top e^{L^\top t} \right) \mathrm{d}t = -KK^\top \end{aligned} \tag{4.67}$$

The latter calculation uses the fact $\lim_{t\to\infty} e^{Lt} = 0$, which holds for stable matrices L in which all eigenvalues have a negative real part. In addition, the Gramians as the solution of the Lyapunov equation are unique and positive definite if L is stable.

4.4.1 Balancing

As already mentioned, the basic BT concept is based on identifying and removing the states that are the least controllable and observable at the same time. This task is realised by the so-called *balancing transformation* or *balancing realisation*. In doing so, a transformation is applied in which the Gramians are diagonal, equal and have Hankel singular values on their diagonal. The Hankel singular values are defined by the positive square roots of the eigenvalues of the Gramians product and identify significant dynamic modes associated with the input-output behaviour of the system.

In order to detect the significant directions, a balancing transformation $\boldsymbol{z} = T\boldsymbol{u}$ is applied which balances the given dynamical system. A system is said to be *balanced* if the Gramians are diagonal and equal:

$$W_c = W_o = \mathrm{diag}(\sigma_1, \sigma_2, \ldots \sigma_n) \tag{4.68}$$

When using a nonsingular transformation, the transformed Gramians are given by

$$\widetilde{W}_c = TW_cT^\top, \quad \widetilde{W}_o = T^{-T}W_oT^{-1} \tag{4.69}$$

Obviously, the eigenvalues of W_c and W_o change using this transformation, however, they remain invariant for the product W_cW_o under any state coordinate transformation, since

$$\widetilde{W}_c\widetilde{W}_o = TW_cT^\top T^{-T}W_oT^{-1} = T\left(W_cW_o\right)T^{-1} \tag{4.70}$$

In other words, the eigenvalues of W_cW_o, known as the *Hankel singular values*, defined as

$$\sigma_i = \sqrt{\lambda_i\left(W_cW_o\right)}, \quad i = 1, \ldots, n \tag{4.71}$$

are independent of the representation of the system. This observation is the basis of BT, as the product W_cW_o simultaneously contains the information about the controllable and the observable states, which are defined by the Hankel singular values.

Finally, a representation of the state transformation T is required, which the system transforms into a balanced representation. This can be achieved by using first the Cholesky factorisation and second the SVD as follows:

1.) Cholesky factorisation of

$$W_c = R_cR_c^\top, \quad W_o = R_oR_o^\top \tag{4.72}$$

where R_c and R_o are lower triangular matrices.

2.) SVD of

$$R_o^\top R_c = U\Sigma\widetilde{U}^\top \tag{4.73}$$

3.) Computing the state transformations by

$$T := \Sigma^{-\frac{1}{2}}U^\top R_o^\top, \quad T^{-1} := R_c\widetilde{U}\Sigma^{-\frac{1}{2}} \tag{4.74}$$

As a matter of fact, this procedure achieves a balanced system because

$$\begin{aligned}
\widetilde{W}_c &= TW_cT^\top \\
&= \Sigma^{-\frac{1}{2}}U^\top R_o^\top \left(R_cR_c^\top\right)\left(\Sigma^{-\frac{1}{2}}U^\top R_o^\top\right)^\top \\
&= \Sigma^{-\frac{1}{2}}U^\top R_o^\top \left(R_cR_c^\top\right) R_oU\Sigma^{-\frac{1}{2}} \\
&= \Sigma^{-\frac{1}{2}}U^\top U\Sigma\widetilde{U}^\top\widetilde{U}\Sigma U^\top U\Sigma^{-\frac{1}{2}} \\
&= \Sigma^{-\frac{1}{2}}\Sigma^2\Sigma^{-\frac{1}{2}} = \Sigma
\end{aligned} \tag{4.75}$$

with $\widetilde{U}^\top\widetilde{U} = U^\top U = I$. The same holds for W_o in an analogous manner.

Let us stress that the original BT method by Moore [188] was developed through spectral decompositions, but the calculation method presented above, cf. [159], is a more efficient and numerically more robust version.

4.4.2 Truncation

Once the balanced representation (after transformation $\boldsymbol{z} = T\boldsymbol{u}$) has been created, in which the state variables are ordered according to their importance from high to low due to their sorting σ_i, the order reduction can be performed analogously to the MCR method by truncation. In practice, however, the balancing and truncating is done in an alternative way without calculating the complete balanced system. Usually, the projection matrices $V \in \mathbb{R}^{n\times r}$ and $W^\top \in \mathbb{R}^{r\times n}$ are determined via the state transformation (4.74) by means of

$$V := R_c \widetilde{U}_r \Sigma_r^{-\frac{1}{2}}, \quad W^\top := \Sigma_r^{-\frac{1}{2}} U_r^\top R_o^\top \tag{4.76}$$

Based on this construction, the projection matrices are biorthogonal, since it holds that

$$W^\top V = \Sigma_r^{-\frac{1}{2}} U_r^\top R_o^\top R_c \widetilde{U}_r \Sigma_r^{-\frac{1}{2}} = \Sigma_r^{-\frac{1}{2}} U_r^\top U_r \Sigma_r \widetilde{U}_r^\top \widetilde{U}_r \Sigma_r^{-\frac{1}{2}} = I_r \tag{4.77}$$

Finally, the reduced matrices of order r can be computed with

$$L_r = W^\top L V, \quad K_r = W^\top K, \quad C_r = CV \tag{4.78}$$

and the reduced system is given by

$$\begin{aligned} \dot{\boldsymbol{u}}_r(t) &= L_r \boldsymbol{u}_r + K_r \boldsymbol{w}(t) \\ \boldsymbol{y}_r(t) &= C_r \boldsymbol{u}_r \end{aligned} \tag{4.79}$$

Besides the stability preservation (assuming that the original system is stable), which results from the positive definite solutions of the Lyapunov equations, the most important characteristic of BT is that a global error bound is provided by the Hankel singular values that are truncated

$$\left\| \boldsymbol{H}(s) - \boldsymbol{H}_r(s) \right\|_\infty \leq 2 \sum_{j=r+1}^{n} \sigma_j \tag{4.80}$$

Consequently, the practical choice of r depends only on the decay rate of the Hankel singular values. This means that the BT method can be used automatically by setting a user-defined error bound and finding the smallest possible dimension r of the reduced system, which satisfies that bound. More details can be found, e.g. in [8, 106, 176, 193].

It should be noted that the error bound (4.80) is based on the assumption of zero initial conditions and in general no longer holds if $\boldsymbol{u}^0 \neq \boldsymbol{0}$. However, this issue can be resolved for instance by transforming the system to zero initial conditions. Some other methods for dealing with linear systems and inhomogeneous initial conditions are proposed in [27, 71, 124, 170].

Computational Aspects In total, the application of BT requires five calculation steps. At first, the Gramians W_c and W_o are computed by solving the Lyapunov equations (4.66). Then a Cholesky factorisation and an SVD are performed, and the projection matrices V, W are calculated as mentioned in (4.72)-(4.74). Finally, the system is projected to obtain a reduced order model (4.78)-(4.79). The BT method is clearly performed without computing a complete SVD, as only the largest Hankel singular values are of interest, which can be computed iteratively.

While BT is attractive in theory and in practice provides accurate reduced order models, its use is generally limited to medium-sized systems. The main disadvantage of the BT method is directly related to the computation complexity for solving the Lyapunov equations (4.66). Due to the fact that the matrices involved are of the size of the original model, the BT technique is linked to high computational costs. Even if the system matrix L is sparse, the Gramians are generally dense and cause very high memory requirements. In order to solve the Lyapunov equation there exists direct and iterative methods. Usually, Lyapunov equations are solved with the standard Bartels-Stewart algorithm [23] which requires the computation of the Schur form. Some interesting iterative methods are based on low-rank approximations via ADI iteration [163], the sign function [230] or Krylov subspaces [260]. For an overview of the solution techniques we refer to [36,38,149]. In connection with MOR, some discussion on efficient methods for *approximate BT* are presented in [25, 37, 239, 240]. Let us stress that the stated beneficial properties of BT are generally lost due to the approximate solution of the Lyapunov equations.

Although in recent years many techniques have been developed and the iterative methods that compute an approximation of the Sylvester equation have become more efficient, the BT method is generally not preferred for large-scale systems because the approximation of the Lyapunov equation is still quite computationally intensive. Overall, the BT technique is naturally applied to problem sizes with dimensions up to $\mathcal{O}(10^5) - \mathcal{O}(10^6)$. As a consequence, the BT method is less used in connection with parabolic PDEs, see e.g. [35, 40, 123, 150, 239].

As in the case of MCR, the BT method has no direct dependence on the number of inputs. Nevertheless, a large-scale input implies a larger controllable and observable subspace of the system states. As the weak controllable and observable states are to be eliminated, the reduced system size consequently becomes larger. This can be directly observed during the BT reduction process as the Hankel singular values typically decay much more slowly when a large number of inputs are taken into account.

4.5 Krylov-Based Model Order Reduction

Nowadays, the most frequently used MOR method for linear and large-scale dynamical systems is moment matching using Krylov subspaces, which is referred to as *Krylov-based model order reduction*. In the 1970s and 1980s, moment matching methods [52, 77, 252] were initially developed which built a reduced order model from an explicit knowledge of the Padé approximant. The underlying basic concept gained more interest as the asymptotic waveform evaluation method was introduced in [219]. Nonetheless, moment matching only became a popular tool as numerically reliable algorithms have been proposed that are implicitly based on Krylov subspaces [89]. Implicit moment matching methods generally construct an accurate reduced order model highly efficiently, so this technique is applied to large-scale systems ranging from circuit simulation, power systems, electromagnetics, descriptor systems, control designs to electro-thermal processes or heat conduction models e.g. [29, 58, 89, 95, 266].

The basic idea of moment matching methods is the local approximation of the transfer function $\boldsymbol{H}(s)$ around a frequency σ of interest. In doing so, the transfer function is expanded into an infinite Taylor series in which the coefficients are defined as the moments. Based on this representation, the reduced order model is constructed so that some of the first moments of the original and reduced system are matched, which reflects why the approach

is called *moment matching*. Within this approach, the Krylov subspaces are required in order to construct the projection matrices that correspond to form bases of these subspaces when performing moment matching. For this purpose, efficient iterative methods such as the Lanczos and the Arnoldi algorithms are employed. Since implicit moment matching is based on Krylov subspace methods, mainly sparse matrix-vector multiplications are involved in the reduction process. As a result, Krylov-based MOR is predestined for the reduction of large-scale systems and provides a superior computational technique compared to MCR and BT. However, unlike the two truncation methods, there is generally no guarantee for stability preservation. As is known, the stability is preserved under certain assumptions without additional computational effort, cf. Theorem 4.1. Otherwise, there are some algorithms that can preserve stability, e.g. the ISRK algorithm that combines BT with Krylov-based MOR.

Although moment matching is indeed a simple and powerful tool, its use generally leads difficulties in practice. This essentially follows from the fact that there are no general, efficiently computable error bounds for the approximation quality and that several additional parameters (order r, expansion point σ) within the approach normally have to be determined by the user. To overcome this issue, the CURE algorithm developed in [200] can provide rigorous error bounds and automatic strategies for the selection of the expansion points.

In the following the basics of the Krylov-based MOR technique are presented. For a comprehensive introduction we refer the reader to [19, 29, 96, 109, 125, 167, 242, 243].

4.5.1 Moments

Assuming that the inverse $(sI - L)^{-1}$ exists, the transfer function of the semi-discrete system (4.1) reads as

$$\boldsymbol{H}(s) = C\,(sI - L)^{-1}\,K \tag{4.81}$$

which describes the relation between the input $\boldsymbol{W}(s)$ and the output $\boldsymbol{Y}(s)$ in the frequency domain. Another important representation of the transfer function is based on the Taylor expansion around a (complex) frequency σ defined as

$$\boldsymbol{H}(s) = -\sum_{k=0}^{\infty} m_k(\sigma)(s-\sigma)^k \tag{4.82}$$

where the coefficients $m_k(\sigma)$ are called the *moments* of the transfer function. In particular, the k-th moment around σ is given by

$$m_k(\sigma) = C(L - \sigma I)^{-(k+1)}K \tag{4.83}$$

The equivalence between the two formulations above can be specified as follows: first, the transfer function (4.81) can be expressed as

$$\begin{aligned}
\boldsymbol{H}(s) &= C(sI - L)^{-1}K \\
&= C\Big((\sigma I - L) + (s-\sigma)(\sigma I - L)^{-1}(\sigma I - L)\Big)^{-1}K \\
&= C\Big(I + (s-\sigma)(\sigma I - L)^{-1}\Big)^{-1}(\sigma I - L)^{-1}K \\
&= -C\Big(I - (s-\sigma)(-\sigma I + L)^{-1}\Big)^{-1}(-\sigma I + L)^{-1}K
\end{aligned} \tag{4.84}$$

and by denoting $\widetilde{L} = (-\sigma I + L)^{-1}$ one obtains

$$\boldsymbol{H}(s) = -C\Big(I - (s-\sigma)\widetilde{L}\Big)^{-1}\widetilde{L}K \tag{4.85}$$

Furthermore, by making use of the Neumann series

$$(I - T)^{-1} = \sum_{i=0}^{\infty} T^i \tag{4.86}$$

(4.85) can be extended with $T := (s-\sigma)\widetilde{L}$ into the Taylor series

$$\boldsymbol{H}(s) = -C\Big(I + (s-\sigma)\widetilde{L} + (s-\sigma)^2\widetilde{L}^2 + \dots\Big)\widetilde{L}K = -\sum_{k=0}^{\infty} C\widetilde{L}^{k+1}K(s-\sigma)^k \tag{4.87}$$

The comparison with (4.82) finally gives the expression (4.83). By definition, σ is the so-called *expansion point* by which the Taylor series of the transfer function is expanded. As stated in the literature, the resulting problems for $\sigma = 0$, $\sigma = \infty$ and $0 < \sigma < \infty$ are known as Padé approximation, partial realisation and shifted Padé approximation, respectively. Let us recall that after using the Laplace transform the original problem is considered in the frequency domain. Thus, σ corresponds to the frequencies contained in the original model, so that small values approximate low frequencies and $\sigma \to \infty$ higher frequencies. In most cases the underlying PDEs are characterised by a rather slow dynamic, so approximating the system at the frequency $\sigma \approx 0$ is a natural choice. Otherwise, focusing on fast dynamics results in approximating the system on high frequency range at $\sigma \approx \infty$. In this context, the moments of the transfer function around $\sigma = \infty$ are called *Markov parameters*. More precisely, the Markov parameters are defined as the coefficients of the Taylor series expansion of the transfer function for $\sigma \to \infty$. In an analogous manner as above and using the geometric series it can be shown that the Markov parameters are given by

$$m_k(\infty) = CL^kK \tag{4.88}$$

In particular, the Markov parameters are important to approximate the system's impulse response value with rapidly decaying dynamics at $t = 0$.

4.5.2 Moment Matching

The basic idea of MOR by means of *moment matching* is to approximate the transfer function (4.82), in which the intended reduction focuses on matching the first moments around σ. More precisely, the aim is to find a reduced system (4.2) with transfer function

$$\boldsymbol{H}_r(s) = C_r\left(sI_r - L_r\right)^{-1}K_r = -\sum_{k=0}^{\infty}\widetilde{m}_k(\sigma)(s-\sigma)^k \tag{4.89}$$

whose moments match the first l moments of the original transfer function:

$$m_k(\sigma) = \widetilde{m}_k(\sigma), \quad k = 0, 1, \dots, l-1 \tag{4.90}$$

Obviously, moment matching is by nature a local approach as the transfer function is only approximated in a region around the expansion point. Single-point moment matching (4.90) can be improved by matching of moments at several frequencies (also known as rational interpolation or multi-point moment matching), in particular by expanding the transfer function at multiple points $\{\sigma_1, \ldots, \sigma_k\}$. The main challenges of multi-point moment matching are how to select the expansion points and how many of them are required for an appropriate approximation. Apart from that, it should also be noted that the projection technique based on moment matching implies the loss of the physical interpretability of the original states within the reduced order model. Nevertheless, an approximation of the original state vector can simply be computed by a back projection, i.e. $\boldsymbol{u} \approx V\boldsymbol{u}_r$.

Finally, the question arises how the problem of moment matching can be solved from a numerical point of view. In doing so, the moment matching technique can be done explicitly or implicitly. Let us briefly describe both approaches.

Explicit Moment Matching Explicit approaches to solving the problem of moment matching are based on rational function approximations. It is known that the *Padé approximant* is the best approximation of a given function by a rational function. In other words, the transfer function is approximated by the Padé approximant. Unfortunately, explicit moment matching methods are known to be numerically unstable, especially as the dimension of the reduced order model r grows.

The r-th Padé approximant of $\boldsymbol{H}_r(s)$ that matches the first $l = 2r$ moments of the series expansion of $\boldsymbol{H}(s)$ at $\sigma = 0$ is expressed as

$$\boldsymbol{H}(s) = \boldsymbol{H}_r(s) + \mathcal{O}(s^{2r}) \tag{4.91}$$

In doing so, the transfer functions can be represented as rational functions via

$$\boldsymbol{H}(s) = \frac{B(s)}{A(s)} = \frac{b_{n-1}s^{n-1} + \cdots + b_0}{a_n s^n + \cdots + 1}, \quad \boldsymbol{H}_r(s) = \frac{\widetilde{B}(s)}{\widetilde{A}(s)} = \frac{\widetilde{b}_{r-1}s^{r-1} + \cdots + \widetilde{b}_0}{\widetilde{a}_r s^r + \cdots + 1} \tag{4.92}$$

where $A(s), B(s), \widetilde{A}(s), \widetilde{B}(s)$ are real-valued polynomials. In consequence, the parameters $\widetilde{b}$ and $\widetilde{a}$ are chosen so that the moments $\widetilde{m}_k$ of the reduced order system are equal to those m_k of the original system for $k = 0, \ldots, 2r-1$. To obtain the coefficients for $\widetilde{A}(s)$ and $\widetilde{B}(s)$ from (4.91) and (4.92) one uses the relation

$$\boldsymbol{H}(s) = \boldsymbol{H}_r(s) + \mathcal{O}(s^{2r}) \quad \Longleftrightarrow \quad \sum_{i=0}^{2r} m_i s^i \widetilde{A}(s) = \widetilde{B}(s) + \mathcal{O}(s^{2r})\widetilde{A}(s) \tag{4.93}$$

Consequently, the computation of the parameters requires the solution of two systems of linear equations including Hankel matrices. For example, in order to determine $\widetilde{a}$ it requires to solve for $i = r, \ldots, 2r-1$ the linear system

$$\begin{bmatrix} m_0 & m_1 & \ldots & m_{r-1} \\ m_1 & m_2 & \ldots & m_r \\ \vdots & \vdots & \ddots & \vdots \\ m_{r-1} & m_r & \ldots & m_{2r-2} \end{bmatrix} \begin{bmatrix} \widetilde{a}_r \\ \widetilde{a}_{r-1} \\ \vdots \\ \widetilde{a}_1 \end{bmatrix} = - \begin{bmatrix} m_r \\ m_{r+1} \\ \vdots \\ m_{2r-1} \end{bmatrix} \tag{4.94}$$

with the $2r$ selected moments m_k of the original system being computed explicitly in a preprocess. Another linear system is solved separately for the coefficients $\widetilde{b}$. However, the Hankel matrix contained in (4.94) is ill-conditioned so that the moments are numerically hard to compute. Due to the numerical instability, the explicit moment matching cannot compute accurate moments $\widetilde{m}_k$ as r grows, and therefore results in a less accurate approximation of the reduced transfer function $\boldsymbol{H}_r(s)$.

A numerically stable way of computing the moments is the implicit moment matching based on Krylov subspaces which is usually performed in practice.

Implicit Moment Matching To avoid the numerical instabilities mentioned above, implicit methods match the moments of the transfer functions of the original and reduced model without calculating them explicitly, and compute the projection matrices in which the columns form the bases of particular Krylov subspaces. On this basis, implicit moment matching is highly efficient, numerically robust and achieves a good approximation quality. A compact introduction can be found in [167, 242].

For the sake of simplicity let us consider the SISO system (4.40), so that the system matrices of the reduced order model in state space are given by

$$L_r = \left(W^\top V\right)^{-1} W^\top L V, \quad \boldsymbol{k}_r = \left(W^\top V\right)^{-1} W^\top \boldsymbol{k}, \quad \boldsymbol{c}_r^\top = \boldsymbol{c}^\top V \tag{4.95}$$

The goal of moment matching is to match the first l moments of the original and reduced system in the form (4.90), meaning that the following conditions must be satisfied:

$$\boldsymbol{c}^\top (L - \sigma I)^{-(k+1)} \boldsymbol{k} = \boldsymbol{c}_r^\top (L_r - \sigma I_r)^{-(k+1)} \boldsymbol{k}_r, \quad k = 0, 1, \ldots, l-1 \tag{4.96}$$

It can be shown that the latter is achieved when Krylov subspaces are used for the construction of the projection matrices V and W.

Theorem 4.2 ([11, 109, 242])**.** *Form the columns of the projection matrix $V \in \mathbb{R}^{n \times r}$ used in* (4.95) *a basis of the Krylov subspace $\mathcal{K}_r\left((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\boldsymbol{k}\right)$ around σ, and is $W \in \mathbb{R}^{n \times r}$ arbitrary such that $\det(L_r - \sigma I_r) \neq 0$, then the first $l = r$ moments of the original and the reduced system around σ match.*

Proof. Using (4.95) and (4.96) the first moment of the reduced system reads as

$$\begin{aligned}
\widetilde{m}_0(\sigma) &= \boldsymbol{c}_r^\top \left(L_r - \sigma I_r\right)^{-1} \boldsymbol{k}_r \\
&= \boldsymbol{c}^\top V \left(\left(W^\top V\right)^{-1} W^\top L V - \sigma I_r\right)^{-1} \left(W^\top V\right)^{-1} W^\top \boldsymbol{k} \\
&= \boldsymbol{c}^\top V \left(\left(W^\top V\right)^{-1} \left(W^\top L V - \sigma W^\top V\right)\right)^{-1} \left(W^\top V\right)^{-1} W^\top \boldsymbol{k} \\
&= \boldsymbol{c}^\top V \left(W^\top L V - \sigma W^\top V\right)^{-1} W^\top \boldsymbol{k} \\
&= \boldsymbol{c}^\top V \left(W^\top L V - \sigma W^\top V\right)^{-1} W^\top (L - \sigma I)(L - \sigma I)^{-1} \boldsymbol{k}
\end{aligned} \tag{4.97}$$

Since $(L-\sigma I)^{-1}\boldsymbol{k}$ is the first direction of $\mathcal{K}_r$ and V represents a basis of this Krylov subspace,

the vector can be written as a linear combination of the columns of V via

$$\exists \boldsymbol{r}_0 \in \mathbb{R}^r : (L - \sigma I)^{-1} \boldsymbol{k} = V \boldsymbol{r}_0 \tag{4.98}$$

Thus, (4.97) yields

$$\begin{aligned} \widetilde{m}_0(\sigma) &= \boldsymbol{c}^\top V \left(W^\top L V - \sigma W^\top V \right)^{-1} W^\top (L - \sigma I) V \boldsymbol{r_0} \\ &= \boldsymbol{c}^\top V \left(W^\top L V - \sigma W^\top V \right)^{-1} \left(W^\top L V - \sigma W^\top V \right) \boldsymbol{r_0} \\ &= \boldsymbol{c}^\top V \boldsymbol{r_0} = \boldsymbol{c}^\top \left(L - \sigma I \right)^{-1} \boldsymbol{k} = m_0(\sigma) \end{aligned} \tag{4.99}$$

The evidence of (4.90) for the remaining moments can be done by induction. □

The Krylov subspace $\mathcal{K}_r((L - \sigma I)^{-1}, (L - \sigma I)^{-1} \boldsymbol{k})$ is called the *input Krylov subspace* and based on the fact that W can be chosen optionally, the projection method is called the *one-sided Krylov subspace method*. Another important Krylov subspace that is used for implicit moment matching is the *output Krylov subspace* defined by $\mathcal{K}_r((L - \sigma I)^{-\top}, (L - \sigma I)^{-\top} \boldsymbol{c})$ which can be considered as the dual one to the input Krylov subspace. Due to the duality property, the first r moments can also be achieved using the output Krylov subspace expressed in the following theorem:

Theorem 4.3 ([11,109,242])**.** *Form the columns of the projection matrix $W \in \mathbb{R}^{n \times r}$ used in* (4.95) *a basis of the Krylov subspace $\mathcal{K}_r((L - \sigma I)^{-\top}, (L - \sigma I)^{-\top} \boldsymbol{c})$ around σ, and is $V \in \mathbb{R}^{n \times r}$ arbitrary such that* $\det(L_r - \sigma I_r) \neq 0$*, then the first $l = r$ moments of the original and the reduced system around σ match.*

Proof. The proof follows analogously to the Theorem 4.2 from the duality, by replacing $L, \boldsymbol{k}, V$ with $L^\top, \boldsymbol{c}, W$. □

Obviously, the dual technique is also considered as a one-sided Krylov subspace method. When using the one-sided Krylov subspace method, $W = V$ is a typical choice. Furthermore, the combined use of the input and output Krylov subspaces can double the number of matched moments, while keeping the order r of the reduced system the same, as given by the following theorem:

Theorem 4.4 ([11,109,242])**.** *Form the columns of the projection matrices $V \in \mathbb{R}^{n \times r}$ and $W \in \mathbb{R}^{n \times r}$ used in* (4.95) *a basis of the input and output Krylov subspace, respectively, then the first $l = 2r$ moments of the original and the reduced system around σ match.*

This projection method is called the *two-sided Krylov subspace method*. In total, $2r$ is the maximum number of matching moments, as a transfer function of order r has only $2r$ degrees of freedom. It should be emphasised that two-sided Krylov subspace methods can generate unstable reduced models.

In summary, the number of matching moments depends directly on the method used (one-sided or two-sided). Obviously, a two-sided reduction achieves a better accuracy of the reduced model, as more moments of the transfer function match, whereas a one-sided method can guarantee stability under certain assumptions. The theorems above can easily be extended to match the moments about $\sigma = 0$ or $\sigma = \infty$, and also to ensure multi-point

moment matching. In addition, implicit moment matching can also be applied to MIMO systems by using block Krylov subspaces [242]. When considering MIMO systems, the system matrices of the reduced order model are given by

$$L_r = \left(W^\top V\right)^{-1} W^\top L V, \quad K_r = \left(W^\top V\right)^{-1} W^\top K, \quad C_r = CV \tag{4.100}$$

with $K \in \mathbb{R}^{n\times p}$ and $C \in \mathbb{R}^{q\times n}$. In this regard, the input and output block Krylov subspaces $\mathcal{K}_{r_1}((L-\sigma I)^{-1}, (L-\sigma I)^{-1}K)$ and $\mathcal{K}_{r_2}((L-\sigma I)^{-\top}, (L-\sigma I)^{-\top}C)$, respectively, must be applied. Finally, the previous theorems can be generalised to the MIMO case.

Theorem 4.5 ([242]). *If the matrix V used in* (4.100) *is a basis of the input Krylov subspace $\mathcal{K}_{r_1}$ with rank r (where r is a multiple of p) and the matrix W is chosen such that* $\det(L_r - \sigma I_r) \neq 0$*, then the first $l = \frac{r}{p}$ moments of the original and the reduced system around σ match.*

Theorem 4.6 ([242]). *If the matrices V and W used in* (4.100) *are a bases of the input and output Krylov subspaces $\mathcal{K}_{r_1}$ and $\mathcal{K}_{r_2}$, respectively, both with rank r (where r is a multiple of p and q), then the first $l = \frac{r}{p} + \frac{r}{q}$ moments of the original and the reduced system around σ match.*

As a result, using a one-sided Krylov method will match $\frac{r}{p}$ moments, otherwise a two-sided method match $\frac{r}{p} + \frac{r}{q}$ moments. For MIMO systems, the moments are not scalars and each moment has pq entries. Thus, the number of matching scalar characteristic parameters is $pq\frac{r}{p} = qr$ and $pq(\frac{r}{p} + \frac{r}{q}) = qr + pr$ for the one-sided and the two-sided Krylov method, respectively. Note that the projection matrices V and W must require appropriate dimensions, meaning that the order of the reduced system should be a multiple of the number of inputs and outputs.

In the course of the further work, we denote implicit moment matching as the *Krylov subspace model order reduction (KSMOR)* technique.

4.5.3 Numerical Algorithms

After the theoretical presentation of the KSMOR framework, the question still has to be answered, how the underlying projection can be constructed numerically.

In order to ensure a numerically stable moment matching, the projection matrices V and W have to be computed building on particular Krylov subspaces. However, the explicit computation of the Krylov directions, i.e. $(L-\sigma I)^{-1}\boldsymbol{k}$, must be avoided from a numerical point of view. This can be explained as follows: for convenience only we assume $\sigma = 0$ so that the projection matrix V implies being a basis of the input Krylov subspace $\mathcal{K}_r(L^{-1}, L^{-1}\boldsymbol{k})$. In simple terms, V is constructed via

$$\text{range}(V) = \mathcal{K}_r\left(L^{-1}, L^{-1}\boldsymbol{k}\right) \quad \text{with} \quad \mathcal{K}_r := \text{span}\left(L^{-1}\boldsymbol{k}, \ldots, L^{-r}\boldsymbol{k}\right) \tag{4.101}$$

and gives $V = [L^{-1}\boldsymbol{k}, \ldots, L^{-r}\boldsymbol{k}]$. Obviously, iteratively computing the individual Krylov directions by means of

$$\boldsymbol{v}_1 = L^{-1}\boldsymbol{k}, \quad \boldsymbol{v}_i = L^{-1}\boldsymbol{v}_{i-1}, \quad i = 2, \ldots, r \tag{4.102}$$

leads to a known numerical problem. The successive multiplication with the same matrix yields that the directions $L^{-1}\boldsymbol{v}_{i-1}$ quickly converge towards the eigenvector associated with the dominant eigenvalue of L as i increases. In consequence, the directions $\boldsymbol{v}_i$ become linearly dependent after a few iterations and the projection matrix V is rank deficient.

To overcome the rank deficiency problems mentioned in the construction of the desired Krylov subspace, two main approaches [29, 167] exist which are variations of the Arnoldi and the Lanczos algorithms, and building on the (modified) Gram-Schmidt orthogonalisation process in an iterative manner.

One-Sided Arnoldi Algorithm In one-sided Krylov methods, the most common procedure for constructing the projection is the Arnoldi algorithm. The Arnoldi algorithm computes an orthonormal basis and finds a set of normalised vectors that are orthogonal to each other, more precisely $V^\top V = I_r$. In particular, the iteration rule (4.102) is improved by applying the (modified) Gram-Schmidt method in each iteration. Thus, each newly constructed vector is orthogonal to all the other previous ones and is normalised. An algorithm of the one-sided Krylov subspace method for SISO systems using the input Krylov subspace

$$\operatorname{range}(V) = \mathcal{K}_r\left((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\boldsymbol{k}\right) \tag{4.103}$$

is shown in Figure 4.2. Within the Algorithm 4.1 the inverse $(L-\sigma I)^{-1}$ is required. It should be clear that the inverse will never be computed explicitly, since the problem can be equivalently reformulated as

$$\boldsymbol{v}_i = (L-\sigma I)^{-1}\boldsymbol{v}_{i-1} \quad \Longleftrightarrow \quad (L-\sigma I)\boldsymbol{v}_i = \boldsymbol{v}_{i-1} \tag{4.104}$$

in which a system of linear equations has to be solved in each iteration. For an efficient computation, the underlying matrix $(L-\sigma I)$ is factorised into a product of triangular matrices $(L-\sigma I) = \widetilde{L}\widetilde{U}$ so that the systems for each right-hand side $\boldsymbol{v}_i$ are efficiently solved by forward and backward substitution due to $(L-\sigma I)^{-1} = (\widetilde{L}\widetilde{U})^{-1} = \widetilde{U}^{-1}\widetilde{L}^{-1}$. Consequently, the numerically most expensive part of the one-sided Krylov method is the LU factorisation. For symmetric and positive definite matrices the LU factorisation is replaced by the common Cholesky factorisation.

For the construction of the Krylov subspace V, large and sparse systems of linear equations usually have to be solved. When using sparse direct solvers the factorisation can lead to high computational costs. To avoid this problem, preconditioned sparse iterative solvers can be employed to solve the corresponding linear systems. However, based on the fact that iterative solvers produce approximate solutions, the constructed subspace generated by the Algorithm 4.1 do not longer match with the projection subspace (4.103), i.e.

$$\operatorname{range}(V) \neq \mathcal{K}_r\left((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\boldsymbol{k}\right) \tag{4.105}$$

As a consequence, the moment matching property cannot hold in general. Let us stress that the use of inexact solvers on KSMOR methods of linear dynamical systems was analysed in [26]. More precisely, it was shown that for a well selected expansion point, the iterative KSMOR method can be robust to the perturbations based on the inexact solutions. Otherwise, a poorly selected σ can produce a magnified effect through the model reduction process.

Algorithm 4.1 One-sided Krylov subspace method for SISO systems using input vector

Input: Matrix L; input vector $\boldsymbol{k}$; Krylov subspace order r; expansion point σ

Output: $V = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r]$

1.) Set $i = 1$

a) $\boldsymbol{v}_1 = (L - \sigma I)^{-1}\boldsymbol{k}$

b) $\boldsymbol{v}_1^\star = \frac{\boldsymbol{v}_1}{\|\boldsymbol{v}_1\|}$

2.) Iterate for $i = 2, \ldots, r$

a) $\boldsymbol{v}_i = (L - \sigma I)^{-1}\boldsymbol{v}_{i-1}^\star$

b) Orthogonalise $\boldsymbol{v}_i$ w.r.t. $\{\boldsymbol{v}_1^\star, \ldots, \boldsymbol{v}_{i-1}^\star\}$ using (modified) Gram-Schmidt method

c) $\boldsymbol{v}_i^\star = \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|}$

Figure 4.2: The one-sided Krylov subspace method for constructing the projection matrix V considering SISO systems using the input Krylov subspace. For $\sigma = \infty$, from (4.104) it follows that $\boldsymbol{v}_i = L\boldsymbol{v}_{i-1}$, cf. Markov parameters (4.88). In this case, just sparse matrix-vector multiplications are performed.

The calculation procedure described can be applied analogously for MIMO systems. In doing so, the input Krylov subspace only needs to be replaced by the input block Krylov subspace with the input matrix K. We will discuss the one-sided block Krylov subspace method in more detail in Chapter 6.

Although we are interested in MOR methods that preserve the stability of the original system, a short overview is given about two-sided Krylov subspace methods that match the first $2r$ moments and achieve a more accurate approximation of the transfer function.

Two-Sided Lanczos Algorithm The common two-sided Lanczos algorithm simultaneously creates the projection matrices V and W based on two sequences of basis vectors spanning the input and output Krylov subspaces so that the generated basis vectors are orthogonal to each other, which means $W^\top V = I_r$. On this basis, the classical two-sided Lanczos algorithm is numerically unstable and suffers from the loss of biorthogonality as r increases. Therefore, a reorthogonalisation within the algorithm is proposed to avoid these numerical problems.

Two-Sided Arnoldi Algorithm Another two-sided Krylov subspace method is based on the Arnoldi algorithm presented above, which is numerically stable compared to the Lanczos algorithm. In general, the one-sided Arnoldi process in Algorithm 4.1 is used twice to generate separately the subspaces

$$\begin{aligned} \operatorname{range}(V) &= \mathcal{K}_r\left((L - \sigma I)^{-1}, (L - \sigma I)^{-1}\boldsymbol{k}\right) \\ \operatorname{range}(W) &= \mathcal{K}_r\left((L - \sigma I)^{-\top}, (L - \sigma I)^{-\top}\boldsymbol{c}\right) \end{aligned} \tag{4.106}$$

Thus, each basis is itself orthonormal, i.e. $V^\top V = I_r$ and $W^\top W = I_r$, and by setting $W := W(W^\top V)^{-\top}$ it can also ensure $W^\top V = I_r$. The two-sided Arnoldi algorithm causes higher computational costs, but avoids a possible breakdown within the construction of the projection matrices.

It should be noted that both two-sided Krylov subspace algorithms lead to a reduced system with the same transfer function. In most cases, however, the one-sided Arnoldi process is preferred because, in addition to ensuring numerical stability, the stability of the original system can also be preserved.

In summary, the numerically stable KSMOR method requires a relatively low computational effort and memory storage, which favours this technique for the reduction of large-scale systems. For single-point moment matching only one matrix factorisation of $(L - \sigma I)$ is required, whereby the remaining operations are based on triangular systems that can be solved numerically in a highly efficient manner. As a result, KSMOR usually computes the reduced order model in a fast *offline phase* with a good numerical accuracy, in which the accuracy explicitly depends on the still user-defined parameters.

4.5.4 Computational Aspects

Although the KSMOR methods have been extensively analysed over the past decades to address numerous theoretical and numerical problems, there are still challenges within the process. Let us indicate the most relevant issues.

Error Bounds Apart from the stability preservation, also an a priori knowledge of the error between the original and the reduced model is crucial. Some existing methods, e.g. [20], specify error bounds for the approximation quality under certain conditions. In [28, 300] heuristic error indicators for the reduction of electro-thermal models or dynamical systems with frequency-dependent damping were introduced without a theoretical justification. In order to provide a rigorous error bound describing the qualitative difference between the original and the reduced model, the CURE algorithm [200], which is based on the duality of Krylov subspaces and Sylvester equations, can be applied in an adaptive manner. The CURE framework constructs the reduced order model adaptively and ensures a strictly monotonous decrease in the error norm independently of the choice of the expansion point.

Choice of the Reduced Order Another important aspect in connection with the error bounds is to find a suitable order r for the reduced system. In other words, a procedure is desired that easily specify the approximation quality with respect to the choice of the reduced order. A possible approach to determine the order r is to stop the KSMOR process when no more new linearly independent vectors can be found within the Arnoldi or Lanczos algorithms, see [167]. Another stopping criterion can be achieved by using the above error indicators, where the order r is increased until a user-defined error tolerance is reached. Apart from these heuristic processes, the CURE algorithm can obviously be used which provides an iterative reduction of the error norm and therefore determine an adaptive choice of the reduced order.

Choice of the Expansion Point One of the most important parameters in moment matching methods, in fact, is the choice of the expansion point around which the moments are matched. The value of σ steers the quality of the reduced order model and can also influence the numerical effort if σ is close to an eigenvalue of the underlying system matrix. More precisely, the solution of the linear systems within the construction of the projection matrices can be based on nearly singular matrices.

Typically, slow dynamics are of interest and thus the choice $\sigma = 0$ (or $\sigma \approx 0$) is widely used in a first setup as it often gives good results in a large neighbourhood of the low-frequency part of the spectrum. Otherwise, $\sigma = \infty$ is often chosen if higher frequencies are relevant. For a more automatic selection, the ICOP algorithm [82] is proposed which is a simple and numerically efficient procedure for computing an optimised single expansion point. The optimality properties result from the time domain interpretation of moment matching based on matching the Laguerre coefficients of the impulse response.

If a multi-point moment matching for SISO systems is additionally considered, the number of expansion points selected also has a significant influence. In this situation, the IRKA algorithm [113] is often employed which computes the expansion points in an iterative manner. This technique has also been generalised to the MIMO case. Alternatives for optimising the selection of expansion points are proposed by [55, 200], whereby the latter is an improved version of the IRKA algorithm.

Large Number of Inputs The introduced KSMOR method is a popular tool for large-scale SISO and MIMO systems, assuming that the number of inputs is limited to a relatively small number. This no longer holds to model problems with a high number of inputs, which can be explained as follows.

In general, the problem actually lies in the computation of the projection matrices. For instance, let us consider a one-sided Krylov subspace method and assume that the system having order n with p inputs. For this reason, the input block Krylov subspace has p initial vectors for constructing the Krylov subspace. Consequently, for each additional moment that requires to be matched, further p columns are added to the previous V matrix and the order of the reduced model increases by p. As an example, let $n = 10000$ and $p = 1000$ then matching the first ten moments will result in a reduced model of the same order as the original one. In addition to the intensive computational costs related to the underlying input block Krylov subspace in the offline phase, the online costs for simulating the resulting reduced model increase strongly.

In consequence, the dependence of the reduction efficiency and the number of inputs is directly linked. We will discuss this issue in more detail later in Chapter 6.

Approximation of the Complete Output In many problems the state response at all grid nodes is required, meaning that $C = I$ in (4.1). This is the case, for instance, when the whole temperature field is needed for practical purposes, which is referred to as *single-input-complete-output (SICO)* or *multi-input-complete-output (MICO)* setup. In particular, when using a one-sided Krylov subspace method coupled with an input Krylov subspace the Arnoldi process does not explicitly include the output matrix in the KSMOR process. Consequently, the computation of the reduced order model depends only on the input and the full state output is easily recovered. Otherwise, one-sided (using output Krylov subspace) and two-sided

methods, which require the output matrix for the constructing W, are unsuitable in practice.

Overall, based on these observations the original KSMOR technique cannot be generally used as an automatic method as in the case of MCR and BT. In general, the user has to specify a certain set of parameters (order r, expansion point σ) to which the selected algorithm constructs the projection. The resulting reduced order model is then examined and a fine-tuning of the parameters is performed until satisfactory results are obtained. However, applying the CURE algorithm enables the adaptive selection of these parameters and thus employs an automation at the expense of additional computational costs. Because the procedure is based on an optimisation problem and uses a trust-region method.

4.6 Proper Orthogonal Decomposition

Finally, we introduce the *proper orthogonal decomposition (POD)* method, which is a widespread technique in the MOR community. The original concept was developed 1901 by Pearson [212], but this technique has also been rediscovered many times in the past, so multiple names will be found in the literature such as Karhuenen-Loeve expansion, principal component analysis and Hotelling transformation. The mathematical formulation of POD is also directly related to the statistical analysis of vector data and the covariance matrix, see for instance [147].

The POD method is based solely on data reduction (also known as data-driven reduction), meaning that dominant structures, more specifically an optimal orthonormal basis in the least-squares sense is extracted from a given set of theoretical, experimental or simulation data. Building on this optimal basis the reduced order model is then obtained by truncating. The most common approach for data reduction is the *method of snapshots* introduced by Sirovich [263]. In this approach, the data are created as time snapshots by a numerical simulation of the underlying model problem at certain time instances so that this dataset appropriately reflects the system dynamics. In contrast to the other MOR methods presented in this section, the POD method is based on the time domain formulation of the input-output behaviour of the system.

Based on the extracted data, POD has been employed to build reduced order models that are intensively used in fluid dynamics, control theory, inverse problems as well as signal analysis and pattern recognition, cf. [133, 147, 165, 220] and references therein. The correct choice of the underlying dataset is obviously an essential component within this technique, as the computed POD basis depends on the data. In contrast to other MOR techniques, POD as a data-based method is not limited to a specific model structure and can also easily be used for nonlinear problems as well as for problems with time-dependent coefficients. For this reason, this technique is considered as a powerful and universal tool for large-scale dynamical systems in many fields of science and engineering. Besides these positive features, it should be noted that POD does not guarantee stability preservation [221, 233], even if the original model is stable. However, in fact, POD belongs to the class of one-sided projection methods and thus preserves the stability (cf. Lemma 4.1), independent of the underlying data, if the system matrix is negative semi-definite [221]. In this case, the data only affect the quality of the resulting reduced model.

In general, the POD method is often applied to nonlinear dynamical systems in practice because of its ease of use and the simultaneous competitive approximation quality of the

reduced model. The successful use of POD for practical applications in connection with (parabolic) PDEs can be found in [5, 12, 33, 48, 85, 195]. Let us describe the POD approach in more detail.

4.6.1 Proper Orthogonal Decomposition Method

As mentioned above, the basic idea of POD is to use the system's time responses given by a certain input, that contains the essential behaviour of the system, and to provide a basis that optimally represents the given data in the least squares sense. Assuming that the dynamical system is measured s times at different time instances with $s \ll n$, where each measurement $\boldsymbol{u}_k$ for $k = 1, \ldots, s$ being a large vector of real entries, i.e. $\boldsymbol{u}_k \in \mathbb{R}^n$. The measurements, also called *snapshots*, are stored in a matrix

$$U = [\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots, \boldsymbol{u}_s] \in \mathbb{R}^{n \times s} \tag{4.107}$$

To obtain a reduced order model, the application of a projection can be formulated mathematically as an optimisation problem or, to be more precise, as a matrix optimisation problem of the form

$$\min_{\operatorname{rank}(P_r)=r} \|U - P_r U\|_2^2 \quad \text{s.t.} \quad P_r^2 = P_r \in \mathbb{R}^{n \times n} \tag{4.108}$$

where $P_r = \Phi_r \Phi_r^\top, \Phi \in \mathbb{R}^{n \times r}$ and $\Phi_r^\top \Phi_r = I_r$. Thus, the given data is used to find an orthogonal projection P of fixed (best) rank r to the data in the L_2-norm.

Let $\mathcal{U}$ be a separable Hilbert space with inner product $\langle \cdot, \cdot \rangle_2$ and orthonormal basis $\{\phi\}_{i \in I}$. Then, any element $u(\boldsymbol{x}, t) \in \mathcal{U}$ take the form

$$u(\boldsymbol{x}, t) = \sum_i a_i(t) \phi_i(\boldsymbol{x}) = \sum_i \langle u(\boldsymbol{x}, t), \phi_i(\boldsymbol{x}) \rangle_2 \, \phi_i(\boldsymbol{x}) \tag{4.109}$$

with time-dependent Fourier coefficients a_i. With this formulation, the matrix optimisation problem (4.108) can be reformulated into

$$\min_{\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_r} \sum_{j=1}^{s} \left\| \boldsymbol{u}_j - \sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2 \, \boldsymbol{\phi}_i \right\|_2^2 \quad \text{s.t.} \quad \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle_2 = \delta_{i,j} \tag{4.110}$$

with Kronecker delta $\delta_{i,j}$ and $r < s$. The orthogonality of the basis implies for all i, j that

$$\begin{aligned} 0 &\leq \left\| \boldsymbol{u}_j - \sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2 \, \boldsymbol{\phi}_i \right\|_2^2 \\ &= \left(\boldsymbol{u}_j - \sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2 \, \boldsymbol{\phi}_i \right)^\top \left(\boldsymbol{u}_j - \sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2 \, \boldsymbol{\phi}_i \right) \\ &= \boldsymbol{u}_j^\top \boldsymbol{u}_j - 2 \sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2^2 + \sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2^2 \\ &= \|\boldsymbol{u}_j\|_2^2 - \sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2^2 \end{aligned} \tag{4.111}$$

Consequently, the problem (4.110) is equivalent to

$$\max_{\boldsymbol{\phi}_1,\dots,\boldsymbol{\phi}_r} \sum_{j=1}^{s}\sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i\rangle_2^2 \quad \text{s.t.} \quad \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j\rangle_2 = \delta_{i,j} \tag{4.112}$$

Note that (4.112) is an equality constrained optimisation problem and can be solved by the method of *Lagrange multipliers*. The corresponding Lagrangian function of (4.112) reads

$$\begin{aligned} &\mathcal{L}(\boldsymbol{\phi}_1,\dots,\boldsymbol{\phi}_r,\Lambda) = \sum_{j=1}^{s}\sum_{i=1}^{r} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_i\rangle_2^2 + \sum_{i,j=1}^{r} \lambda_{i,j}\left(\delta_{i,j} - \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j\rangle_2\right) \\ &\mathcal{L} : \underbrace{\mathbb{R}^n \times \cdots \times \mathbb{R}^n}_{r\text{-times}} \times \mathbb{R}^{r\times r} \end{aligned} \tag{4.113}$$

with $\boldsymbol{\phi}_1,\dots,\boldsymbol{\phi}_r \in \mathbb{R}^n$ and multipliers $\Lambda = (\lambda_{i,j}) \in \mathbb{R}^{r\times r}$. The necessary first order optimality condition, for which the convex objective function and the constraints are continuously differentiable, is then given by the gradient of the Lagrangian $\mathcal{L}$ via

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_k}(\boldsymbol{\phi}_1,\dots,\boldsymbol{\phi}_r,\Lambda) = \mathbf{0} \quad (k \in \{1,\dots,r\}), \quad \nabla_\Lambda \mathcal{L} = \mathbf{0} \tag{4.114}$$

The partial derivatives of $\mathcal{L}$ can be expressed as

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{\phi}_k}(\boldsymbol{\phi}_1,\dots,\boldsymbol{\phi}_r,\Lambda) = \sum_{j=1}^{s} 2\,\langle \boldsymbol{u}_j, \boldsymbol{\phi}_k\rangle_2\, \boldsymbol{u}_j - \sum_{i=1}^{r}(\lambda_{i,k} + \lambda_{k,i})\boldsymbol{\phi}_i \tag{4.115}$$

From (4.114) and (4.115) it holds that

$$\sum_{j=1}^{s} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_k\rangle_2\, \boldsymbol{u}_j = \frac{1}{2}\sum_{i=1}^{r}(\lambda_{i,k} + \lambda_{k,i})\boldsymbol{\phi}_i, \quad k \in \{1,\dots,r\} \tag{4.116}$$

The left-hand side of the latter equation can be rewritten by means of

$$\sum_{j=1}^{s} \langle \boldsymbol{u}_j, \boldsymbol{\phi}_k\rangle_2\, \boldsymbol{u}_j = UU^\top \boldsymbol{\phi}_k \tag{4.117}$$

Using induction implies: for $r = 1$ one has $k = 1$ and with (4.116) it follows that

$$UU^\top \boldsymbol{\phi}_1 = \lambda_1 \boldsymbol{\phi}_1 \tag{4.118}$$

with $\lambda_1 = \lambda_{1,1}$. Supposing now, for $r \geq 1$ the first order optimality conditions are given by

$$UU^\top \boldsymbol{\phi}_k = \lambda_k \boldsymbol{\phi}_k, \quad k \in \{1,\dots,r\} \tag{4.119}$$

so that one has to show that for a basis $\{\boldsymbol{\phi}_i\}_{i=1}^{r+1}$ of rank $r+1$ the conditions

$$UU^\top \boldsymbol{\phi}_k = \lambda_k \boldsymbol{\phi}_k, \quad k \in \{1,\dots,r+1\} \tag{4.120}$$

hold. For this reason it must be shown:

$$UU^\top \boldsymbol{\phi}_{r+1} = \lambda_{r+1}\boldsymbol{\phi}_{r+1} \tag{4.121}$$

With (4.116) it can be obtained first

$$UU^\top \boldsymbol{\phi}_{r+1} = \frac{1}{2}\sum_{i=1}^{r+1}(\lambda_{i,r+1} + \lambda_{r+1,i})\boldsymbol{\phi}_i \tag{4.122}$$

Using the orthogonality conditions $\langle \boldsymbol{\phi}_{r+1}, \boldsymbol{\phi}_j \rangle_2 = 0$, the symmetry of $UU^\top$ and (4.119), it holds for any $j \in \{1, \ldots, r\}$:

$$\begin{aligned} 0 &= \lambda_j \langle \boldsymbol{\phi}_{r+1}, \boldsymbol{\phi}_j \rangle_2 = \langle \boldsymbol{\phi}_{r+1}, UU^\top \boldsymbol{\phi}_j \rangle_2 = \langle UU^\top \boldsymbol{\phi}_{r+1}, \boldsymbol{\phi}_j \rangle_2 \\ &= \frac{1}{2}\sum_{i=1}^{r+1}(\lambda_{i,r+1} + \lambda_{r+1,i})\langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle_2 = (\lambda_{j,r+1} + \lambda_{r+1,j}) \end{aligned} \tag{4.123}$$

This implies then

$$\lambda_{r+1,i} = -\lambda_{i,r+1}, \quad i \in \{1, \ldots, r\} \tag{4.124}$$

Inserting (4.124) into (4.122) yields

$$\begin{aligned} UU^\top \boldsymbol{\phi}_{r+1} &= \frac{1}{2}\sum_{i=1}^{r}(\lambda_{i,r+1} + \lambda_{r+1,i})\boldsymbol{\phi}_i + \lambda_{r+1,r+1}\boldsymbol{\phi}_{r+1} \\ &= \frac{1}{2}\sum_{i=1}^{r}(\lambda_{i,r+1} - \lambda_{i,r+1})\boldsymbol{\phi}_i + \lambda_{r+1,r+1}\boldsymbol{\phi}_{r+1} = \lambda_{r+1,r+1}\boldsymbol{\phi}_{r+1} \end{aligned} \tag{4.125}$$

and setting $\lambda_{r+1,r+1} = \lambda_{r+1}$ results in (4.121). In summary, the necessary optimality conditions for (4.112) are given by the symmetric $n \times n$ eigenvalue problem

$$UU^\top \boldsymbol{\phi}_i = \lambda_i \boldsymbol{\phi}_i, \quad i = 1, \ldots, r \tag{4.126}$$

The functions $\boldsymbol{\phi}_i$ are called POD *modes* (or POD *basis*) and λ_i are the POD *eigenvalues*. Moreover, the matrix $UU^\top$ is positive semi-definite and closely related to the covariance matrix from the field of statistics. Finally, it can be shown that the optimality is given by

$$\sum_{j=1}^{s}\left\| \boldsymbol{u}_j - \sum_{i=1}^{r}\langle \boldsymbol{u}_j, \boldsymbol{\phi}_i \rangle_2 \boldsymbol{\phi}_i \right\|_2^2 = \lambda_{r+1} \tag{4.127}$$

Selection of the Dimension The question arises how many basis functions are required in order to achieve a good approximation of the given dataset. In most cases, not all of the basis POD modes are necessary to capture the main characteristics of the dataset. The condition (4.127) forms the basis for the selection of the dimension, meaning that the eigenvalues of $UU^\top$ give a measure of the relevance. Thus, large eigenvalues correspond to the main characteristics of the system, whereas small eigenvalues have less contributions to the system dynamics. In this context, the number of dimension r is often based on an energy criterion

in the format

$$\mathcal{E}(r) := \frac{\sum_{i=1}^{r} \lambda_i}{\sum_{i=1}^{s} \lambda_i} \tag{4.128}$$

The employed tolerance $\mathcal{E}_{\min}$ with $\mathcal{E}(r) \geq \mathcal{E}_{\min}$ is depending on the model problem. Obviously, for $r \ll s$ the eigenvalues should decrease sufficiently fast. In fact, in many applications such as heat transfer one often observes an exponential decrease in eigenvalues, so a low order model may possibly be sufficient to represent the original model.

4.6.2 Method of Snapshots

Unfortunately, the basic POD concept requires an eigendecomposition of the underlying $n \times n$ matrix $UU^\top$. To overcome this computationally hard problem the method of snapshots proposed by Sirovich [263] is often used in practice. The general idea is based on the SVD of the data U, so let

$$U = \Phi \Sigma \Psi^\top \tag{4.129}$$

with orthogonal matrices Φ, Ψ satisfying

$$U\boldsymbol{\psi}_i = \sigma_i \boldsymbol{\phi}_i, \quad U^\top \boldsymbol{\phi}_i = \sigma_i \boldsymbol{\psi}_i, \quad i = 1, \ldots, \operatorname{rank}(U) \tag{4.130}$$

Furthermore, it holds that

$$\begin{aligned} UU^\top &= \Phi \Lambda \Phi^\top \in \mathbb{R}^{n \times n} \\ U^\top U &= \Psi \Lambda \Psi^\top \in \mathbb{R}^{s \times s} \end{aligned} \tag{4.131}$$

where $\Lambda = \Sigma^2$. More precisely, Ψ and Φ contains the eigenvectors of $U^\top U$ and $UU^\top$, respectively. From (4.130) it follows that $U\Psi = \Sigma\Phi = \Lambda^{\frac{1}{2}}\Phi$ so that

$$\Phi = \Lambda^{-\frac{1}{2}} U \Psi \tag{4.132}$$

and therefore $\Phi = [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_s]$ contains the desired POD modes. In other words, solving the symmetric $s \times s$ eigenvalue problem

$$U^\top U \boldsymbol{\psi}_i = \lambda_i \boldsymbol{\psi}_i, \quad i = 1, \ldots, s \tag{4.133}$$

gives the same eigenvalues as $UU^\top$, so the corresponding POD modes are given by

$$\boldsymbol{\phi}_i = \frac{1}{\sqrt{\lambda_i}} U \boldsymbol{\psi}_i, \quad i = 1, \ldots, s \tag{4.134}$$

Model Order Reduction by POD In the following we specify the POD algorithm. To obtain a reduced order model by this projection technique, five steps are necessary:

1.) Form the snapshot matrix $U = [\boldsymbol{u}(t_1), \ldots, \boldsymbol{u}(t_s)] = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_s] \in \mathbb{R}^{n \times s}$ with $s \ll n$ and $\operatorname{rank}(U) = k, k \leq s$.

2.) Solve the eigenvalue problem

$$U^\top U\boldsymbol{\psi}_i = \lambda_i \boldsymbol{\psi}_i, \quad i = 1, \ldots, k \tag{4.135}$$

and compute the corresponding POD modes

$$\boldsymbol{\phi}_i = \frac{1}{\sqrt{\lambda_i}} U\boldsymbol{\psi}_i, \quad i = 1, \ldots, k \tag{4.136}$$

with eigenvector matrix $\Phi_k \in \mathbb{R}^{n\times k}$ and $\Phi_k^\top \Phi_k = I_k$.

3.) Set a threshold, e.g. using the measurement (4.128), to pick the r highest eigenvalues.

4.) Assemble the projection matrix V that corresponds to the r modes selected in step 3 via $V := [\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_r] \in \mathbb{R}^{n\times r}$.

5.) Approximation of the state space vector $\boldsymbol{u}(t)$ by the reduced POD basis $\boldsymbol{u}(t) \approx V\boldsymbol{u}_r(t)$. Using this reduced representation and projecting along the subspace generated by V results in the reduced model.

For example, if a general dynamical system of the form $\dot{\boldsymbol{u}}(t) = \boldsymbol{f}\left(t, \boldsymbol{u}(t)\right)$ is given, the reduced model reads

$$\dot{\boldsymbol{u}}_r(t) = V^\top \boldsymbol{f}\left(t, V\boldsymbol{u}_r(t)\right) \tag{4.137}$$

The snapshot matrix in step 1.) of the algorithm is usually formed by the simulation of the original model using a suitable numerical solver. Of course, the selection of the snapshots is very important as the POD basis depends on the initial condition and the input.

Remarks The POD concept building on data reduction makes this technique highly flexible so that this MOR method can also be applied to nonlinear dynamical systems. In addition, using the method of snapshots the dominant POD modes are computed in a relatively easy way. However, as described above, the POD basis is constructed directly from the system responses. This highlights the main weakness of the technique as the main retained characteristics contained in the POD basis are signal-dependent by nature. The reduced model therefore only has a good approximation quality compared to the original system if the input is close to the model input. In other words, the computed POD modes do not provide a general physical interpretation compared to the MCR modes, but they provide a good characterisation of the dynamics. The efficient selection of the number and the sample of snapshots is still a research subject. Obviously, the snapshots should be captured precisely when the dynamics of the system is change. Otherwise, it should be mentioned that the POD technique aims to describe the given data using the same global features. This means that the characteristics of a large dataset often vary in space and the use of local features can be beneficial to represent different system behaviours of the given data set. The latter observations indicate that POD cannot be used automatically.

For further details from a theoretical and practical point of view, we refer to e.g. [190, 276]. Let us note that there are other types of POD methods such as frequency-domain POD or POD in combination with BT. Apart from that, POD is often successfully applied in combination with the Galerkin projection [155, 220] which is an analytical-based method.

4.7 Summary of Linear Model Order Reduction Methods

In summary, we have given a detailed description of the most important methods for linear MOR. Within the scope of this thesis the methods MCR and KSMOR are of interest. This can be explained as follows: first, the model problems dealt with in this work contain large and sparse dynamical systems with more than $\mathcal{O}(10^5) - \mathcal{O}(10^6)$ variables, these problem sizes often arising in the field of image processing, computer vision or engineering problems. Second, the model problems have to be solved only once with the underlying setting, meaning that the reduced order model is computed in an *online-based*[2] process. This eliminates the need for intensive *offline-based* computations that often arise when problems related to parameter estimation, uncertainty quantification, optimisation and control are must be repeatedly solved.

In particular, BT fails to be an efficient MOR method for the underlying model problems considered here, since the core of this technique is linked to the solution of computationally intensive Lyapunov equations. Furthermore, the POD technique is also not relevant in our setting as training snapshots are first needed to build the reduced order model. This approach therefore contradicts the MOR concept to a certain extent, since the large and unreduced system must first be solved in order to reduce exactly this system. Solving the large original model problem and constructing the projection cause high computational costs and is in consequence exactly what we want to avoid.

The use of MCR naturally requires to perform an eigenvalue decomposition and can lead to high computational costs, especially if the approximation quality of the reduced model involves retaining a large number of MCR modes. Nevertheless, this technique can be efficient for model problems with many different initial conditions along with a smaller number of dominant eigenvalues that are required to properly represent the main characteristics of the underlying dynamical system. In this case, only one eigendecomposition is necessary. In contrast, all other MOR methods presented have to recomputed the reduced order model for new system parameters such as initial conditions or input vectors.

For linear model problems with a small number of initial conditions or inputs, KSMOR is in general the most efficient method nowadays as it is based on efficient numerical linear algebra techniques, namely Krylov subspaces, and often only requires a small dimension of the resulting reduced order model.

4.8 Outlook of Model Order Reduction

In the previous sections we introduced and discussed the state-of-the-art MOR methods with an emphasis on linear time-invariant dynamical systems. Since MOR is a very active research area due to the industrial need, there exist alternative methods. For example, the class of nonprojective MOR methods including vector fitting methods, see e.g. [192]. Besides the linear time-invariant systems dealt with this thesis, MOR methods are also conceptually applied to different classes of model problems. Therefore, a brief overview of the use of MOR in relation to various model problems that often arise in practical applications is given below.

[2] In principle, the statement "online-based" corresponds to "an extremely fast offline computation" and should emphasise that we want to avoid time-consuming MOR methods, although these may achieve a more accurate approximation quality of the reduced order model.

Second Order Dynamical Systems In practical applications such as electrical, mechanical and structural dynamics, second order dynamical systems are often of interest. In general, a time-invariant second order MICO system is described by

$$\begin{aligned} M\ddot{\boldsymbol{u}}(t) + D\dot{\boldsymbol{u}}(t) + K\boldsymbol{u}(t) &= P\boldsymbol{w}(t) \\ \boldsymbol{y}(t) &= E\boldsymbol{u}(t) \end{aligned} \tag{4.138}$$

with system matrices $M, D, K \in \mathbb{R}^{n\times n}$, input matrix $P \in \mathbb{R}^{n\times p}$, output matrix $E \in \mathbb{R}^{n\times q}$ and initial conditions $\boldsymbol{u}(0) = \boldsymbol{u}^0$ as well as $\dot{\boldsymbol{u}}(0) = \widetilde{\boldsymbol{u}}^0$. To deal with such problems using MOR methods, the second order system can be reformulated into an equivalent linear first order system of the form

$$\begin{aligned} C\dot{\boldsymbol{x}}(t) + G\boldsymbol{x}(t) &= B\boldsymbol{w}(t) \\ \boldsymbol{y}(t) &= L\boldsymbol{x}(t) \end{aligned} \tag{4.139}$$

with the components

$$\boldsymbol{x}(t) = \begin{bmatrix} \boldsymbol{u}(t) \\ \dot{\boldsymbol{u}}(t) \end{bmatrix}, \quad C = \begin{bmatrix} D & M \\ W & 0 \end{bmatrix}, \quad G = \begin{bmatrix} K & 0 \\ 0 & -W \end{bmatrix}, \quad B = \begin{bmatrix} P \\ 0 \end{bmatrix}, \quad L = \begin{bmatrix} E \\ 0 \end{bmatrix} \tag{4.140}$$

where $W \in \mathbb{R}^{n\times n}$ is a suitable nonsingular matrix. Afterwards, any of the MOR techniques mentioned can be applied to the *linearised* system (4.139), see e.g. [19]. Obviously, the linearised system differs from the original system structure (4.1) by the matrix C in front of $\dot{\boldsymbol{x}}(t)$, but this is not a limitation with regard to the MOR techniques introduced. If the matrix C is regular, the original structure is obtained by multiplying from the left with C^{-1} to (4.139). Another option is the direct reduction of the second order system (4.138), whereby the definiteness properties of the matrices M, D, K can be more easily preserved.

Parametric Dynamical Systems Another research field focuses on the problem class in which the system dynamics depend on a parameter set $\boldsymbol{p} = [p_1, \ldots, p_d]^\top$ of the form

$$\begin{aligned} \dot{\boldsymbol{u}}(t) &= L(\boldsymbol{p})\boldsymbol{u}(t) + K(\boldsymbol{p})\boldsymbol{w}(t) \\ \boldsymbol{y}(t) &= C(\boldsymbol{p})\boldsymbol{u}(t) \end{aligned} \tag{4.141}$$

with the parameters involved representing, for example, material properties, system geometry or system configuration. The problem class of parameter-varying systems is also known as *parametric MOR*. The parametric dependence represents different new challenges for MOR, whereby the presented techniques cannot simply be applied. In particular, the reduced model is also parameter dependent, which means that the parametric dependence must be introduced into the projection matrices V and W. Consequently, the main task of parametric MOR is to preserve the parameter dependence within the reduced model, so that the variation of the parameters should not be accompanied by a recomputation of the reduction process. A detailed survey about projection-based parametric MOR can be found e.g. in [39].

Closely related to parametric dynamical systems are linear time-varying systems, in which the system matrices are time-dependent. A possible application purpose is, for example, moving loads in which the position of the input (load) can vary over the considered time horizon. More details can be found e.g. in [66, 256].

Nonlinear Dynamical Systems Apart from the model reduction of time-variant or parameter-dependent systems, many scientific problems contain nonlinearities so that a natural consequence is to deal with MOR of nonlinear dynamical systems. Numerical simulations of complex nonlinear problems usually cause high computational costs due to large spatial and temporal grids within the discretisation process. Therefore, reduced order models are generally indispensable in practice. Although linear MOR techniques have been extensively studied in the past, the extension of these concepts to nonlinear problems in general is not successfully applicable. This results directly from the complexity and diversity of nonlinear systems compared to the clearly defined structure of linear systems. Because of this, MOR is still a major challenge for nonlinear problems, and extensive research for establishing a formal concept is being done. Nevertheless, several methods have been developed to handle nonlinear dynamical systems. The main approaches are linearisation and quadratic methods, piecewise linear trajectory-based MOR, empirical Gramians methods and POD techniques.

The linearisation methods are the simplest approach for the model reduction of nonlinear dynamical systems. The main strategy is based on a linearisation of the state space system around an operating point, so that any linear MOR technique can be applied to obtain a reduced order model. The linearisation of the nonlinear functions uses Taylor series expansion which is truncated after the first order (linear) term. As a result, the Jacobian of the nonlinear system is linearised and only linear effects are considered within the new system. The main disadvantage of this reduction technique is that the approximation quality of the reduced model is only accurate for the system behaviour close to this operating point.

The quadratic methods are an improvement over the linearisation methods. The basic difference to the previous approach is that also the second order (quadratic) term of the Taylor expansion is retained. This aims to improve the accuracy of the reduced representation and should extend its validity to a larger state space. Although the quadratic method is more precise than its linearised counterpart, both approaches only build a good approximative reduced model in the neighbourhood of the operating point. Thus, the reduced models are only locally accurate and are practically useful for weakly nonlinear dynamical systems.

To overcome this weak nonlinearity limitation, a modified approach called the piecewise linear trajectory-based MOR technique can be used. The central idea of this class of methods is to use a set of operating points such that a globally accurate reduced model is achieved. More precisely, the linear model reduction consists of local linear submodels, in which the projection matrix is generated by individual projection bases for each local submodel. The final reduced model is then obtained via the weighted combination of all the reduced models. In particular, any MOR technique can be applied to the linear submodels.

Another approach is the empirical Gramians method (empirical BT) which is an extension of BT to nonlinear problems. The technique is solely based on balancing the nonlinearities and uses the (discrete) empirical Gramians. In this setting, empirical Gramians are a type of a covariance matrix and provide the input-output behaviour of the nonlinear dynamical system. The construction of empirical Gramians built on the averaging over local Gramians for any varying quantity (input, initial condition, parameter) around an operating point, see for example [132, 156].

The most popular MOR method for nonlinear dynamical systems is the POD method and their variants. Despite the drawbacks mentioned in Section 4.6, model reduction via POD is currently state-of-the-art for many nonlinear problems due to its simple implementation and promising accuracy.

Let us emphasise that the evaluation of the nonlinear terms within the reduction process is also important from a computational point of view. This means that during the MOR process the nonlinear term still has to be evaluated in the original dimension. In each iteration, the low-dimensional solution must be projected back onto the full state space and vice versa. Both processes are computationally intensive and a speed-up of the computation of the nonlinearities is desirable. A common method for dealing with such problems is the discrete empirical interpolation method introduced by Chaturantabut [60], in which the nonlinear function is cleverly interpolated.

For a deeper insight into MOR of nonlinear dynamical systems we refer the reader to [25, 91, 110, 133, 192] and the references therein. Note that there are also simulation-free methods [236] for computing a reduced order model in the nonlinear regime.

Surrogate Modelling and Artificial Intelligence Numerical simulations of very complex systems arising from the solution of the recent society problems in modern sciences require extremely expensive computational costs, especially when simulations need to be repeated frequently due to changes of initial values and parameters e.g. in applications of design, control, optimisation and uncertainty quantification. In this situation, high offline costs are often accepted in order to obtain a reduced order model that enables fast, yet accurate simulation results in the online phase. At present, the field of *artificial intelligence* is one of the most exciting developments. A more recent aspect are therefore MOR techniques building on artificial intelligence, the synonym MOR usually being referred to as *surrogate modelling*.

Surrogate modelling strategies for reducing the computational burden are generally constructed using a *data-driven* approach. Data-driven methods substitute conceptually expensive numerical simulations with a surrogate model so that the input-output mapping of a specific numerical simulation is basically approximated by a black box. The black box is often built by interpolation or regression of simulation data based on e.g. Gaussian processes or Kriging methods coming from statistics. Kriging methods have been successfully used as surrogate models, but such techniques only use scalar information from the extensive simulations. In consequence, the vast amount of information generated by these simulations remains unused for the statistical surrogate models. In order to overcome these problems, new data-driven surrogate models are being developed that are based on artificial neural networks and can also use physical information, which are also referred to as physics-informed neural networks.

Besides the advantages such as versatility, low evaluation costs and a large number of modelling techniques, where the implementations are open source (ready-to-use), a substantial advantage of neural networks is that the offline process of deriving the surrogate model is handled relatively nonintrusive as only a collection of input-output data is needed. In contrast, statistical methods typically lead to a loss of flexibility in their surrogate model, since the models may only be applicable to the specific conditions under which their was derived. However, neural networks are numerically expensive to train, and the acquisition and storage of the training data normally incur significant computational costs. Recent works also deal with online learning and transfer learning. The former enables parallel training to a running simulation, while the latter uses functional similarities of different systems to reduce the data and time required to train accurate surrogate models. For an overview about data-driven surrogate modelling we refer e.g. to [153]. We also refer to some recent techniques based on MOR coupled learning [122, 241, 291] or PDE learning [121, 168, 223–225, 246, 262].

Chapter 5

Efficient Descriptor-Based Shape Analysis

In this chapter we are interested in the highly efficient computation of shape descriptors proposed in [67, 68], which are based on time integration methods and whose intrinsic shape signatures are useful for shape analysis purposes. The numerical signatures obtained by time integration methods of the underlying PDEs lead to significant improvements over state-of-the-art-methods for finding correct shape correspondences. However, its computation by solving a large system of linear equations for a huge number of right-hand sides is linked to high computational costs and make it generally impractical for high resolution shapes. Therefore, it is absolutely essential to find a fast and accurate numerical scheme within this framework. To this end, we analyse and evaluate direct, iterative and MOR methods and their influence to shape correspondence applications which are validated on standard shape datasets with different resolutions.

We will identify that MOR methods, more precisely KSMOR and MCR, provide simple and efficient time integrators with an equally accurate shape matching performance compared to the original works [67, 68]. For this reason, we show in more detail how to define a computational framework by MOR which can be distinguished as spectrum-free and spectrum-based computation by KSMOR and MCR, respectively. Furthermore, within the construction of our framework we elaborate several substantial details of the MCR technique which are necessary in order to enhance the usability. In this context, we will also provide a detailed description of the differences and similarities to the methods which are based on the analytical solutions of the underlying geometric PDEs.

Our conducted experiments will also show that spectral decomposition methods are beneficial for high resolution shapes. Although solving an eigenvalue problem is considered to be quite expensive, in general only a small number of eigenvalues and eigenvectors are required for many shape correspondence purposes, so that the corresponding computational costs are relatively low. The latter statement is identified by evaluating at hand of the complete TOSCA dataset and is an important practical topic for the methods discussed with regard to *pointwise shape correspondence (PSC)*. In addition, we demonstrate that the MCR technique presented here, which is linked to the IE scheme, is superior in the class of spectrum-based methods due to a much higher matching accuracy.

Apart from that, we will introduce the soft correspondence map and the mapping indicator function for detecting specific geometric regions. On this basis, we show that the KSMOR method can achieve a high correspondence quality, while the spectrum-based approaches produce a less accurate correspondence quality. In this context, we also demonstrate that the MCR technique outperforms their direct counterparts, the *heat kernel signature (HKS)* and the *wave kernel signature (WKS)*.

We note that a part of this chapter was already presented in our work [18]. Employing the MCR technique in its raw version introduced in [18] for the shape matching purpose have shown some promising first results. However, we introduce novel aspects and techniques compared to the preliminary examinations. In particular, the extended experimental evaluations allow to highlight some relevant properties of the MOR signatures.

Chapter Organisation After an overview on related work in Section 5.1, we briefly recall the general framework we rely on along with the arising PDEs and the numerical discretisations employed in space and time in the Sections 5.2 and 5.3, respectively. As our work relies very much on related numerical techniques, we give in Section 5.4 a short discussion of numerical solvers with an emphasis on solvers for systems of linear equations that arise. We also give a detailed exposition of MOR methods and some related issues is given in our setting. Our first experimental evaluation presented in Section 5.5 focuses on the evaluation of all introduced numerical solvers. This is followed by a discussion of the numerical similarity transform, time rescaling and modification of the initial condition, we propose here for optimisation the MCR technique in Section 5.6. Beginning with Section 5.7 the kernel-based methods are included in our discussion. In Section 5.8 we give a detailed comparison of the spectrum-based approaches focusing thereby on the most promising MOR methods, identified in the previous sections, and the kernel-based methods HKS and WKS. The chapter is finished by a summary.

5.1 Introduction

In computer graphics the Laplace-Beltrami operator, which is connected to geometric PDEs such as the heat, wave or Schrödinger equation, is successfully used in several applications in the field of shape parameterisation, deformation, compression, segmentation, comparison and analysis. Geometric PDEs are characterised by the fact that they take into account geometric surface information, although the geometry does not change during time evolution. A main task within shape analysis is called shape matching where it is often important to decrypt information about the relation between three-dimensional objects. The investigation of correspondences between three-dimensional shapes is a fundamental problem and has a wide variety of potential applications, including e.g. shape comparison or texture transfer, see e.g. [282] for some discussion. The basic task, cf. Figure 5.1, of finding shape correspondences is to identify a relation between elements of two or more shapes, where a nonnegligible challenging setting for this is concerned with nonrigid shapes that are assumed to be just almost isometric, compare for instance [50]. One of the possible strategies to find pointwise correspondences is to construct a feature descriptor, or shape signature, which characterises geometry around the points that define the surface of a given shape. Moreover, a suitable feature descriptor is required which is invariant under almost isometric transformations. An interesting class of models for such descriptors is based on the Laplace-Beltrami operator which enables to describe intrinsic geometric properties of a shapes' surface [162, 226, 235]. To this end, several PDEs have been proposed [14, 67, 267] for the construction of shape signatures that rely on the Laplace-Beltrami operator as a crucial component. In order to conduct the construction, not only a variety of PDEs but also several ways to solve them have been considered in previous works [14, 18, 67, 267, 285] or in related applications [208, 209, 301]. All approaches are based on time-evolution processes, so let us give a short overview.

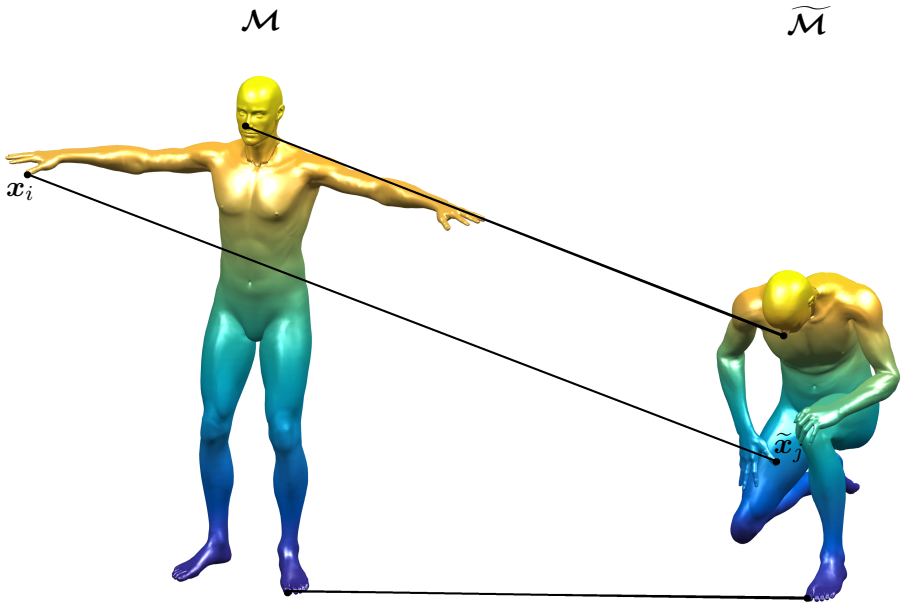


Figure 5.1: The fundamentalPSC task. Find the pointwise correspondence between the given two shapes $\mathcal{M}$ and $\widetilde{\mathcal{M}}$. More precisely, find a pointwise map S which matches points $\boldsymbol{x}_i \in \mathcal{M}$ on $\widetilde{\boldsymbol{x}}_j \in \widetilde{\mathcal{M}}$. In this work we use a simplified representation, the so-called pointwise feature descriptor.

Time-Evolution Methods A popular trend in shape analysis consists of exploiting intrinsic time-evolution processes carried by PDEs on geometric shapes. In this framework, diffusion processes are well established, allowing a meaningful interpretation relating the propagation of information and intrinsic distances. For example, the propagation of heat on a shape can be interpreted as a random walk among surface points [49, 62].

In the spirit of this framework, [267] introduced the HKS based on the heat equation. The HKS describes the amount of heat that remains at a certain point after a certain amount of time. The geometric interpretation of this approach is that one can determine a connection between the heat kernel and intrinsic distances via Varadhan's formula [284]. Later, a scale invariant extension of the HKS was developed [51]. Other authors in [87] propose a point-based signature for three-dimensional mesh segmentation, called as heat mean signature, that is based on the average of heat kernels used by HKS. In [14] another feature descriptor namely the WKS inspired by equations of theoretical physics is proposed. Based on the Schrödinger equation, the WKS represents the average probability of measuring a quantum mechanical particle at a specific location. At the same time, [49] proposed a scheme that is able to generalise the diffusion-based approaches. All of these methods are computed using the kernels of the analytical solutions of the underlying PDEs. In particular, the feature descriptors are based on the spectral decomposition of the Laplace-Beltrami operator and can be represented by a truncated series using the eigenvalues and eigenfunctions of the discrete Laplacian. In general, it is not immediately evident when to truncate the series for practical purposes, yet some strategies have been given in the literature [50]; see also [69] for a recent, related investigation.

In order to avoid eigendecompositions some works [208, 209, 285] employ rational approximations of the matrix exponential, where the eigenvalues of the discrete Laplacian are not of any great relevance in the actual computations. The mentioned work [285] is dealt with fast multi-scale heat kernel computation. The basic idea is the combination of a multi-resolution approach (low to high resolution representation of the given shape) and computing the matrix exponential by the popular scaling and squaring method. The multi-resolution approach can achieve fast and suitable approximations, but the method also has some limitations, especially from a theoretical point of view. In a similar context regarding diffusion distance computations, Patané [208, 209] suggests the use of the wFEM heat kernel with an additional computation of the matrix exponential via Padé-Chebyshev approximation. The proposed wFEM heat kernel is intrinsically scale-invariant and consequently robust to shape and scale changes. Nevertheless, the Padé approximation is linked to high computational costs when considering all points of the shape. To reduce the CPU time, the computation of the matrix exponential can be replaced by an approximation to the matrix exponential operator on an operand vector. The matrix-vector product can be efficiently computed by the Krylov subspace projection technique as performed in [301] for image smoothing using the heat kernel. However, the accuracy of this method depends on the spectrum of the underlying matrix and is thus often used as an iterative procedure. In consequence, the Krylov subspaces have to be computed newly at each time level.

An alternative approach to the kernel-based methods are the time integration methods of the PDEs considered in [68]. In addition to the abovementioned types of PDEs the shape signature defined via the classic wave equation [67] has been realised in this setting. Compared to the kernel-based methods the numerical integration as reported in [67, 68] generally yield more accurate shape correspondences, while the computation of the numerical signatures is much more time-consuming than the kernel-based signatures.

In [179] the authors developed a feature descriptor using a specific discrete diffusion process without solving an eigenvalue problem. This approach is characterised via a discrete time evolution and using geodesic distances instead of the Laplace-Beltrami operator. In other words, it derives the relation between elements by exploiting an alternative evolution paradigm rather than considering geometric PDEs. This construction achieves better results in terms of matching performance than the kernel-based methods, whereas the method based on the computation and storage of pairwise geodesic distances is still very computationally intensive.

Finally, an overview of the mentioned approaches for shape analysis by time-evolution methods is given in Figure 5.2. Let us emphasise that the works [210, 282, 302] provide a good introduction when working with the Laplace-Beltrami operator on surfaces and volumes for the first time, with particular attention to shape correspondence, spectral mesh processing or Laplace spectral distances and kernels.

5.2 About the Shape Correspondence Framework

In this section we introduce the basic facts that are necessary to define the shape correspondence framework. Concerning the general shape analysis set-up, we largely follow concepts as discussed for instance in [50]. For notions from differential geometry as employed here we refer the reader to [79].

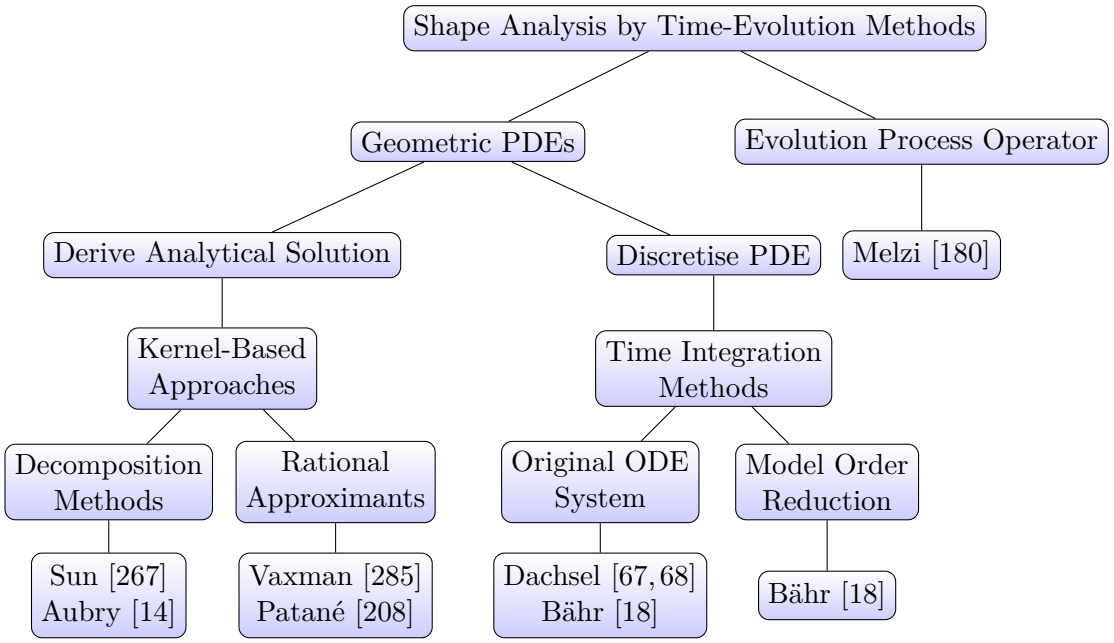


Figure 5.2: Approaches for shape analysis and matching by time-evolution methods.

5.2.1 Almost Isometric Shapes

A three-dimensional geometric shape can be described by its bounding surface. Thus, our shape model consists of compact two-dimensional Riemannian manifolds $\mathcal{M} \subset \mathbb{R}^3$, equipped with the metric tensor $\underset{\sim}{g} \in \mathbb{R}^{2\times 2}$ that describes locally the geometry.

Two shapes $\mathcal{M}$ and $\widetilde{\mathcal{M}}$ may be considered as isometric if there is a smooth homeomorphism $T : \mathcal{M} \to \widetilde{\mathcal{M}}$ between the corresponding object surfaces that preserves the intrinsic distances between surface points:

$$d_{\mathcal{M}}(\boldsymbol{x}_1, \boldsymbol{x}_2) = d_{\widetilde{\mathcal{M}}}\left(T(\boldsymbol{x}_1), T(\boldsymbol{x}_2)\right), \quad \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{M} \tag{5.1}$$

The intrinsic distance between two surface points $\boldsymbol{x}_k$, $k = 1, 2$ can be interpreted as the shortest path along the surface $\mathcal{M}$ connecting $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$.

In many applications, the notion of isometric shapes may be too restrictive. For instance, small noise in a dataset could be considered as an elastic deformation yielding some distortions in intrinsic distances. To take into account this issue, we call two shapes $\mathcal{M}$ and $\widetilde{\mathcal{M}}$ almost isometric, if there exists a transformation $S : \mathcal{M} \to \widetilde{\mathcal{M}}$ with

$$d_{\mathcal{M}}(\boldsymbol{x}_1, \boldsymbol{x}_2) \approx d_{\widetilde{\mathcal{M}}}\left(S(\boldsymbol{x}_1), S(\boldsymbol{x}_2)\right), \quad \forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathcal{M} \tag{5.2}$$

5.2.2 PDE-Based Models for Shape Description

A classic but still modern descriptor class that can handle almost isometric transformations is based on physical models that are conveniently described by PDEs. In the following we introduce the two[1] fundamental PDEs that we employ to this end.

[1] The geometric Schrödinger equation can also be used in this setting, see [68].

The *geometric heat equation* that yields a useful shape descriptor [267] involves the Laplace-Beltrami operator. This is the geometric[2] version of the Laplace operator that takes into account the curvature of a smooth manifold in 3D. Given a parameterisation of such a two-dimensional manifold, the Laplace-Beltrami operator applied to a scalar function $u : \mathcal{M} \to \mathbb{R}$ can be expressed in local coordinates as

$$\Delta_{\mathcal{M}} u = \frac{1}{\sqrt{|g|}} \sum_{i,j=1}^{2} \partial_i \left(\sqrt{|g|} g^{ij} \partial_j u \right) \tag{5.3}$$

where g^{ij} are the entries of the inverse of the metric tensor and $|g|$ is its determinant. Using this the geometric heat equation reads as

$$\partial_t u(\boldsymbol{x}, t) = \Delta_{\mathcal{M}} u(\boldsymbol{x}, t), \qquad \boldsymbol{x} \in \mathcal{M},\ t \in I \tag{5.4}$$

and describes how heat would diffuse along a surface $\mathcal{M}$.

The *geometric wave equation* is the second PDE that is going to be discussed in this chapter. It has been introduced in [67] as a useful model for computing pointwise a shape descriptor. Assuming that the speed of wave propagation is identical to one in all directions on the manifold, the corresponding PDE is

$$\partial_{tt} u(\boldsymbol{x}, t) = \Delta_{\mathcal{M}} u(\boldsymbol{x}, t), \qquad \boldsymbol{x} \in \mathcal{M},\ t \in I \tag{5.5}$$

Both of the PDEs described require an initial condition in order to be meaningful. In the context of shape correspondence construction, we employ a Dirac delta function $u(\boldsymbol{x}, 0) = u_0(\boldsymbol{x}) = u_{\boldsymbol{x}_i}$ centred around a point of interest $\boldsymbol{x}_i \in \mathcal{M}$. The PDE (5.5) is of second order in time, so that it needs to be supplemented not only by a spatial function as an initial state, but also an account of the initial velocity of that initial state is needed. As it is a canonical choice, we consider the zero initial velocity condition $\partial_t u(\boldsymbol{x}, 0) = 0$.

Remark 5.1. *Many shapes appear as a closed manifold with $\partial\mathcal{M} = \emptyset$, where it is not necessary to define additional boundary conditions. For the case $\mathcal{M}$ is bounded, often homogeneous Neumann boundary conditions are used.*

5.2.3 Feature Descriptor and Shape Correspondence

We now make precise how geometric feature descriptors are obtained by employing the introduced PDEs, and how we construct shape correspondence on that basis.

Feature Descriptor For many shape analysis tasks, it is useful to consider a pointwise feature descriptor as a shape representation. The purpose of the feature descriptor is to give an account of the geometry of the surface at a certain local region centred about a considered point. To this end, we restrict the spatial component of solutions $u(\boldsymbol{x}, t)$ of the PDEs used to

$$f_{\boldsymbol{x}_i}(t) := u(\boldsymbol{x}, t)_{|\boldsymbol{x}=\boldsymbol{x}_i} \quad \text{with} \quad u(\boldsymbol{x}, 0)_{|\boldsymbol{x}=\boldsymbol{x}_i} = u_{\boldsymbol{x}_i} \tag{5.6}$$

and call the $f_{\boldsymbol{x}_i}(t)$ the *pointwise feature descriptor* at the location $\boldsymbol{x}_i \in \mathcal{M}$.

[2] Let us emphasise again that we refer to these PDEs in this thesis as geometric PDEs as they take into account geometric surface information, although the geometry does not change during time evolution.

Let us comment that there exists a physical interpretation related to the feature descriptors we are using. The heat-based feature descriptor $f_{\boldsymbol{x}_i}(t)$ describes the rate of heat transferred away from the considered point $\boldsymbol{x}_i$, see Figure 5.3. The spreading of heat takes into account the geometry of the surface using the Laplace-Beltrami operator. In turn, the wave-based feature descriptor describes the motion amplitudes of an emitted wave front observed at the considered point $\boldsymbol{x}_i$ during time evolution. More precisely, the latter feature descriptor catches the typical wave interaction observable in the solution of the wave equation as can e.g. be seen in the well-known formula of d'Alembert in the one-dimensional case. Analogously to the situation for the heat signature, the observable motion of the waves is influenced by the intrinsic geometry of the surface. As time evolves, the waves spread over the surface so that their amplitude observed via $f_{\boldsymbol{x}_i}(t)$.

Remark 5.2. *The feature descriptors discussed here cannot distinguish between intrinsic symmetry groups as they are based on intrinsic shape properties.*

The feature descriptor defined above is in general a simplified representation and make it therefore a very lucrative candidate for a pointwise signature in the PSC application. It is also possible to define a compact point signature as a family of functions

$$f_{\boldsymbol{x}_i}(\boldsymbol{x}, t) := u(\boldsymbol{x}, t) \quad \text{with} \quad u(\boldsymbol{x}_i, 0) = u_{\boldsymbol{x}_i} \tag{5.7}$$

On this basis, the complexity of storing and comparing the signatures (5.7) of two different points would be extremely high. Nonetheless, we note that there are some related works in this case, e.g. [199].

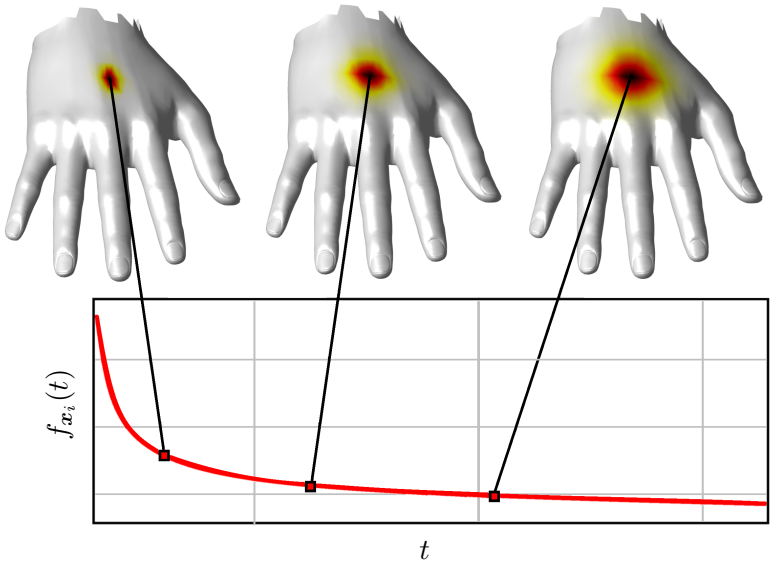


Figure 5.3: The dynamics described by the geometric heat equation (5.4). The initial condition is based on the Dirac delta function $u_0(\boldsymbol{x}) = u_{\boldsymbol{x}_i}$. The time evolution of u is shown from left to the right. The feature descriptors $f_{\boldsymbol{x}_i}(t)$ at the location $\boldsymbol{x}_i \in \mathcal{M}$ contains geometric information of the local neighbourhood. In simple terms, $f_{\boldsymbol{x}_i}(t)$ describes how much heat remains at $\boldsymbol{x}_i$ after t seconds.

Shape Correspondence To compare the feature descriptors for different locations $\boldsymbol{x}_i \in \mathcal{M}$ and $\widetilde{\boldsymbol{x}}_j \in \widetilde{\mathcal{M}}$ on respective shapes $\mathcal{M}$ and $\widetilde{\mathcal{M}}$, we employ a distance $d_f(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j)$ using the L_1-norm as

$$d_f(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j) = \int_I \left| f_{\boldsymbol{x}_i}(t) - f_{\widetilde{\boldsymbol{x}}_j}(t) \right| \mathrm{d}t \tag{5.8}$$

It is clear that the tuple of locations $(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j) \in \mathcal{M} \times \widetilde{\mathcal{M}}$ with the smallest feature distance should belong together. This consideration naturally leads to a minimisation problem for all locations in the form:

$$(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j) = \underset{\widetilde{\boldsymbol{x}}_k \in \widetilde{\mathcal{M}}}{\arg\min}\, d_f(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_k) \quad \Longleftrightarrow \quad \boldsymbol{x}_i \longleftrightarrow \widetilde{\boldsymbol{x}}_j \tag{5.9}$$

The latter relation can also be expressed using $\widetilde{\boldsymbol{x}}_j = S(\boldsymbol{x}_i) = \boldsymbol{x}_i$, so that the map S can pointwise be restored for all $\boldsymbol{x}_i$. Let us remark that without further alignment it cannot be expected that the restored map S is injective or surjective, since the minimisation condition is not unique.

5.3 Basic Discretisation of Continuous-Scale Models

This section will recall the basics of the discretisation of the PDEs that we use. In order to prepare for later developments that are at the heart of the contributions of this chapter, let us note that we really give just the description of the fundamentals. The concrete schemes used for computations are relying on the technical building blocks we introduce here.

5.3.1 Discretising in Space and Time

The discrete surface representation for the computations is given by a triangular mesh which we denote as $\mathcal{M}_d = (P, T)$, cf. Figure 5.4. The underlying point cloud $P := \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ contains a finite number of vertices in terms of coordinate points. The mesh is constructed by connecting the vertices $\boldsymbol{x}_i$ so that one obtains triangular cells. The individual triangles T contain the neighbourhood relations between corresponding vertices. As visualised in Figure 5.4, let Ω_i be the barycentric cell volume that surrounds the i-th vertex. Turning from space to time discretisation, we define time intervals $I_k = [t_k, t_{k+1}]$ and set $t_0 = 0$ to subdivide the complete integration time $[0, t_F]$.

5.3.2 Finite Volumes: Semi-Discrete Form

Letting for the moment ∂_* be either ∂_t or ∂_{tt}, we consider the PDEs (5.4) and (5.5) over a so-called control volume Ω_i and a time interval I_k. Integration in space and time yields

$$\int_{I_k} \int_{\Omega_i} \partial_* u(\boldsymbol{x}, t) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t = \int_{I_k} \int_{\Omega_i} \Delta_{\mathcal{M}} u(\boldsymbol{x}, t) \, \mathrm{d}\boldsymbol{x} \, \mathrm{d}t \tag{5.10}$$

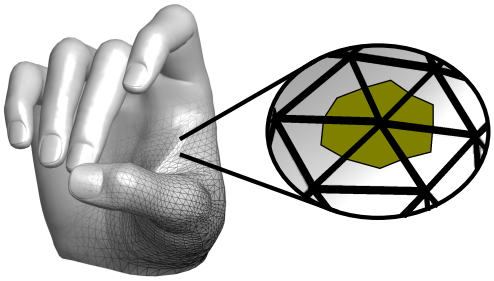


Figure 5.4: Continuous and discrete shape representation. The discrete shape is given by nonuniform linear triangles. Volume cells as shown here in green are constructed using the barycentric area around a vertex.

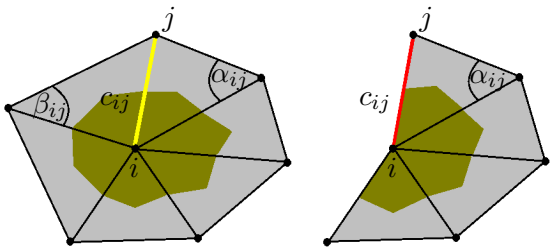


Figure 5.5: The cotangent weight scheme as discretisation of the Laplace-Beltrami operator. **Left:** Interior edge. **Right:** Boundary edge.

In a finite volume method the quantities that are considered in the computations are cell averages, i.e. for the i-th control volume, or cell, we define

$$u_i(t) = u(\bar{\boldsymbol{x}}_i, t) = \frac{1}{|\Omega_i|} \int_{\Omega_i} u(\boldsymbol{x}, t)\, \mathrm{d}\boldsymbol{x} \tag{5.11}$$

where $|\Omega_i|$ denotes the area of the i-th control volume. Therefore, the averaged Laplacian is defined as

$$Lu_i(t) = \frac{1}{|\Omega_i|} \int_{\Omega_i} \Delta_{\mathcal{M}} u(\boldsymbol{x}, t)\, \mathrm{d}\boldsymbol{x} \tag{5.12}$$

As for the meaning of the latter integral on the right hand side, one has to apply the divergence theorem to substitute the volume integral into a line integral over the boundary of the cell volume. For the discretisation of the arising integral quantities, the widespread cotangent weight scheme as introduced in [184] is employed.

The arising discrete Laplace-Beltrami operator $L \in \mathbb{R}^{N \times N}$ is composed of the sparse matrix representation that can be written as $L = D^{-1}C$. The appearing symmetric *cotangent weight* matrix C contains the entries

$$C_{i,j} = \begin{cases} -\sum\limits_{j \in N_i} c_{ij}, & \text{if } i = j \\ c_{ij}, & \text{if } i \neq j \text{ and } j \in N_i \\ 0, & \text{else} \end{cases} \tag{5.13}$$

where N_i denotes the set of points adjacent to the vertex $\boldsymbol{x}_i$. The weights c_{ij} of the edge (i, j) between corresponding vertices distinguish between interior E_i and boundary edges E_b as shown in Figure 5.5, and are given by

$$c_{ij} = \begin{cases} \dfrac{\cot \alpha_{ij} + \cot \beta_{ij}}{2}, & \text{if } (i,j) \in E_i \\ \dfrac{\cot \alpha_{ij}}{2}, & \text{if } (i,j) \in E_b \end{cases} \tag{5.14}$$

Furthermore, α_{ij} and β_{ij} denote the two angles opposite to the edge (i, j), and the matrix

$$D = \operatorname{diag}\left(|\Omega_1|, \ldots, |\Omega_i|, \ldots, |\Omega_N|\right) \tag{5.15}$$

contains the local volume cell areas.

Remark 5.3. *The Laplacian matrix L is ultimately not symmetric. This fact has a significant influence on the resulting computational setting and is discussed in detail in Section 5.4.1.*

Lastly, we now put together and summarise the components of the discretisation as we developed it until now. In this way we end up with a semi-discrete form of the scheme. Let a function defined on all cells be represented by now as an N-dimensional vector

$$\boldsymbol{u}(t) = \left(u_1(t), \ldots, u_N(t)\right)^\top \tag{5.16}$$

Rewriting (5.10) using volume cell averages we obtain a semi-discrete system of ODEs, one for each control volume:

$$\overset{*}{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) \qquad \text{where} \qquad \overset{*}{\boldsymbol{u}}(t) = \frac{\mathrm{d}^*\boldsymbol{u}(t)}{\mathrm{d}t^*} \tag{5.17}$$

where the use of the star derivative $\overset{*}{\boldsymbol{u}}$ indicates the time derivatives of first and second order, as introduced in (5.10).

Standard methods for the numerical solution of (5.17) in general deal directly with first order ODE systems. For the geometric heat equation the system (5.17) reads as

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t), \quad t \in (0, t_F],\, \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{5.18}$$

In contrast, the geometric wave equation is of second order, and a transformation into a first order system

$$\dot{\boldsymbol{q}}(t) = K\boldsymbol{q}(t), \quad t \in (0, t_F] \tag{5.19}$$

with the matrix

$$K = \begin{pmatrix} 0 & I \\ L & 0 \end{pmatrix} \in \mathbb{R}^{2N \times 2N} \tag{5.20}$$

where $I \in \mathbb{R}^{N \times N}$ is the identity matrix and

$$\boldsymbol{q}(t) = \left(\boldsymbol{q}_1(t), \boldsymbol{q}_2(t)\right)^\top = \left(\boldsymbol{u}(t), \dot{\boldsymbol{u}}(t)\right)^\top \tag{5.21}$$

is necessary.

We now turn to the discrete initial conditions of the time evolutions described. As indicated, the initial velocity function $\dot{\boldsymbol{u}}(0)$ for use with the geometric wave equation is identical to zero. Thus, we have only to describe here the discrete setting for the initial spatial density $\boldsymbol{u}(0)$ which is used for both the geometric heat and wave equation. To this end, a discretised version of the Dirac delta function has to be constructed that we formally employed in the

continuous-scale model. Using the cell average

$$\int_{\Omega_i} u(\boldsymbol{x}, 0)\, \mathrm{d}\boldsymbol{x} = 1 \tag{5.22}$$

where $u(\boldsymbol{x}, 0)$ may be interpreted now as a box function with unit area, being expressed as

$$u(\boldsymbol{x}, 0) = \begin{cases} \frac{1}{|\Omega_i|}, & \text{if } \boldsymbol{x} \in \Omega_i \\ 0, & \text{else} \end{cases} \tag{5.23}$$

the initial condition at the location $\boldsymbol{x}_i$ can be formulated as

$$\boldsymbol{u}^{i,0} := \Big(0, \ldots, 0, |\Omega_i|^{-1}, 0, \ldots, 0\Big)^\top \tag{5.24}$$

that implicitly bears a dependence on the index i.

5.3.3 Time Integration

Solving the arising ODE systems (5.18) and (5.19) involves the application of numerical integration and can be done using the time stepping methods described in Section 2.1, such as the EE method, the IE method or the CN method. In a similar application for smoothing meshes, the authors [78] suggest to use an implicit integrator for the underlying diffusion equation. This also holds for the PSC application, which can be explained as follows.

Obviously, the EE method leads to limitations of the time step size depending on the volume cell areas when applied for shape analysis tasks. The main reason is that in typical discrete meshes representing shapes, one has to face in general a large variety of mesh widths, and especially also very small mesh widths arise. As the spatial mesh width and the allowed time step size of explicit methods are coupled, exactly this issue makes the explicit methods not being attractive in our setting. Although the application of fast explicit methods as employed in [18], such as the FEDRK[3] scheme, can substantially reduce the computational costs compared to the EE method, its use is not preferable as a consequence of the L_∞-stability violation. Thus, the discrete maximum-minimum principle (positivity property) is not fulfilled which may yield undesirable oscillations with respect to the discrete feature descriptors, see Figure 5.6.

Usually, the implicit CN scheme is frequently used, see for example Chapter 6. However, we do not consider this approach to be practicable here due to the underlying initial condition. The latter method is not L-stable which may result in undesirable oscillations in the numerical solution for problems with discontinuous initial conditions as employed herein, see Figure 5.6. In other words, oscillations-free numerical signatures are of interest in our approach. In such a case, naturally L-stable schemes are preferred, so the IE method, as [67, 68] has shown, represents a reasonable choice for our purpose.

In total, we only consider here the IE method for the numerical solution of the underlying geometric PDEs. In Section 5.6.4 it will also be shown that the IE scheme provides a practical feature because of its artificial damping.

[3] Although the system matrix L is not symmetric, FEDRK can be applied. By multiplication of D to the equation (5.18) we have $D\boldsymbol{u}^{k+1} = (D + \tau C)\boldsymbol{u}^k$, where C is symmetric and negative semi-definite.

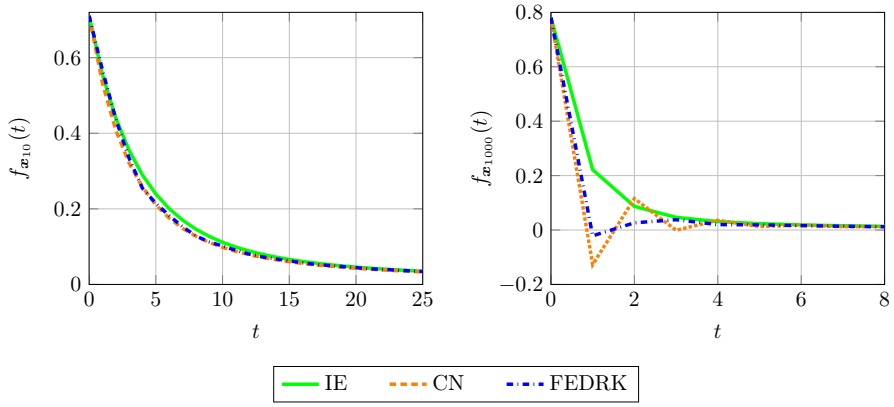


Figure 5.6: Results for the *wolf* dataset using the geometric heat equation for $t_F = 25$. We compare the discrete feature descriptor $f_{\boldsymbol{x}_{10}}(t)$ (**left**) and $f_{\boldsymbol{x}_{1000}}(t)$ (**right**) obtained by the methods IE, CN and FEDRK. To enhance the visual comparability, the feature descriptor $f_{\boldsymbol{x}_{1000}}(t)$ is only plotted within the range $[0, 8]$. Obviously, oscillations-free numerical signatures are generated via IE method, contrary to CN and FEDRK as on the right of the figure.

Implicit Euler Method for the Geometric PDEs Applying the fundamental theorem of calculus and using the right-hand rectangle method for the integral approximation of the right-hand side of (5.18), we obtain the IE method

$$(I - \tau L)\, \boldsymbol{u}^{k+1} = \boldsymbol{u}^k \tag{5.25}$$

using the notation $\boldsymbol{u}(t_{k+1}) = \boldsymbol{u}^{k+1}$ and with the uniform time step size $\tau = t_{k+1} - t_k$, $k \in \{0, \dots, F-1\}$ as well as $\boldsymbol{u}^0 = \boldsymbol{u}^{i,0}$. The analogous application of the same approximation scheme to the geometric wave equation (5.19) leads to

$$\boldsymbol{q}^{k+1} = \boldsymbol{q}^k + \tau K \boldsymbol{q}^{k+1} \tag{5.26}$$

In (5.26) the component $\boldsymbol{q}_1$ at times t_{k+1} and t_k reads

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \tau \dot{\boldsymbol{u}}^{k+1} \tag{5.27}$$

$$\boldsymbol{u}^k = \boldsymbol{u}^{k-1} + \tau \dot{\boldsymbol{u}}^k \tag{5.28}$$

while the component $\boldsymbol{q}_2$ at t_{k+1} can be expressed as

$$\dot{\boldsymbol{u}}^{k+1} = \dot{\boldsymbol{u}}^k + \tau L \boldsymbol{u}^{k+1} \tag{5.29}$$

The combination of (5.27)-(5.29) transform (5.26) into a two step approach

$$\left(I - \tau^2 L\right) \boldsymbol{u}^{k+1} = 2\boldsymbol{u}^k - \boldsymbol{u}^{k-1} \tag{5.30}$$

with a system size $N \times N$. The wave equation requires to define two initial conditions, namely $\boldsymbol{u}(t_0)$ and $\dot{\boldsymbol{u}}(t_0)$. With $\boldsymbol{u}^0 = \boldsymbol{u}^{i,0}$ and the fixed initial velocity $\dot{\boldsymbol{u}}^0 = \boldsymbol{0}$ it follows that

$$\left(I - \tau^2 L\right) \boldsymbol{u}^1 = \boldsymbol{u}^0 \quad \text{for} \quad k = 0 \tag{5.31}$$

Remark 5.4. *The IE scheme can be applied analogously to the geometric Schrödinger equation [68]. However, at each time level a linear system with complex coefficients has to be solved which is much more cost intensive. At the same time, the descriptors based on the wave equation generally achieve better results, cf. [67]. Therefore, we only consider here the geometric heat and wave equation.*

5.4 Numerical Solvers

In this section we provide some relevant information about the sparse direct solver, the CG method, the KSMOR technique and the MCR approach for the PSC application. Based on the fact that the methods strongly depend on the properties of the system matrix L, we give a discussion of the discrete Laplace-Beltrami operator beforehand. In doing so, the eigenvalues, eigenvectors and the definiteness of the discrete Laplacian are analysed.

5.4.1 Discrete Laplace-Beltrami Operator

We focus now on the properties of the discrete Laplace-Beltrami operator L, more precisely on the eigenvalues λ and eigenfunctions $\boldsymbol{\phi}$ of the matrix L. The basic problem, also called the (standard) eigenvalue problem, is to determine $\lambda \in \mathbb{C}$ and $\boldsymbol{\phi} \in \mathbb{R}^N$, $\boldsymbol{\phi} \neq \mathbf{0}$ such that

$$L\boldsymbol{\phi} = \lambda\boldsymbol{\phi} \tag{5.32}$$

is fulfilled, where $L \in \mathbb{R}^{N\times N}$ is nonsymmetric here. This task causes both theoretical and numerical problems. On the one hand, nonsymmetric matrices do not guarantee real eigenvalues and eigenvectors. On the other hand, their numerical computation may yield complex-valued results even if they were real. These aspects are discussed below. A general overview on the Laplace operator and its properties is presented in [302].

Generalised Eigenvalue Problem As mentioned, the discrete Laplace-Beltrami operator is given by $L = D^{-1}C$, where D is a regular diagonal matrix with positive entries on the diagonal, and C is a symmetric matrix. Under these conditions the eigenvalue problem (5.32) can be reformulated as a *generalised eigenvalue problem (GEP)* via $D^{-1}C\boldsymbol{\phi} = \lambda\boldsymbol{\phi}$ or

$$C\boldsymbol{\phi} = \lambda D\boldsymbol{\phi} \tag{5.33}$$

which have the same eigenvalues and eigenvectors as the original problem. It should be noted that if C and D are symmetric and D is also positive definite, which is the case here, then all eigenvalues λ are real and the N eigenvectors $\boldsymbol{\phi}$ are linearly independent, whereby the eigenvectors are D-orthogonal with $\boldsymbol{\phi}_i^\top D\boldsymbol{\phi}_j = \delta_{ij}$, see [207]. This means, the eigenvectors are orthogonal with respect to the inner product

$$\langle \boldsymbol{f}, \boldsymbol{g}\rangle_D = \boldsymbol{f}^\top D\boldsymbol{g} \tag{5.34}$$

As a result, the equalities hold

$$L = \Phi\Lambda\Phi^\top D, \quad I = \Phi^\top D\Phi, \quad \Lambda = \Phi^\top C\Phi \tag{5.35}$$

where Λ is a diagonal matrix of eigenvalues and Φ corresponds to the right eigenvector matrix of L. Therefore, the eigenvalues of the underlying matrix L are real and the eigenfunctions are D-orthogonal. The definiteness of L can specified by the following proposition:

Proposition 5.1. *The discrete Laplace-Beltrami operator L is negative semi-definite.*

Proof. The cotangent weight matrix C is a symmetric diagonally dominant matrix with real negative diagonal entries. According to the Gershgorin's circle theorem [100] it follows that C is negative semi-definite. This implies that L is also negative semi-definite with respect to the inner product (5.34), because of

$$\langle \boldsymbol{x}, L\boldsymbol{x} \rangle_D = \boldsymbol{x}^\top DL\boldsymbol{x} = \boldsymbol{x}^\top C\boldsymbol{x} \leq 0, \quad \forall \boldsymbol{x} \neq \mathbf{0} \tag{5.36}$$

□

In particular, the eigenvalues of the discrete Laplacian are real nonpositive analogous to the continuous Laplace-Beltrami operator. Let us highlight that the numerical solution of (5.32), although the eigenvalues of L are real, may produce complex-valued results as exemplarily shown Figure 5.7. It is generally advantageous to compute $\boldsymbol{\phi}$ and λ by making use of (5.33) due to the fact, that numerical methods for the GEP recognise the developed theoretical properties and generate real eigenvalues and eigenvectors.

Remark 5.5. *Another characteristic of the discrete Laplacian is its zero row sum property. Consequently, zero is an eigenvalue of L with associated constant eigenvector $\boldsymbol{g} := (c, \ldots, c)^\top$, since $L\boldsymbol{g} = \mathbf{0}$.*

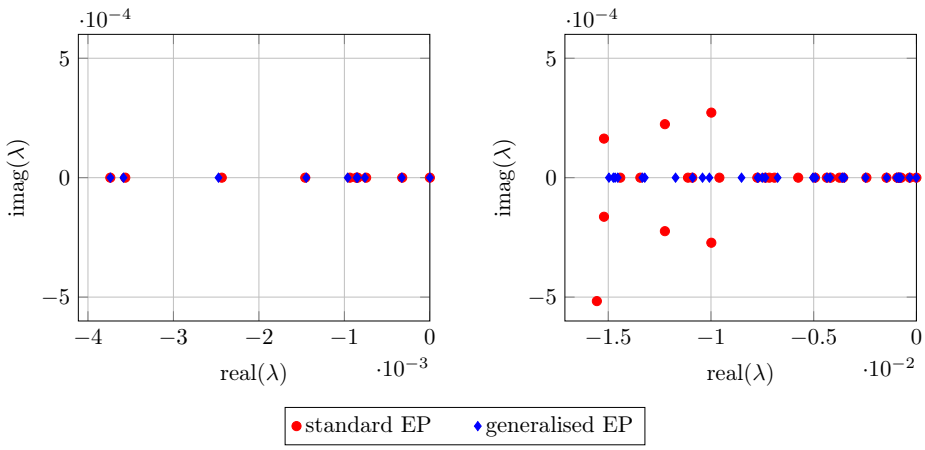


Figure 5.7: Comparison of the numerical eigenvalue problem computation on the *wolf* dataset using the standard eigenvalue problem (5.32) and the GEP (5.33). The eigenvalues are computed by the internal MATLAB R2018b function *eigs*. **Left:** Computation of the first ten smallest eigenvalues. **Right:** Computation of the first thirty smallest eigenvalues. It can be seen that the internal MATLAB solvers regarding the standard eigenvalue problem and GEP produces not the same results, since the eigenvalues are not equally even for a small number. Interestingly, the numerical solution of (5.32), although the eigenvalues are theoretically real, produces complex-valued results as presented on the right of the figure.

5.4.2 Implicit Solvers

As described in the last section the temporal integration will be done implicitly. An essential key requirement for our objective of a correct shape matching is a sufficient accuracy of the computed numerical solution. Apart from that, the underlying PDEs that are used to this end have to be solved for each point and on each shape for a fixed time interval $t \in (0, t_F]$. Consequently, the computational costs are directly related to the number of points of the regarded shapes. This suggests that one may forego high accuracy in exchange for a faster CPU time. In order to evaluate this proceeding, an analysis of the numerical solvers related to shape matching is absolutely essential.

The implicit schemes (5.25) and (5.30) result in a large sparse systems of linear equations and can be expressed as

$$A\boldsymbol{x} = \boldsymbol{b} \tag{5.37}$$

with $A = I - \tau L$, $\boldsymbol{b} = \boldsymbol{u}^k$, $\boldsymbol{x} = \boldsymbol{u}^{k+1}$ for the geometric heat equation and $A = I - \tau^2 L$, $\boldsymbol{b} = 2\boldsymbol{u}^k - \boldsymbol{u}^{k-1}$, $\boldsymbol{x} = \boldsymbol{u}^{k+1}$ for the geometric wave equation. The matrix $A \in \mathbb{R}^{N \times N}$ is positive definite, nonsymmetric, large, sparse and structured since it is based on a discretisation of the Laplace-Beltrami operator. As stated in Section 2.2 the linear system (5.37) can be solved with either sparse direct or sparse iterative solver.

Sparse Direct Solver In this case, the underlying matrix A will be factorised just once into a product of triangular matrices $A = LU$ using a complete and sparse LU factorisation. Then such systems are solved for each right-hand side by forward and backward substitution, which is apparently very efficient.

To improve the performance of the sparse direct solver, an alternative object-oriented factorisation is useful to solve the linear system. In contrast to the LU factorisation, precomputing the matrix factors in the object-oriented framework is more expensive, but this only has to be done once and it yields in total a faster solver with exactly the same results. In order to accelerate the computation in this way we use the *SuiteSparse* package [74], more precisely the function *factorize*.

Remark 5.6. *The internal MATLAB-function decomposition can also be used. For the PSC setup considered here, the SuiteSparse package provides a better performance.*

At this point, we mention another possibility to use the direct solver which may be very effective in this framework. As indicated, we are interested in constructing pointwise feature descriptors $f_{\boldsymbol{x}_i}(t)$ which are essentially not based on a solution of the heat and wave equation in a global sense. From a practical point of view it therefore does not seem useful to solve the geometric PDEs on the entire discrete mesh. The latter statement suggests to solve the PDEs only locally, i.e. in a certain area, close to the considered point $\boldsymbol{x}_i$. The localised solving can be obtained by localising the underlying Cholesky factorisation as proposed in [126, 127]. This technique enables to significantly speed up the computations and may yield a further advancement in the future.

Sparse Iterative Solver In contrast to direct methods, iterative solvers are very efficient in computing approximate solutions. This may be very useful in the PSC application considered here. As mentioned before, the purpose of the feature descriptor $f_{\boldsymbol{x}_i}(t)$ is to give an account

of the geometry of the surface at a certain local region centred around $\boldsymbol{x}_i$, cf. Figure 5.3. Consequently, the construction of feature descriptors generally requires only a sufficiently accurate numerical solution in a local area. This may suggest that a good representation of the shape signature is achieved after a small number of iterations.

We propose to employ the highly efficient CG method for problems involving sparse symmetric and positive definite matrices, cf. Section 2.2.1. In order to realise the symmetric case in (5.37), the multiplication from the left of the matrix D in (5.15) to the equations (5.25) and (5.30) results in

$$(D - \tau C)\,\boldsymbol{u}^{k+1} = D\boldsymbol{u}^k \tag{5.38}$$

$$\left(D - \tau^2 C\right)\boldsymbol{u}^{k+1} = 2D\boldsymbol{u}^k - D\boldsymbol{u}^{k-1} \tag{5.39}$$

where $D - \tau^\alpha C$ with $\alpha = 1, 2$, is symmetric positive definite due to the properties of the cotangent weight matrix C. Therefore, the CG method is solving the new system

$$\widetilde{A}\boldsymbol{x} = \widetilde{\boldsymbol{b}} \tag{5.40}$$

with $\widetilde{A} = D - \tau C$, $\widetilde{\boldsymbol{b}} = D\boldsymbol{u}^k$ for the geometric heat equation and $\widetilde{A} = D - \tau^2 C$, $\widetilde{\boldsymbol{b}} = 2D\boldsymbol{u}^k - D\boldsymbol{u}^{k-1}$ for the geometric wave equation.

For the PSC application, we will investigate the effects of the user-defined termination of the CG algorithm if the approximate solution reaches a specific convergence tolerance or a maximum number of iterations. The use of the MIC preconditioner for large systems is also being investigated.

5.4.3 Model Order Reduction

The implicit methods have to handle large sparse systems, whereby the computational costs depend on the point cloud size. The MOR techniques, as presented in Chapter 4, can be used to approximate the original linear and time-invariant first order ODE system (5.18) and (5.19) by a very low dimensional system, thereby preserving the main characteristics of the original ODE system. Before we discuss specific techniques, let us briefly describe the general procedure of MOR on the system (5.18). Of course, the approach can be applied analogously to the geometric wave equation (5.19).

Applying the MOR concept to (5.18) may be understood as projecting the original system

$$\begin{cases} \dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) \\ y_i(t) = \boldsymbol{e}_i^\top \boldsymbol{u}(t), \quad \boldsymbol{u}_i(0) = \boldsymbol{u}^{i,0},\ i = 1, \ldots, N \end{cases} \tag{5.41}$$

with the state variable $\boldsymbol{u} \in \mathbb{R}^N$, the single output variable $y_i(t) \in \mathbb{R}$ and the i-th unit vector $\boldsymbol{e}_i \in \mathbb{R}^N$ onto a reduced order model

$$\begin{cases} W^\top V \dot{\boldsymbol{u}}_r(t) = W^\top L V \boldsymbol{u}_r(t) \\ \qquad y_{r,i}(t) = \boldsymbol{e}_i^\top V \boldsymbol{u}_r(t), \quad V\boldsymbol{u}_{r,i}(0) = \boldsymbol{u}^{i,0} \end{cases} \tag{5.42}$$

using a reduced basis representation $\boldsymbol{u}(t) \approx V\boldsymbol{u}_r(t)$ with $\boldsymbol{u}_r(t) \in \mathbb{R}^r$, $r \ll N$ and projection matrices $V \in \mathbb{R}^{N \times r}$ and $W^\top \in \mathbb{R}^{r \times N}$. We note that the dynamical system in (5.41) can be

interpreted as a *zero-input-single-output (ZISO)* system, whereby no input variable exists due to the considered boundary conditions. Notably, N different initial conditions are taken into account, which corresponds to extracting the feature descriptor $y_i(t)$ for each point on the given shape. By multiplication from the left with $(W^\top V)^{-1}$, assuming that the inverse exists, and using biorthogonal matrices $W^\top V = I_r$ the system (5.42) leads to the reduced system of order r as follows:

$$\begin{cases} \dot{\boldsymbol{u}}_r(t) = L_r \boldsymbol{u}_r(t) \\ y_{r,i}(t) = \boldsymbol{e}_{r,i}^\top \boldsymbol{u}_r(t), \quad V\boldsymbol{u}_{r,i}(0) = \boldsymbol{u}^{i,0} \end{cases} \tag{5.43}$$

with $L_r = W^\top L V \in \mathbb{R}^{r\times r}$, $\boldsymbol{e}_{r,i} = \boldsymbol{e}_i^\top V \in \mathbb{R}^r$ and $\boldsymbol{u}_{r,i} \in \mathbb{R}^r$.

As mentioned in Chapter 4, MCR, BT, POD and KSMOR are common and widespread approaches for constructing the projection matrices V, W. For our application purposes, however, the usability of the BT and POD method can be excluded on grounds of efficiency in advance. Both methods are based on performing an SVD. In addition, BT has to solve Lyapunov equations and POD has to form the snapshot matrix for each point on the given shape which results in inefficient processes.

An alternative approach to dealing with different initial condition could be the use of parametric MOR techniques. In this way, the dynamical system can be reformulated as a parameter-varying system, but this issue is beyond the scope of this work. In conclusion, our aim is to provide a short overview of the remaining two methods mentioned above, KSMOR and MCR.

Krylov Subspace Model Order Reduction The KSMOR methods presented in Section 4.5 are based on moment matching and approximate the transfer function of the original system (5.41), which describes the dependence between the input and the output. Based on the fact that no input variable exists, the output depends only on the initial condition which corresponds to the consideration of the zero-input response of the system. As a result of the latter fact a coordinate transformation is not necessary, although nonzero initial conditions are given.

Remark 5.7. *Since no coordinate transformation is required, the computational costs can be reduced. This follows from the fact, that a transformation is accompanied by an additional sparse matrix-vector multiplication, when translating the nonzero initial condition to the right-hand side of the ODE system. In the PSC application considered here, this proceeding would have to be performed for each point on each shape.*

Applying the Laplace transform to the ZISO system (5.41) we obtain

$$\begin{aligned} \boldsymbol{U}(s) &= (sI - L)^{-1}\, \boldsymbol{u}^{i,0} \\ \boldsymbol{Y}_i(s) &= \boldsymbol{e}_i^\top \boldsymbol{U}(s) = \boldsymbol{e}_i^\top\, (sI - L)^{-1}\, \boldsymbol{u}^{i,0} \end{aligned} \tag{5.44}$$

where the inverse $(sI - L)^{-1}$ exists for $s \neq \lambda_i$. Therefore, the transfer function is defined as

$$\boldsymbol{H}(s) = \boldsymbol{e}_i^\top\, (sI - L)^{-1}\, \boldsymbol{u}^{i,0} \tag{5.45}$$

which describes here the direct relation between the initial condition $\boldsymbol{u}^{i,0}$ and the output

$\boldsymbol{Y}_i(s)$ of the original system in the frequency domain. Based on this, the Taylor series of the transfer function expanded around σ is given by

$$\boldsymbol{H}(s) = -\sum_{k=0}^{\infty} m_k(\sigma)(s-\sigma)^k \tag{5.46}$$

where $m_k(\sigma) = \boldsymbol{e}_i^\top (L-\sigma I)^{-(k+1)} \boldsymbol{u}^{i,0}$ is the k-th moment. For the shape matching purpose the one-sided Arnoldi approach is applied, which constructs an orthogonal basis

$$\begin{aligned} \text{range}(V) &= \mathcal{K}_r\left((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\boldsymbol{u}^{i,0}\right) \\ \text{with } \mathcal{K}_r &:= \text{span}\left((L-\sigma I)^{-1}\boldsymbol{u}^{i,0}, \ldots, (L-\sigma I)^{-r}\boldsymbol{u}^{i,0}\right) \end{aligned} \tag{5.47}$$

with $W = V$ such that $W^\top V = V^\top V = I_r$.

Proposition 5.2. *Let V being a bases of $\mathcal{K}_r$ in (5.47) and $W = V$. For $L_r = W^\top L V$, $\boldsymbol{e}_{r,i} = \boldsymbol{e}_i^\top V$, $W^\top \boldsymbol{u}_i = \boldsymbol{u}_{r,i}$ and $\sigma \neq \lambda_i$, where λ_i is an eigenvalue of L, then the first r moments around σ are matched:*

$$m_k(\sigma) = \widetilde{m}_k(\sigma), \quad k = 0, \ldots, r-1 \tag{5.48}$$

where $\widetilde{m}_k(\sigma)$ are the moments of the transfer function $\boldsymbol{H}_r(s)$ of the reduced system (5.43).

Proof. Analogous to the Theorem 4.2. □

Remark 5.8. *As a result of the underlying stable dynamical system (5.41) built on the negative semi-definite matrix L, the one-sided KSMOR method, i.e. $W = V$, preserves the stability of the reduced system, cf. Lemma 4.1.*

Due to the above construction using Krylov subspaces, the KSMOR method is obviously not based on the computation of eigenvalues and eigenvectors and especially represents a *spectrum-free method*.

A parameter that still has to be determined is the choice of the expansion point σ. The underlying PDEs are obviously characterised by a rather slow dynamic, so approximating the system at the frequency $\sigma \approx 0$ is a natural choice. In particular, for $\sigma = 0$ the inverse $(L-\sigma I)^{-1}$ does not exist, since $\lambda = 0$ is an eigenvalue of L. Moreover, the setting $\sigma = \infty$ could be useful. This choice does not appear logical because high frequencies correspond to fast dynamics. However, in the PSC application the feature descriptors $f_{\boldsymbol{x}_i}(t)$ only require a sufficiently accurate numerical solution in a local neighbourhood of the considered point $\boldsymbol{x}_i$. This suggests that a potentially small number r achieves reasonable numerical signatures. The latter conjecture would be cost-effective, since the underlying construction of the Krylov subspace $V = \mathcal{K}_r(L, L\boldsymbol{u}^{i,0})$ is based solely on sparse matrix-vector multiplications.

For constructing the Krylov subspace $V = \mathcal{K}_r((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\boldsymbol{u}^{i,0})$ large sparse systems of linear equations have to be solved. This requires the application of the previously mentioned sparse direct or sparse iterative solvers. We apply here the sparse direct solver in combination with the SuiteSparse package [74]. The computational costs of KSMOR are directly linked to the costs of constructing the Krylov subspace V. At this point it should be stressed again that the projection matrix V must be recalculated at each point and on each

shape due to the need to consider various initial conditions. Therefore, the computational costs scale substantially when increasing the number of Krylov subspaces. Nonetheless, the KSMOR method can be highly practicable if a low number of subspaces represent a large part of the system dynamics.

Remark 5.9. *In general, the discrete feature descriptors $f_{\boldsymbol{x}_i}(t)$ are generated built on the integration time $[0, t_F]$, which is subdivided into $(F+1)$ time levels. Thus, on each point, F large linear systems are solved. Provided that the selected number of subspaces r is small ($r \ll F$), consequently only r large and F reduced linear systems have to be solved, which strongly reduces the computational costs.*

Modal Coordinate Reduction An alternative way to realise MOR in the context of the application discussed is the MCR technique, cf. Section 4.3, which is based on the eigenvalue decomposition of the underlying system matrix L of the original system (5.41). The concept of MCR is to transform the full model from physical coordinates in physical space to modal coordinates in modal space by using the eigenvectors of L that are usually put together to form column by column an eigenvector matrix. Subsequently, those modes are removed that have less important contributions to the system responses. In general, only a few modes have a significant impact on the system dynamics within the frequency range of interest.

It should be noted that the use of the MCR method in the field of computer graphics is not new, cf. [141, 213], and is also often used, e.g. for the physical simulation of fluid-object interaction [274], realistic sound rendering [57] and deformable objects [131, 258, 290]. Nevertheless, we show how to define a computational framework by MCR that yields efficient time integration linked with accurate shape signatures. Within the construction of our framework some technical improvements in Section 5.6 are introduced that in this framework turns out to be highly beneficial for this approach. The following describes how to apply MCR for the PSC framework.

The application of a regular *modal* transformation $\boldsymbol{u} = \Phi \boldsymbol{z}$ to the system (5.41), where $\Phi \in \mathbb{R}^{N \times N}$ is the unit eigenvector matrix of L, results in

$$\Phi \dot{\boldsymbol{z}}(t) = L \Phi \boldsymbol{z}(t) \tag{5.49}$$

Then the multiplication from the left by D and $\Phi^\top$ yields

$$\Phi^\top D \Phi \dot{\boldsymbol{z}}(t) = \Phi^\top D L \Phi \boldsymbol{z}(t) \tag{5.50}$$

From (5.35) we have $I = \Phi^\top D \Phi$ and $\Lambda = \Phi^\top C \Phi$, which due to $L = D^{-1} C$ also implies that $\Lambda = \Phi^\top D L \Phi$. Inserting the last identities in (5.50) finally leads to

$$\dot{\boldsymbol{z}}(t) = \Lambda \boldsymbol{z}(t) \tag{5.51}$$

The latter equation is the starting point for choosing eigenvalues and eigenvectors and explicitly clarifies in this way the affiliation to the class of *spectrum-based methods*.

It is well-known that the low frequencies which correspond to small eigenvalues are supposed to dominate the dynamics of the system. Suppose $r \ll N$ ordered eigenvalues $0 = |\lambda_1| < |\lambda_2| \leq \cdots \leq |\lambda_r|$ are of interest. Consequently, we obtain with $\Lambda_r \in \mathbb{R}^{r \times r}$ and

$\Phi_r \in \mathbb{R}^{N\times r}$ extracted from Λ and Φ, respectively, the *reduced model* of order r given by

$$\dot{\boldsymbol{z}}_r(t) = \Lambda_r \boldsymbol{z}_r(t) \qquad \text{where} \qquad \boldsymbol{z}_r = \Phi_r^\top D\boldsymbol{u} \tag{5.52}$$

This low dimensional and decoupled ODE system is much faster to solve than the original one. Applying the IE method to (5.52) leads to

$$(I_r - \tau\Lambda_r)\, \boldsymbol{z}_r^{k+1} = \boldsymbol{z}_r^k, \quad \boldsymbol{z}_r^0 = \Phi_r^\top D\boldsymbol{u}^0 \tag{5.53}$$

or otherwise to

$$\boldsymbol{z}_r^{k+1} = P\boldsymbol{z}_r^k, \quad \boldsymbol{z}_r^0 = \Phi_r^\top D\boldsymbol{u}^0 \tag{5.54}$$

with the simple matrix inversion

$$P = (I_r - \tau\Lambda_r)^{-1} = \operatorname{diag}\left(\frac{1}{1-\tau\lambda_1}, \ldots, \frac{1}{1-\tau\lambda_r}\right) \tag{5.55}$$

The reduced system (5.54) is based solely on diagonal matrices and can easily be solved by sparse matrix-vector multiplications.

To summarise, the computational costs only depend on the eigenvalues and eigenvectors of L, which are known to be computationally intensive to obtain. Based on the above considerations, only the smallest eigenvalues have to be computed iteratively so that the MCR technique is practicable for a convenient number of modes. Apart from that the MCR method constructs the projection matrix $V = \Phi_r$ only once independent of the corresponding initial condition, which therefore differs from the KSMOR method.

Remark 5.10. *Obviously, the MCR method is for physical simulations with a very large basis r of several hundreds or thousands of modes not practical. However, we will show that in all of our experiments a sufficiently small number of modes used are suitable to compute numerical signatures accurately enough.*

Connection to HKS: The reduced system (5.52) can also be solved exactly. It is well-known, that the solution of the linear ODE in modal coordinates is given by

$$\boldsymbol{z}_r(t) = e^{\Lambda_r t}\boldsymbol{z}_r(0) \tag{5.56}$$

or in physical coordinates as

$$\boldsymbol{u}(t) \approx \Phi_r \boldsymbol{z}_r(t) = \Phi_r e^{\Lambda_r t}\Phi_r^\top D\boldsymbol{u}(0) = \sum_{i=1}^{r} e^{\lambda_i t}\boldsymbol{\phi}_i\boldsymbol{\phi}_i^\top D\boldsymbol{u}(0) \tag{5.57}$$

The latter formulation corresponds to the raw[4] version of the HKS method which uses the analytical solution of the continuous heat equation expressed as a series expansion. This is not surprising, as the series is based on the eigenfunction expansion and is consequently the analytical counterpart of MCR built on the eigendecomposition of the discrete Laplacian.

[4] Let us stress that in order to make the series expansion techniques HKS effective, it is recommended to employ heuristics such as the scaling of time [267].

However, we consider the MCR method to be more beneficial. On the one hand, this technique is based on a decoupled ODE system which can be solved exactly in a more simple manner contrary to the approach of separation when dealing with the continuous PDEs. Otherwise, it does not generally require any knowledge of the analytical solution and can be applied if the semi-discretised model is available. As a result, MCR is much more comfortable due to its flexibility.

5.5 Comparison of Implicit Euler Solvers Based on Two Datasets

As seen before, the temporal integration for (5.18) or (5.19) is done using the IE method which requires to solve systems of linear equations. For the latter task exists various numerical solvers, that have different properties in terms of computational effort and accuracy of the computed solution which can significantly influence a shape matching task. For this reason, the numerical solvers and their performance in terms of matching quality and CPU time are analysed and evaluated at hand of two different datasets. The experiments are only evaluated for the geometric heat equation (5.18), analogous results have been achieved for (5.19) in undocumented tests. As methods of choice we consider:

1. Sparse direct solver. The internal function *factorize* included in the SuiteSparse package is employed.

2. Sparse iterative solver. The user-defined parameters $\varepsilon > 0$ and l related to the tolerance of the relative residual (2.136) and the maximum number of iterations, respectively, are used to terminate the CG algorithm. Furthermore, the MIC preconditioner is being investigated for large systems. In this case, a numerical fill-in strategy is used where the associated parameter is fixed[5] to $\gamma = 10^{-3}$.

3. KSMOR method. The solver can be tuned by the number of projection subspaces $\mathcal{K}_r$ used. For computing the Krylov subspace $V = \mathcal{K}_r((L - \sigma I)^{-1}, (L - \sigma I)^{-1}\boldsymbol{u}^{i,0})$ the sparse direct solver is applied. In addition, different values for σ are investigated, more precisely $\sigma \approx 0$ and $\sigma = \infty$.

4. MCR method. The performance of the solver can be tuned by the number of eigenvalues and eigenvectors used, here called modes r. For a fair comparison between the solvers used, we apply MCR combined with the IE method as developed in (5.54).

As indicated, we are particularly interested in CPU time as well as actual accuracy of the results related to shape matching. In order to evaluate the accuracy of the methods a dense point-to-point correspondence is performed, involving all vertices the shapes are made off. In detail, the experiments are evaluated as follows.

Discrete Feature Descriptor The discrete version of the introduced feature descriptor (5.6) is generated by numerical time integration of the underlying geometric PDE. Therefore, the time axis $[0, t_F]$ is subdivided using $(F + 1)$-time levels into $0 := t_0 < t_1 < \cdots < t_F$ with

[5] In practice, cf. [16, 42] or the Chapter 6, a good performance is obtained for values of $\gamma \in [10^{-4}, 10^{-2}]$.

$F = \frac{t_F}{\tau}$, where the time step size τ is uniformly chosen. The corresponding discrete feature descriptors are of the form

$$f_{\boldsymbol{x}_i}(t) := u(\boldsymbol{x}_i, t)_{|_{t=t_k}}, \quad k = \{0, 1, \ldots, F\} \tag{5.58}$$

for $i = 1, \ldots, N$, meaning that $f_{\boldsymbol{x}_i}(t) \in \mathbb{R}^{F+1}$. The computation of a discrete feature descriptor at $\boldsymbol{x}_i$ for a given shape requires solving F sparse large linear systems of size $N \times N$. Consequently, the computation of all $f_{\boldsymbol{x}_i}(t)$ implies that one has to solve in total $N \cdot F$ linear systems for each shape.

Hit Rate The percentage hit rate is defined as $TP/(TP + FP)$, where TP and FP are the number of true positives and false positives, respectively.

Geodesic Error For the evaluation of the correspondence quality, we follow the Princeton benchmark protocol [148]. This procedure evaluates the precision of the computed matchings $\boldsymbol{x}_i$ by determining how far are those away from the actual ground-truth correspondence $\boldsymbol{x}^*$. Therefore, a normalised intrinsic distance $d_{\mathcal{M}}(\boldsymbol{x}_i, \boldsymbol{x}^*)/\sqrt{A_{\mathcal{M}}}$ on the transformed shape is introduced. Finally, a matching is accepted to be true if the normalised intrinsic distance is smaller than the threshold 0.25, as illustrated in Figure 5.8.

Dataset For the experimental evaluation, datasets at two different resolutions are compared, namely small and large. The small ($N = 4344$) shapes of the *wolf* are taken from the TOSCA dataset [50]. The *baby* shapes have a large resolution ($N = 59727$) and are taken from the KIDS dataset [231]. The datasets are available in the public domain, examples of it are shown in Figure 5.9. All shapes provide ground-truth, and degenerated triangles were removed.

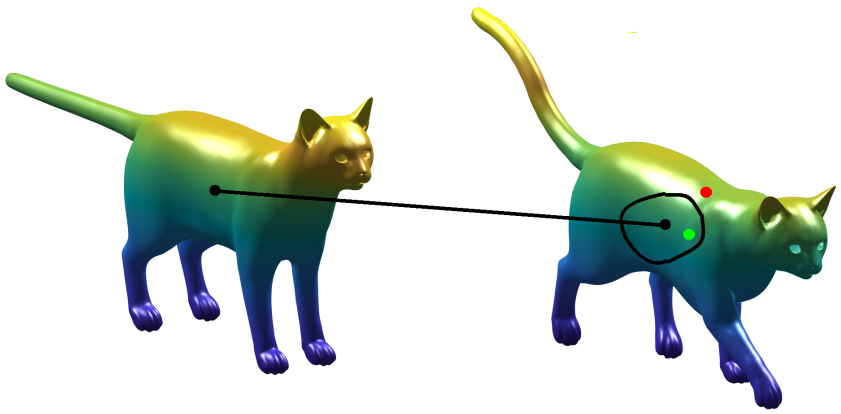


Figure 5.8: Evaluation of PSC using the geodesic error. The (correct) ground truth matching is visualised by the black line. On the transformed shape we allow a certain radius of tolerance around the ground truth point. Matchings within the radius (e.g. the green point) are considered to be correct while points outside of the radius (e.g. the red point) are considered as outliers.

wolf $N = 4344$ | *baby* $N = 59727$

Figure 5.9: For experimental evaluation, shapes at two different resolutions are compared, namely small and large. These are represented by the dataset *wolf* and *baby*, taken from the TOSCA [50] and KIDS [231] dataset, respectively.

General Parameters For this experiment, we set the stopping time and the uniform time increment to $t_F = 25$ and $\tau = 1$, respectively. The parameters are chosen without a fine-tuning, as we are interested to figure out the differences of the numerical methods compared to accuracy and computational costs. In this context, some studies on setting the diffusion time have been done in [67, 68].

Another issue when using time integration methods is the choice of the initial condition $u(\boldsymbol{x}, 0) = u_0(\boldsymbol{x})$. As mentioned in Section 5.3.2, we employ here a discrete Dirac delta peak in the form of

$$\boldsymbol{u}^{i,0} = \Big(0, \dots, 0, |\Omega_i|^{-1}, 0, \dots 0\Big)^\top \tag{5.59}$$

Finally, besides the specification $\sigma = \infty$, the expansion point for constructing the Krylov subspace $V = \mathcal{K}_r((L - \sigma I)^{-1}, (L - \sigma I)^{-1}\boldsymbol{u}^{i,0})$ is fixed to $\sigma = 0.1$ (without loss of generality).

Computational Aspects All experiments were done in MATLAB R2018b with an *Intel(R) Xeon(R) CPU E5-2609 v3*. The CPU times presented incorporate the precomputation (factorisation, preconditioning, eigendecomposition, reduction) and the numerical resolution, so that the performances are therefore easily comparable. The eigenvalues and eigenvectors for MCR are computed by the MATLAB internal function *eigs*.

We also note that the computations were taken using the *Parallel Computing Toolbox* integrated in MATLAB. As already indicated, all methods have to solve the geometric heat equation for each point of the given shape independently. This step can easily be parallelised by distributing the code to 6 workers using the *parfor* loop.

Remark 5.11. *The two datasets used were selected because they are useful for demonstrating in detail the described behaviour of the solvers as it can be observed in all tests we performed. They show the typical range of results that one obtains in terms of quality and efficiency.*

5.5.1 Experimental Results

In what follows we compare the four important numerical approaches at hand of the two selected test datasets.

Results on Sparse Direct Solver First, we consider the *wolf* dataset with a small point cloud size of only $N = 4344$ points for each of the two shapes, cf. Figure 5.9. In order to compute the discrete feature descriptors, the geometric heat equation has to be solved numerically once for each point and on each shape. This ultimately results in dealing with 217200 sparse linear systems (5.37) of size $N \times N$. Solving the geometric heat equation via the sparse direct solver gains very good results by around 91.6% matching performance. By using the LU factorisation provided by MATLAB, the CPU time with around 600 seconds causes significant computational costs and is extremely inefficient. However, the computational costs can be reduced to around 25 seconds with the powerful SuiteSparse package, which generates the same geodesic error accuracy.

For the *baby* dataset with $N = 59727$ points on each shape, it is necessary to solve in total 2986350 sparse linear systems. The direct solver combined with the SuiteSparse package performs this task in exactly 9267 seconds ($\approx 2\ 1/2$ hours) and yields a matching performance of around 48%. At this point it is recognisable, that large datasets cause high computational costs and that the direct solver appears to be impractical for such shape matching applications.

The results (CPU time and geodesic error) of the direct solver including SuiteSparse factorisation are used in the further course for a comparison with the remaining solvers.

Results on Sparse Iterative Solver Solving the sparse linear systems (5.38) with the CG method involves setting the stopping criterion of the algorithm. In case of the *wolf* dataset, increasing ε leads to a faster CG approximation, but it turns out that the accuracy remains almost unchanged even for the relatively large value $\varepsilon = 10^{-1}$, cf. on the top row in Figure 5.10. This implies, a good representation of the shape signature is achieved already after a small number of iterations. In addition, the repeated experiment for $\varepsilon = \{10^{-1}, 10^{-2}\}$ and various maximum numbers l of CG iterations, i.e. the iterative scheme terminates if one of the conditions is satisfied, is also shown on the bottom row in Figure 5.10. In this case, the reduction of l has only a minor influence on the geodesic error accuracy, but leads to a fast CPU time of a few seconds with almost equally matching performance as the direct solver.

Regarding the *baby* dataset the matrix size should be taken into account which generally affects the convergence rate of the CG algorithm. Therefore, we first analyse the PCG method including MIC(10^{-3}) as shown on the top row in Figure 5.11. Compared to the performance of the direct solver no improvement is achieved. Furthermore, a closer examination of the iterations required to terminate the iterative algorithm shows that PCG needed just about $l = 1$ iteration. As a result, PCG cannot be further tuned in terms of its performance. The latter observation again inspires the idea to perform the CG method for a very small number of iterations l, which accordingly should be sufficient to gain acceptable results in fast CPU time. The results of the CG method for $\varepsilon = 10^{-1}$ and $l \leq 10$ are shown on the bottom row in Figure 5.11. Also in this case, the iterative solver can reduce the computational effort significantly up to 80%, but the percentage deviation of the accuracy in relation to the direct solver is approximately up to 10%.

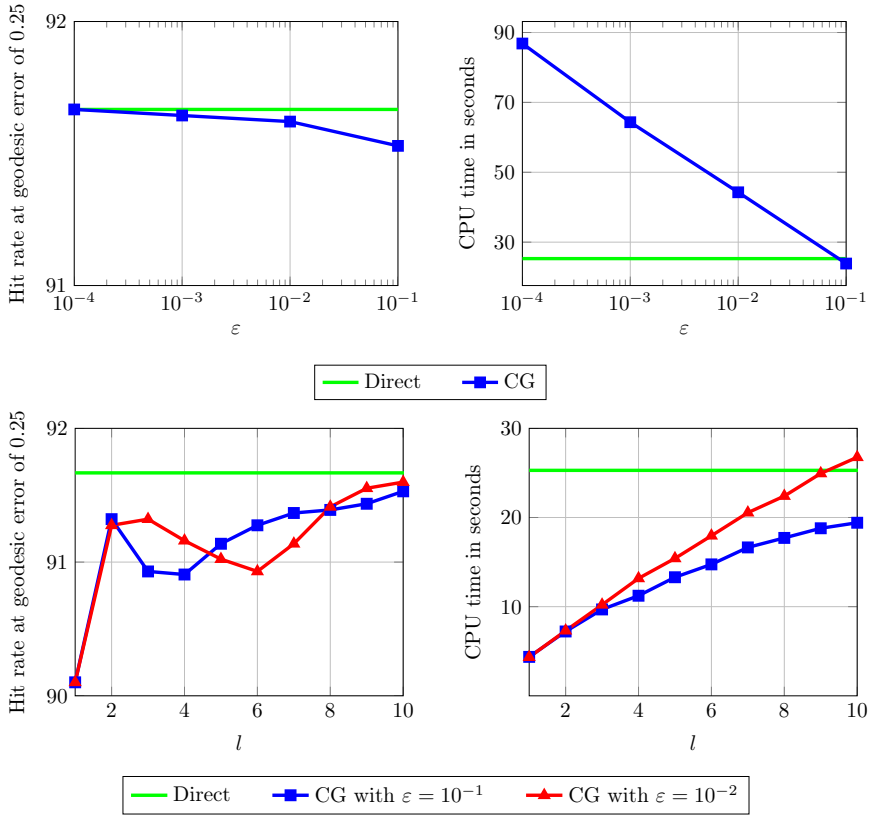


Figure 5.10: Results for the *wolf* dataset using the geometric heat equation. We compare the geodesic error at 0.25 (**left**) and the CPU time (**right**) between the direct solver and the CG method for various ε (**top**) and different number of CG iterations l (**bottom**) for $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-2}$. The latter means, the iterative scheme terminates if one of the conditions is satisfied.

Results on KSMOR First, let us discuss the results of the KSMOR method using $\sigma = \infty$ for the *wolf* dataset illustrated on the top row in Figure 5.12. As expected, a good matching performance is already obtained for a small number r of Krylov subspaces in an additionally very fast CPU time. Unfortunately, a large number r is needed to converge towards the same geodesic error accuracy compared to the direct solver.

For the *baby* dataset, however, the numerical signatures are only accurate enough for larger numbers r as a consequence of approximating a higher frequency spectrum within the KSMOR technique, see on the bottom row in Figure 5.12. Nevertheless, the computational costs can be highly reduced up to 99%, whereby the percentage deviation of the accuracy in relation to the direct solver is approximately up to 10%.

The results of KSMOR using $\sigma = 0.1$ for the *wolf* dataset are illustrated on the top row in Figure 5.13. As expected, increasing the dimension of Krylov subspaces r are leading to a more accurate approximation, whereby the same geodesic error accuracy compared to the direct solver can already be achieved for $r \leq 10$. The computational costs required correspond to those of CG, but we obtain a higher accuracy in terms of the geodesic error.

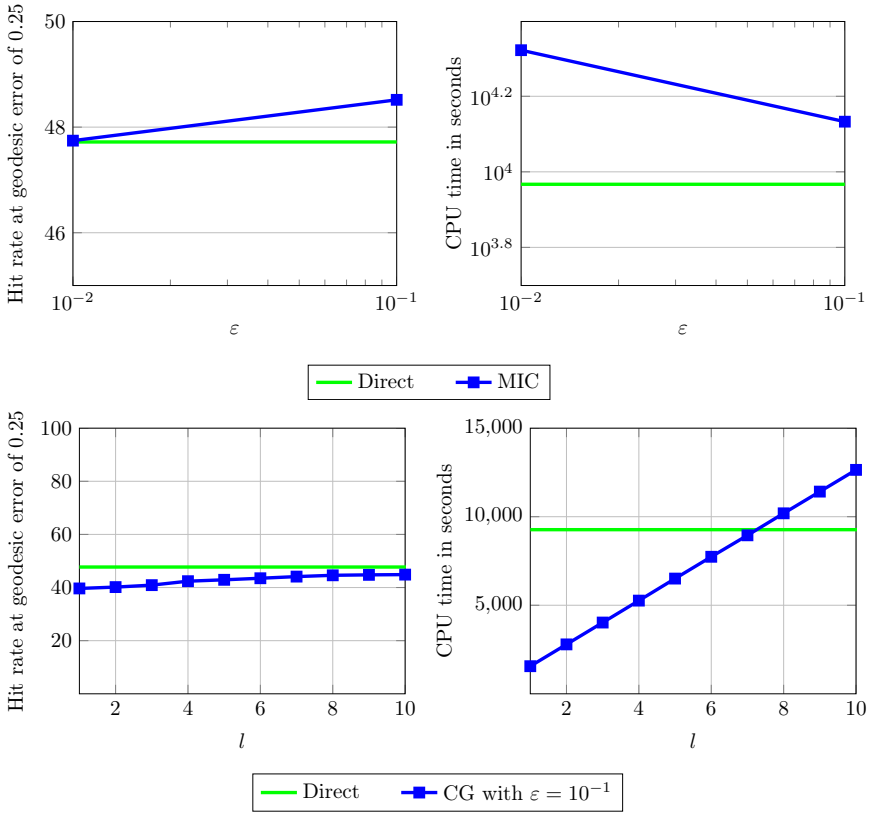


Figure 5.11: Results for the *baby* dataset using the geometric heat equation. We compare the geodesic error at 0.25 (**left**) and the CPU time (**right**) between the direct solver and the PCG method, including MIC(10^{-3}), for $\varepsilon = 10^{-1}$ and $\varepsilon = 10^{-2}$ (**top**). In addition, we present a comparison between the direct method and the CG method (**bottom**) for $\varepsilon = 10^{-1}$ and the first ten CG iterations l. The latter means, the iterative scheme terminates if one of the conditions is satisfied.

The same performance output is obtained by applying KSMOR for the *baby* dataset, cf. on the bottom row in Figure 5.13. Also for this dataset, the use of approximately $r = 1$ is deemed sufficient to obtain an excellent trade-off between quality and efficacy and can save around 95% of the CPU time in relation to the direct solver.

Overall, the best performance in the context of the KSMOR technique is achieved when $\sigma \approx 0$ is used to capture the slow dynamics. However, we note that the CPU time for $r = 1$ with around 420 seconds is still relatively high for this standard size shape.

Remark 5.12. *A further improvement in efficiency may be the coupling of KSMOR with the CG method when constructing the Krylov subspaces. Due to the fact that r can be kept very small the use of an equally small tolerance value ε is deemed be useful, since the accumulated projection error remains negligible within the few iteratively constructed subspaces.*

Results on MCR Finally, we examine the MCR technique. For the experiments the number of ordered modes used are increased, starting from $r = 5$ and going up to $r = 1000$. Regarding

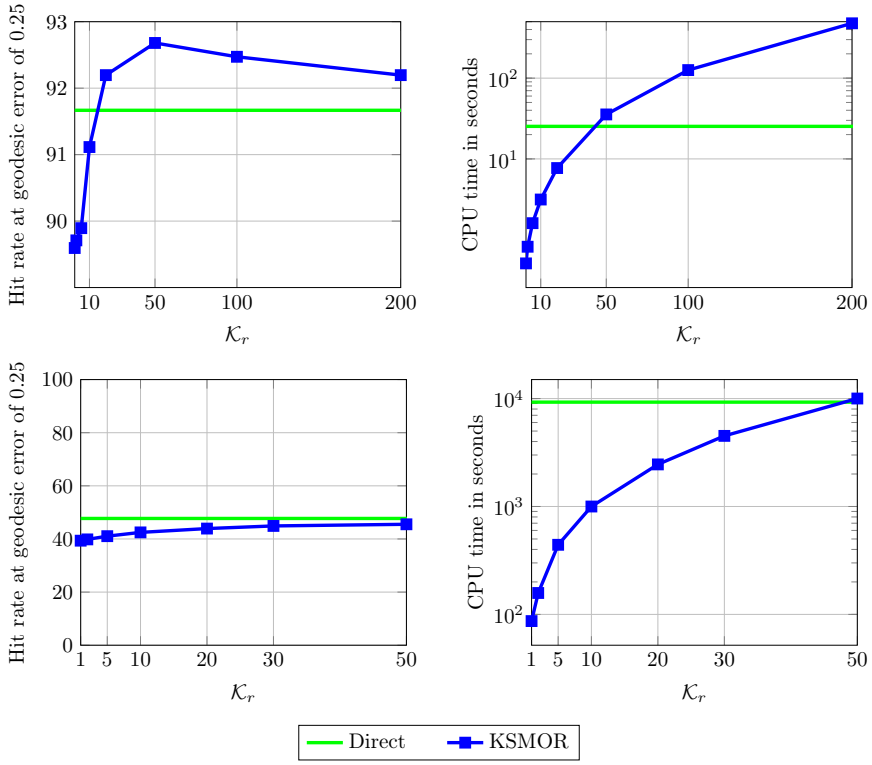


Figure 5.12: Results for the dataset *wolf* (**top**) and *baby* (**bottom**) using the geometric heat equation. We compare the geodesic error at 0.25 (**left**) and the CPU time (**right**) between the direct solver and KSMOR method with $\sigma = \infty$ for a different number of Krylov subspaces $\mathcal{K}_r$. The construction of the Krylov subspace V is only based on sparse matrix-vector multiplications.

to the *wolf* dataset we would expect the correspondence quality improve as the number of modes is increased. However, the evaluation on the top row in Figure 5.14 does not show this desirable behaviour. The outcome for a small spectrum $r \approx 10$ is much better than for a large spectrum[6] $r \approx 1000$, which seems at first glance not intuitive. Nevertheless, let us emphasise that the CPU time is incredibly fast when using a small number of modes. For values up to $r = 100$ the approximate solutions are computed in less than 1 second, but the obtained matching performance is far too low.

Applying the MCR technique to the *baby* dataset yields in total a similar behaviour, but we also observe a surprising result, see on the bottom row in Figure 5.14. In this experiment again, a small spectrum leads to better results compared to larger numbers of modes. In contrast to the *wolf* dataset, two notable observations can be made. First, using $r = \{5, 10\}$ even outperforms the geodesic error accuracy in relation to the direct solver and, second, the matching performance of both solvers are nearly equal already for $r = 50$. In this case, MCR has an extremely short CPU time of around 10 seconds, which reduces the computational effort compared to the direct solver by around 99.9%.

[6] The geodesic error converges towards the solution of the direct method beginning from about $r = 2000$ which is an excessive high number of modes for the MCR approach.

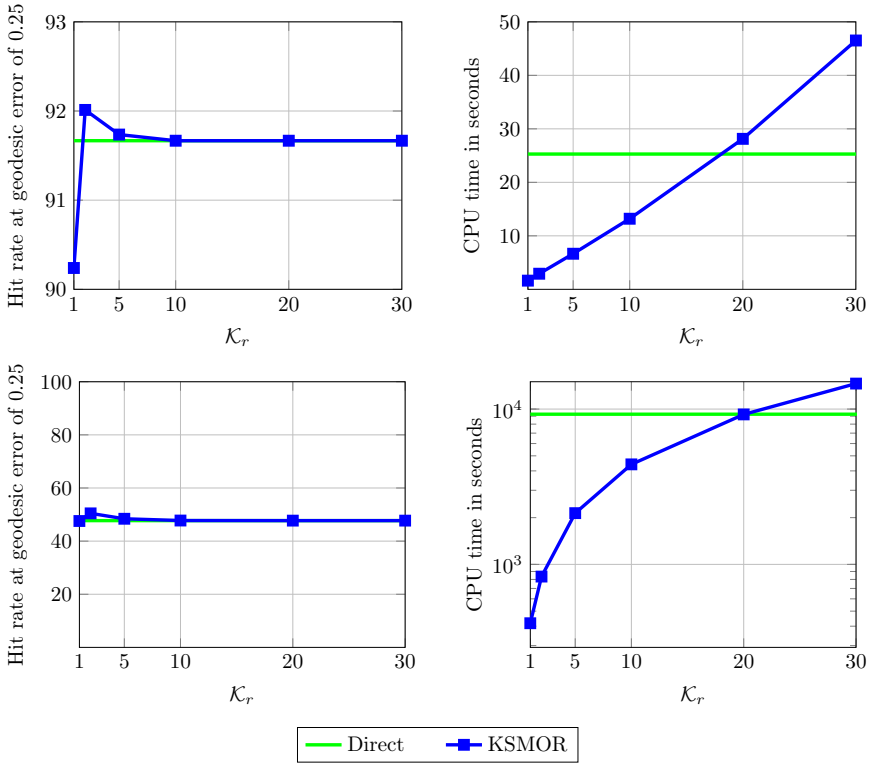


Figure 5.13: Results for the dataset *wolf* (**top**) and *baby* (**bottom**) using the geometric heat equation. We compare the geodesic error at 0.25 (**left**) and the CPU time (**right**) between the direct solver and KSMOR method with $\sigma = 0.1$ for a different number of Krylov subspaces $\mathcal{K}_r$. The construction of the Krylov subspace V requires the solution of sparse linear systems.

5.5.2 Discussion of the Solvers

The experiments based on the two datasets illustrated the performances of all the applied solvers. The direct solver generally produces the best results in relation to the geodesic error accuracy. For high resolution shapes ($N > 50000$), however, the method is quite inefficient and causes computational costs amounting to several hours.

In contrast, the CG method may reduce the computational costs by around 80%, whereby the percentage deviation of the geodesic error accuracy in relation to the direct method is approximately less than 10%. As was to be expected, a preconditioning is not useful for the considered PSC application.

By using KSMOR technique, even a better performance is achieved, where $\sigma \approx 0$ is the better option here. In this study, the choice of just $r = 1$ saves around 95% of the CPU time, whereby the percentage deviation of the matching quality compared to the direct solver is only around 1%. Although the efficiency of KSMOR as spectrum-free approach is superior compared to the direct and iterative solvers, its computational costs are still relatively high as the projection matrices have to be reconstructed at each point. In contrast, the main advantage of the KSMOR technique is to compute a factorisation (in the case $\sigma \neq \infty$) of $L - \sigma I$ only once.

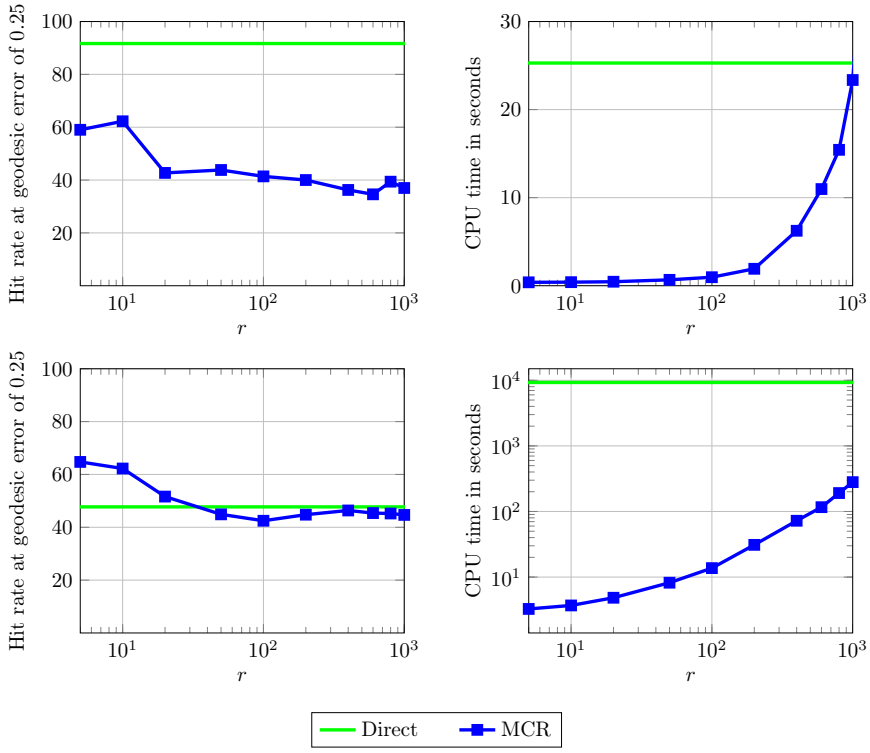


Figure 5.14: Results for the dataset *wolf* (**top**) and *baby* (**bottom**) using the geometric heat equation. We compare the geodesic error at 0.25 (**left**) and the CPU time (**right**) between the direct solver and the MCR technique for different number of modes $r \in [5, 1000]$.

The computational effort of the MCR method grows exponentially (by increasing r) and accordingly a practicable value r should be small. It is remarkable that the matching performance for a small spectrum ($r \approx 10$) is even better to the ones for a significantly larger spectrum ($r \approx 1000$). In case of using a small spectrum, the MCR method is highly efficient and can save around 99.9% CPU time, which suggests that this technique is favourable for solving shape matching by time integration.

Due to the incredible power of MCR, we follow this technique within this chapter. In this context, an interesting aspect would be to tune the method so that it achieves a more stable and higher matching performance, e.g. when using it with shapes such as the *wolf* shape.

5.6 Optimised MCR Signatures

As a result of the last section, the MCR method appears favourable for computing the numerical shape signature, especially considering the extremely low computational costs when using a small number of modes. Remarkably, the matching performance based on the geodesic error is highly dependent on the number of modes used. For this reason, the computation of the MCR signature leads to new challenges, as the aim is to ensure in a reliable way a high quality of results without additional computational costs.

In the following we introduce three technical approaches in order to enhance matching

robustness and quality. First, we discuss the calculation of the eigendecomposition again in order to prevent numerical instabilities. Second, we propose a rescaling of the integration domain $[0, t_F]$, and third the initial condition is adapted to obtain a more unique signature.

It should be emphasised that the adaption of the integration domain is depending on the underlying geometric PDE. In a first step we are therefore going into the geometric heat equation in more detail. Then the geometric wave equation is dealt with. Lastly, we will analyse the essential differences of the MCR technique in terms of solving the decoupled reduced ODE system exactly and numerically.

5.6.1 Stable Eigenvalue Computation

An important aspect of the MCR method is the way how the eigenvalues λ and eigenvectors $\boldsymbol{\phi}$ of the system matrix $L = D^{-1}C$ are computed. As described in Section 5.4.1, the computation is performed by solving the GEP (5.33), which uses certain properties of the underlying matrices D and C to generate numerically real-valued eigenvalues and eigenvectors. However, the computation in this way may exhibit traces of instability in numerical practice and strongly affects the matching performance of the MCR signatures.

Another more favourable approach is to transform $C\boldsymbol{\phi} = \lambda D\boldsymbol{\phi}$ into the standard eigenvalue problem without loss of the symmetry of C, so that more robust numerical eigensolvers can be used. The *symmetric eigenvalue problem (SEP)* can be achieved by a similarity transformation in the following way:

$$L = D^{-1}C = D^{-\frac{1}{2}}D^{-\frac{1}{2}}CD^{-\frac{1}{2}}D^{\frac{1}{2}} = D^{-\frac{1}{2}}BD^{\frac{1}{2}} \tag{5.60}$$

where the matrix $B = D^{-\frac{1}{2}}CD^{-\frac{1}{2}}$ is symmetric due to the symmetry of C. Thus, the matrices L and B are similar, have the same real eigenvalues and an eigenpair $(\lambda, \boldsymbol{\phi})$ of $B\boldsymbol{\phi} = \lambda\boldsymbol{\phi}$ corresponds to an eigenpair $(\lambda, \widetilde{\boldsymbol{\phi}}) = (\lambda, D^{-\frac{1}{2}}\boldsymbol{\phi})$ of $L\widetilde{\boldsymbol{\phi}} = \lambda\widetilde{\boldsymbol{\phi}}$. The matrix $D^{-\frac{1}{2}}$ in (5.60) is well-defined due to the positive diagonal matrix D^{-1} and is also positive diagonal. The eigenvectors $\boldsymbol{\phi}$ of B are orthonormal with $\boldsymbol{\phi}_i^\top \boldsymbol{\phi}_j = \delta_{i,j}$ so that the eigenvectors of L are D-orthogonal

$$\boldsymbol{\phi}_i^\top \boldsymbol{\phi}_j = \left(D^{\frac{1}{2}}\widetilde{\boldsymbol{\phi}}_i\right)^\top D^{\frac{1}{2}}\widetilde{\boldsymbol{\phi}}_j = \widetilde{\boldsymbol{\phi}}_i^\top D\widetilde{\boldsymbol{\phi}}_j = \delta_{i,j} \tag{5.61}$$

A comparison of the numerical computation of eigenvalues with GEP and SEP is visualised in Figure 5.15. This test clearly demonstrates that the eigenvalues computed via GEP are not numerically robust, since the first twenty dominant eigenvalues are not identical when computing the eigendecomposition for $r = \{20, 50\}$. In contrast, SEP provides stable results. Let us comment on an important issue here that may arise when adapting our framework to different types of spatial discretisations such as the prominent finite element method which is one of the frequently used methods for numerical computations. Obviously, if D^{-1} is not a diagonal matrix, computing $D^{-\frac{1}{2}}$ is not a viable option. Nevertheless, the proposed transformation is applicable if D^{-1} is symmetric positive definite. This situation arises, for instance, when the finite element ansatz is employed instead of the presented finite volume set-up in Section 5.3.2, whereby D^{-1} represents the *mass* matrix in that setting cf. [302]. Then $C\boldsymbol{\phi} = \lambda D\boldsymbol{\phi}$ can be reformulated using the existing Cholesky factorisation $D = LL^\top$,

with a lower triangular matrix L containing positive diagonal entries, as

$$\begin{aligned} C\boldsymbol{\phi} = \lambda D\boldsymbol{\phi} \quad &\Leftrightarrow \quad C\boldsymbol{\phi} = \lambda LL^\top \boldsymbol{\phi} \\ &\Leftrightarrow \quad L^{-1}C\boldsymbol{\phi} = \lambda L^\top \boldsymbol{\phi} \\ &\Leftrightarrow \quad L^{-1}CL^{-\top}L^\top \boldsymbol{\phi} = \lambda L^\top \boldsymbol{\phi} \\ &\Leftrightarrow \quad L^{-1}CL^{-\top}\widetilde{\boldsymbol{\phi}} = \lambda \widetilde{\boldsymbol{\phi}} \\ &\Leftrightarrow \quad B\widetilde{\boldsymbol{\phi}} = \lambda \widetilde{\boldsymbol{\phi}} \end{aligned} \tag{5.62}$$

with $B = L^{-1}CL^{-\top}$ and $\widetilde{\boldsymbol{\phi}} = L^\top \boldsymbol{\phi}$.

Based on this framework, we analyse the abovementioned computation (5.60) of the eigenpairs $(\lambda, \boldsymbol{\phi})$ using datasets *wolf* and *baby*. The evaluation by solving the GEP and the SEP is visualised in Figure 5.16. As indicated above, we observe that the ansatz of using the similarity transformation for tackling the eigenvalue problem achieves significantly more stable matching results. A stable eigenvalue computation is therefore essential when using MCR signatures for the PSC application.

Remark 5.13. *The use of this stability improvement obviously shows the accuracy behaviour, especially in the case of the wolf shape, that we would have expected when increasing the number of modes, which indicates a better robustness of the approach. A quality improvement for the baby shape is also be obtained.*

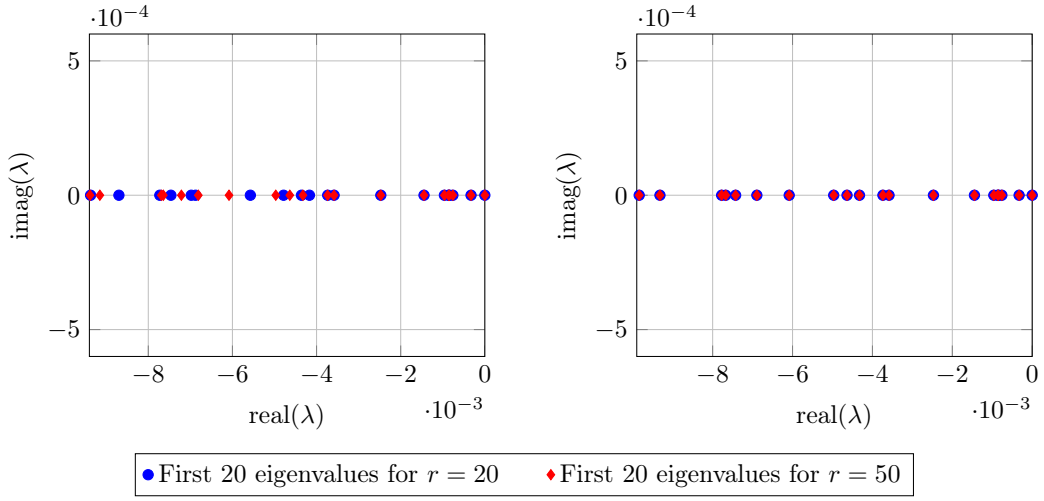


Figure 5.15: Comparison of the numerical eigenvalue problem computation on the *wolf* dataset with GEP (5.33) and SEP (5.60). The eigenvalues are computed by the MATLAB R2018b internal function *eigs*. Here we illustrate the first twenty eigenvalues when computing the first $r = 20$ and $r = 50$ dominant (smallest) eigenvalues of the underlying discrete Laplacian. **Left:** Eigenvalues computed with GEP. **Right:** Eigenvalues computed with SEP. Usually, the eigenvalues should remain identical regardless of the computation of a certain number of dominant eigenvalues. In contrast to SEP, however, GEP generates no stable eigenvalue calculations.

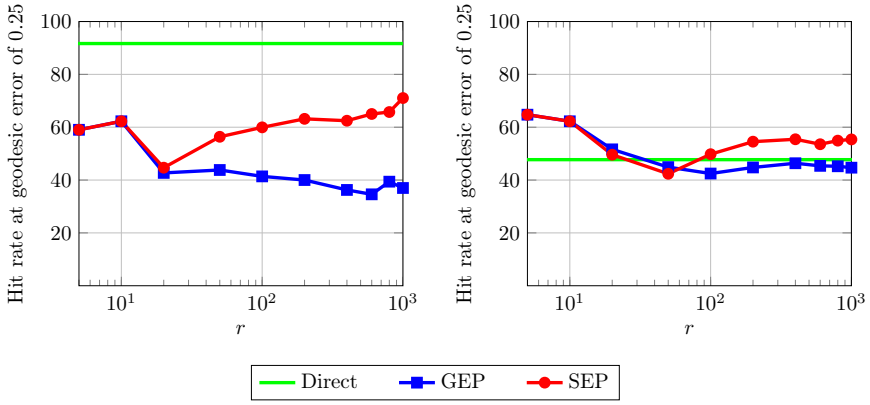


Figure 5.16: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the geometric heat equation. We compare the geodesic error at 0.25 between the direct solver and the MCR method for different number of modes $r \in [5, 1000]$. The eigenvalue problem is computed by solving GEP (5.33) and SEP (5.60). Using the similarity transformation achieves better matching results.

Remark 5.14. *The mentioned similarity transform is well known, cf. [226, 302]. However, GEP is often used for computing the eigendecomposition, see [14, 51, 131, 199, 226, 227, 235, 267]. As shown, for example, when using the MCR technique for the PSC application in this chapter, the implementation of solving GEP and SEP numerically should not be underestimated in terms of stable eigenvalue computations. Whether this transformation must always be performed is not known and depends on the software used. This is essential when using MATLAB 2018b.*

5.6.2 Improved Scaling of the Integration Domain

We now discuss the effect of the choice of the parameter t_F with respect to the MCR technique. In principle, the computed numerical signatures are constructed over a temporal domain $[0, t_F]$, which makes them easily measurable for the shape matching purposes. Nonetheless, we show that it pays off to extend the considered time scale in the numerical MCR signature. In doing so, we describe how the use of a small spectrum affects the characteristics of the feature descriptor and thus the matching performance.

MCR Heat Signature In what follows we focus on solving the decoupled reduced ODE system (5.56) exactly. Of course, the same reasoning is also true when the reduced system is solved numerically, e.g. using the IE scheme.

Let us start our discussion of time scaling by considering the basic idea for motivation. The analytical solution of the (coupled) semi-discrete geometric heat equation $\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t)$ is given in the form

$$\begin{aligned}\boldsymbol{u}(t) = e^{Lt}\boldsymbol{u}(0) &\overset{(5.35)}{=} e^{\Phi\Lambda\Phi^\top Dt}\boldsymbol{u}(0) = \Phi e^{\Lambda t}\Phi^\top D\boldsymbol{u}(0) \\ &= \sum_{i=1}^{N} e^{\lambda_i t}\boldsymbol{\phi}_i\boldsymbol{\phi}_i^\top D\boldsymbol{u}(0) = \sum_{i=1}^{N} \boldsymbol{\phi}_i^\top D\boldsymbol{u}(0) e^{\lambda_i t}\boldsymbol{\phi}_i \end{aligned} \tag{5.63}$$

where $\boldsymbol{\phi}_i^\top D\boldsymbol{u}(0)$ is a scalar. Therefore, the pointwise feature descriptor results in

$$f_{\boldsymbol{x}_j}(t) := u(\boldsymbol{x},t)_{|\boldsymbol{x}=\boldsymbol{x}_j} = \left(\sum_{i=1}^{N} \boldsymbol{\phi}_i^\top D\boldsymbol{u}_j(0) e^{\lambda_i t}\boldsymbol{\phi}_i\right)^\top \hat{\boldsymbol{e}}_j = \left(\sum_{i=1}^{N} \boldsymbol{\phi}_i^\top \hat{\boldsymbol{e}}_j e^{\lambda_i t}\boldsymbol{\phi}_i\right)^\top \hat{\boldsymbol{e}}_j \tag{5.64}$$

with $D\boldsymbol{u}_j(0) = \hat{\boldsymbol{e}}_j$, where $\hat{\boldsymbol{e}}_j$ is the j-th unit vector. The representation of the solution (5.64) describes an exponential decay of heat transferred away from the considered point $\boldsymbol{x}_j$. Using only a small number of r dominant frequencies (small eigenvalues) of a system, the reduced basis solution as an approximative feature descriptor is given by

$$f_{\boldsymbol{x}_j}(t) \approx \widetilde{u}(\boldsymbol{x}_j,t) = \left(\sum_{i=1}^{r} \boldsymbol{\phi}_i^\top \hat{\boldsymbol{e}}_j e^{\lambda_i t}\boldsymbol{\phi}_i\right)^\top \hat{\boldsymbol{e}}_j \tag{5.65}$$

Furthermore, the reduced solution (5.65) with eigenvalues $0 \leq |\lambda_i| \ll 1$ $(i = 1,\ldots,r)$, can be simplified for small times t as

$$\widetilde{u}(\boldsymbol{x}_j,t) = \left(\sum_{i=1}^{r} \boldsymbol{\phi}_i^\top \hat{\boldsymbol{e}}_j \underbrace{e^{\lambda_i t}}_{\approx 1}\boldsymbol{\phi}_i\right)^\top \hat{\boldsymbol{e}}_j \approx \left(\Phi_r \boldsymbol{y}_j\right)^\top \hat{\boldsymbol{e}}_j = c_{r,j} \tag{5.66}$$

for all time levels $t = t_k$, where $c_{r,j}$ is a constant and $\boldsymbol{y}_j$ is the j-th column vector of $\Phi_r^\top$. This implies that the reduced feature descriptor is almost constant for small times t.

After this observation, we now turn to the MCR method which is also built on dominant modes (low frequencies). The solution of the (decoupled) reduced model $\dot{\boldsymbol{z}}_r(t) = \Lambda_r \boldsymbol{z}_r(t)$ with eigenvalues $0 \leq |\lambda_i| \ll 1$ $(i = 1,\ldots,r)$ reads

$$\boldsymbol{z}_r(t) = \underbrace{e^{\Lambda_r t}}_{\approx I}\boldsymbol{z}_r(0) \approx \boldsymbol{z}_r(0) \tag{5.67}$$

so that with $\boldsymbol{u}(t) \approx \Phi_r \boldsymbol{z}_r(t)$ follows

$$\boldsymbol{u}(t) \approx \Phi_r \boldsymbol{z}_r(t) \approx \Phi_r \boldsymbol{z}_r(0) \overset{(5.52)}{=} \Phi_r \Phi_r^\top D\boldsymbol{u}(0) \tag{5.68}$$

Consequently, the pointwise MCR feature descriptor results in

$$f_{\boldsymbol{x}_j}(t) \approx \left(\Phi_r \Phi_r^\top D\boldsymbol{u}(0)\right)^\top \hat{\boldsymbol{e}}_j = \left(\Phi_r \Phi_r^\top \hat{\boldsymbol{e}}_j\right)^\top \hat{\boldsymbol{e}}_j = \left(\Phi_r \boldsymbol{y}_j\right)^\top \hat{\boldsymbol{e}}_j = c_{r,j} \tag{5.69}$$

Obviously, both reduced approximate solutions (5.66) and (5.69) are equal and have nearly constant behaviour. This shows that the MCR descriptor exhibits in general a slow rate of heat transfer as expected. Consequently, the MCR shape signature is not highly precise in terms of the geometric location on a shape, especially when a small number of modes are used in combination with small stopping times t_F, as e.g. shown for $f_{\boldsymbol{x}_1}(t)$ in Figure 5.17 using the *wolf* dataset.

This observation requires an appropriate adaptation of the MCR method concerning the computation of the feature descriptor in (5.67)-(5.69). A useful way to enhance the geometric information for the shape matching process without increasing the computational effort is to

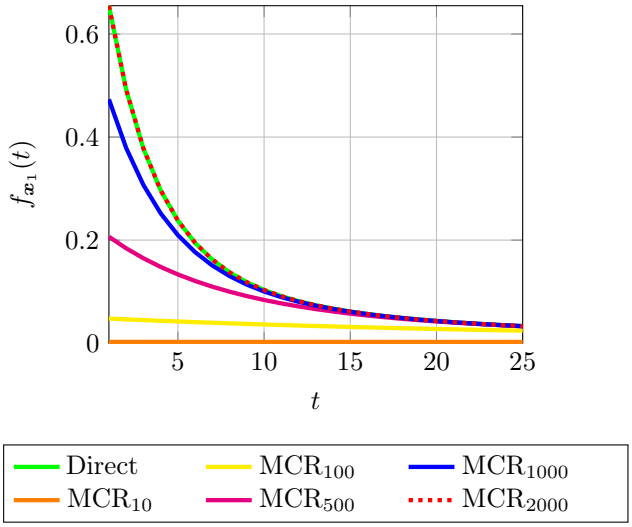


Figure 5.17: Results for the *wolf* dataset using the geometric heat equation for $t_F = 25$. We compare the discrete feature descriptor $f_{\boldsymbol{x}_1}(t)$ obtained by the direct solver and the MCR technique for different number of modes r, shortened here as MCR$_r$. Obviously, a small number of modes used (here $r = 10$ or $r = 100$) correlate to a slow rate of heat transfer. Therefore, the feature descriptors tend to almost constant functions when decreasing the number of modes, and this may lead to shape signature values that give not much information about the geometry of the given surface.

modify the numerical signatures via an *adapted temporal domain*. We intend to construct an *adapted time* $t^\star$ with $t^\star \gg t_F$ for small eigenvalues $0 < |\lambda_i| \ll 1$, that causes $e^{\lambda_i t^\star} \ll e^{\lambda_i t_F} \approx 1$. Obviously, this also generally implies $e^{\lambda_i t} \ll 1$ for all time levels $t = t_k$. Due to $\lambda_i \leq 0$, the latter inequality bears the meaning that we obtain by rescaling time in the indicated way a more significant spreading of values, so that the eigenvalues are more discriminative. This improves the physical characteristics and the diversity of the MCR signatures.

We start with the following thought. Obviously, the rate of heat transfer of the reduced feature descriptor of order r depends on the fastest mode λ_r. In addition, a higher order reduced model $r' > r$ corresponds to faster heat transfer, which in turn implies $t^\star_{r'} < t^\star_r$ to be suitable in order to get a more discriminative descriptor value. Based on this consideration, we modify $[0, t_F]$ to $[0, t^\star_r]$ as follows: we propose to adapt the temporal domain in a simple way using the function

$$t^\star_r = t^\star(\lambda_r) = \frac{k}{\sqrt{|\lambda_r|}} \tag{5.70}$$

where k is a constant. This still to be determined parameter is defined in our construction via the condition

$$t_F \stackrel{!}{=} t^\star(\lambda_N) = \frac{k}{\sqrt{|\lambda_N|}} \tag{5.71}$$

which must be fulfilled for the reduced model of full order N. Therefore, making use of (5.71)

the modified integration domain $[0, t_r^\star]$ is specified by

$$t^\star(\lambda_r) = \frac{t_F \sqrt{|\lambda_N|}}{\sqrt{|\lambda_r|}} \tag{5.72}$$

The corresponding uniform time increment that is required for the time integration can easily be calculated via $\tau = \frac{t_r^\star}{F}$.

Remark 5.15. *One may notice, that the square root in* (5.70) *prevents an extremely large time domain (eigenvalues* $|\lambda_r|$ *grow exponentially) which would lead anew to less descriptive shape signatures.*

Remark 5.16. *The HKS method, described in more detail later in Section 5.7.1, uses logarithmically-spaced time samples for similar reasons, see [267].*

Remark 5.17. *For determining the adapted integration domain one has to compute the largest eigenvalue* $|\lambda_N|$*, but these computational costs are practically negligible.*

As an example to illustrate the usefulness of our rescaling, the strategy given for the *wolf* dataset yields the modified integration domain $[0, t_r^\star]$ shown in Figure 5.18. Based on this modification the adapted feature descriptor $f_{\boldsymbol{x}_1}(t)$, illustrated in Figure 5.19, demonstrates a more suitable numerical signature.

Finally, we compare the matching performances of the MCR technique including the similarity transformation, described in Section 5.6.1, using the *original* $[0, t_F = 25]$ and the *adapted* $[0, t_r^\star]$ temporal domain for the datasets *wolf* and *baby*. The corresponding evaluation is presented in Figure 5.20. In general, the substantial increase of the stopping time $t_r^\star$ of the reduced model significantly improves the geodesic error accuracy. However, the results may differ for a small spectrum as in the case of the *baby* dataset. A possible heuristic explanation could be based on the nature of the first few modes that correspond to a low pass filtered shape signature, meaning that one may obtain a good matching of a coarse shape representation in some cases.

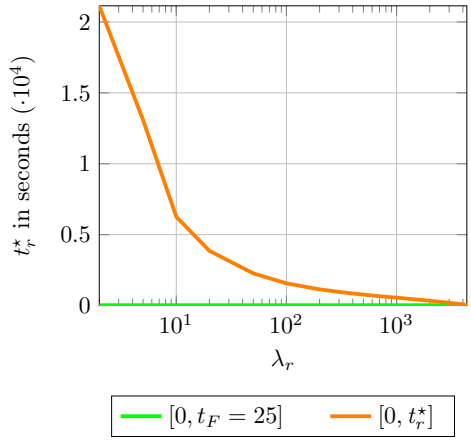


Figure 5.18: Results for the *wolf* dataset using the geometric heat equation. Visualisation of the *original* $[0, t_F = 25]$ and the *adapted* $[0, t_r^\star]$ temporal domain using the formula (5.72) depending on mode λ_r for $r \in [2, 4344]$.

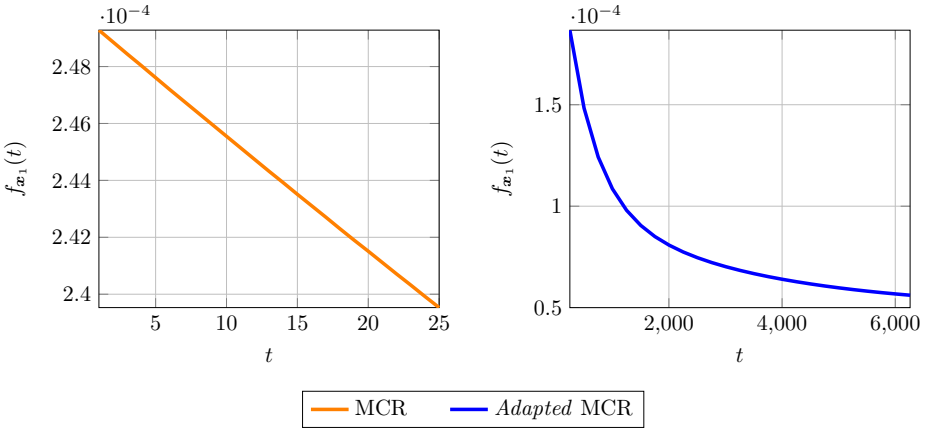


Figure 5.19: Results for the *wolf* dataset using the geometric heat equation (for $F = 25$ time levels). We compare the discrete feature descriptor $f_{\boldsymbol{x}_1}(t)$ between the *original* $[0, t_F = 25]$ (**left**) and the *adapted* $[0, t_r^\star]$ (**right**) temporal domain using the formula (5.72) computed by the MCR technique for $r = 10$. Note that the left figure shows a nearly constant function, whereas a decaying exponential function as on the right after rescaling is the desired result. The modified integration domain clearly provides a more suitable numerical signature.

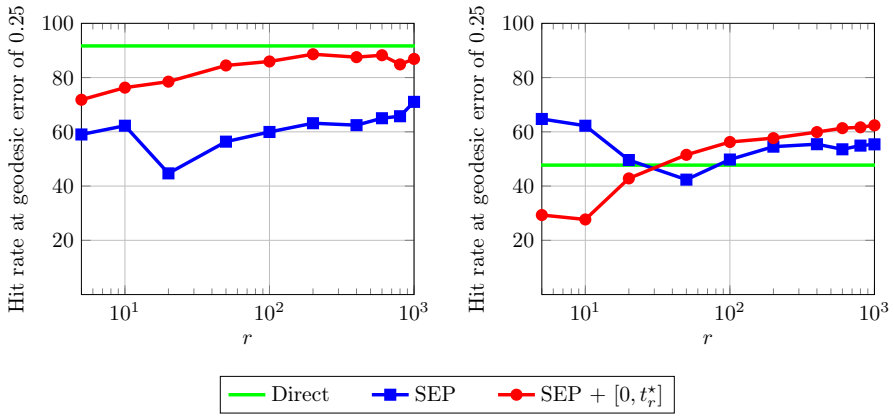


Figure 5.20: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the geometric heat equation. We compare the geodesic error at 0.25 between the *original* $[0, t_F = 25]$ and the *adapted* $[0, t_r^\star]$ temporal domain using formula (5.72) computed by the MCR technique for different number of modes $r \in [5, 1000]$. The eigenvalues and eigenvectors are constructed using the *similarity* transformation (5.60).

MCR Wave Signature The integration domain of the geometric wave equation $\ddot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t)$ can be adapted in an analogous manner. The (decoupled) reduced model $\ddot{\boldsymbol{z}}_r(t) = \Lambda_r \boldsymbol{z}_r(t)$, with the boundary conditions $\boldsymbol{z}_r(0) = \Phi_r^\top D\boldsymbol{u}(0)$, $\dot{\boldsymbol{z}}_r(0) = \Phi_r^\top D\dot{\boldsymbol{u}}(0) = \boldsymbol{0}$, can initially be interpreted as r independent scalar ODEs

$$z_i''(t) = \lambda_i z_i(t), \quad i = 1, \ldots, r \tag{5.73}$$

The analytical solution of (5.73) is given by

$$z_1(t) = A_1 + B_1 t, \quad z_i(t) = A_i \sin(\sqrt{-\lambda_i}t) + B_i \cos(\sqrt{-\lambda_i}t), \quad (i = 2, \dots, r) \tag{5.74}$$

where the terms $A_i \sin(\sqrt{-\lambda_i}t)$ and $B_1 t$ vanish due to the given initial zero velocity conditions, so that we obtain

$$z_1(t) = z_1(0), \quad z_i(t) = \cos(\sqrt{-\lambda_i}t) z_i(0), \quad (i = 2, \dots, r) \tag{5.75}$$

Using only a small number of r dominant eigenvalues with $0 \leq |\lambda_i| \ll 1$ $(i = 1, \dots, r)$ the reduced solution (5.75) can be simplified in matrix form for small times t as

$$\boldsymbol{u}(t) \approx \Phi_r \boldsymbol{z}_r(t) \approx \Phi_r \underbrace{\cos(\sqrt{-\Lambda_r}t)}_{\approx 1} \boldsymbol{z}_r(0) = \Phi_r \Phi_r^\top D\boldsymbol{u}(0) \tag{5.76}$$

Consequently, the corresponding pointwise feature descriptors are given by (5.69) and have again nearly constant behaviour. Based on our experiments we propose to define the modified integration domain $[0, t_r^\star]$ as follows:

$$t^\star(\lambda_r) = \frac{t_F \sqrt[4]{|\lambda_N|}}{\sqrt[4]{|\lambda_r|}} \tag{5.77}$$

Remark 5.18. *We use the 4-th root here because the adapted temporal domain* $[0, t_r^\star]$ *according to the square root* (5.72) *is too large and leads to strong oscillating numerical shape signatures which are less advantageous.*

5.6.3 Modified Initial Condition

The third strategy for improving the matching performance is to modify the initial spatial density introduced in Section 5.3.2. In doing so, we are retaining the Dirac delta function, but the construction of the discrete version is modified. Instead of using the cell average (5.22) linked with unit area, the new condition builds on unit energy which at location $\boldsymbol{x}_i$ can easily be expressed as

$$\boldsymbol{u}_{\boldsymbol{x}_i} = \boldsymbol{u}(\boldsymbol{x}_i, 0) = \hat{\boldsymbol{e}}_i \tag{5.78}$$

The usability of the latter version can be justified as follows. From a theoretical point of view, this modification is incorrect, since the initial condition depends on the volume cell area size. However, this approach works well as the initial condition is independent of the cell size, and remains invariant under remeshing which is an important property in this framework. As a result of this proposal, the higher matching performance using the modified initial condition (5.78) is reflected in Figure 5.21.

Remark 5.19. *The modified initial condition* (5.78) *generally also leads to a higher matching performance for the full model, see Figure 5.21. In contrast, the HKS and WKS methods are based on the original initial condition* (5.24)*. Consequently, this modification highlights another difference to the reference methods.*

In what follows we denote the proposed solver, including all three abovementioned improvements, as *optimised* MCR method. For a fair comparison we will also use the modified

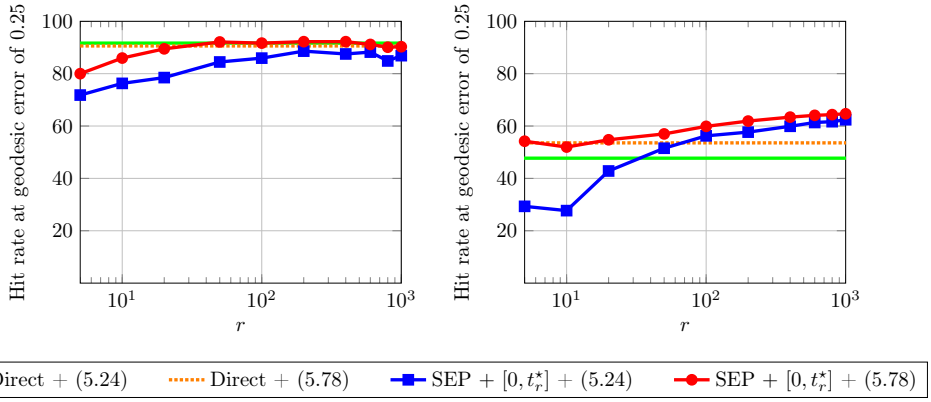


Figure 5.21: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the geometric heat equation. We compare the geodesic error at 0.25 between the *original* (5.24) and the *modified* (5.78) initial condition computed by the MCR technique for different number of modes $r \in [5, 1000]$. The eigenvalues and eigenvectors are constructed by solving the SEP and the adapted temporal domain $[0, t_r^\star]$ is applied. For the sake of completeness, the direct solver including the modified initial condition is also reported. For the *baby* dataset a slightly higher geodesic error accuracy is also achieved using the full model.

initial conditions within the original full model for all further experiments.

So far, the improvements presented have only been demonstrated in relation to the MCR heat signature. For completeness the results of the standard and optimised MCR wave signatures for the given datasets are shown in Figure 5.22.

Remark 5.20. *An interesting aspect is that the optimised MCR technique may achieve a higher matching performance (hit rate at geodesic error of 0.25) compared to the full model, which is exemplarily illustrated in Figures 5.21 and 5.22 when approximately $r > 50$ modes are used. One reason may be that the high-frequency eigenfunctions are more error-prone, the greater the intrinsic deformation. Therefore, low-frequency modes can be more beneficial in this situation.*

5.6.4 MCR: Analytical versus Numerical Solution

An important issue that has not yet been considered up to now is the way in which the solution of the decoupled reduced model is computed. In general, analytical solutions are preferable if they exist, especially in cases where the computational costs are comparatively low such as those of the numerical solver. However, we show that an approximate solution based on the IE scheme will be beneficial in the PSC application. Let us firstly discuss the MCR heat signature, then the wave signature is analysed in more detail.

MCR Heat Signature At the beginning of the investigation, we present the matching performances of the optimised MCR heat signature based on the analytical (5.56) and the numerical (5.54) solution for the dataset *wolf* and *baby*, see Figure 5.23. For the *wolf* dataset both kinds of solutions produce the same geodesic error accuracy, but this is not the case for the *baby* dataset. The latter fact can be explained as follows: the feature descriptor

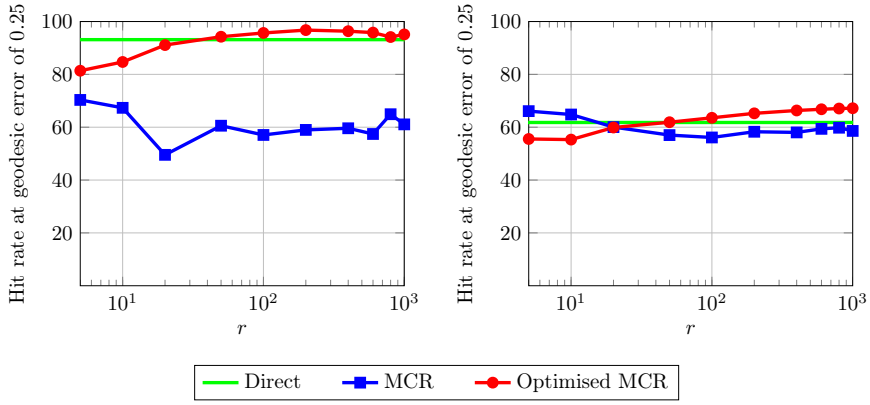


Figure 5.22: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the geometric wave equation. We compare the geodesic error at 0.25 between the *original* and the *optimised* MCR technique for different number of modes $r \in [5, 1000]$. The direct solver also includes the modified initial condition (5.78). As before, the MCR technique is solved by the IE scheme using (5.30)-(5.31).

$f_{\boldsymbol{x}_i}(t)$ constructed on the geometric heat equation decreases exponentially as t increases. Therefore, the variation of the descriptor is large for small times t. In consequence, rapidly decreasing heat signatures are less suitable for the shape matching purpose. However, the IE scheme (5.54) provides a damped version of (5.56) so that its approximation gives a more faithful (in some cases such as here) heat signature, especially at small times t, cf. Figure 5.24. The latter observation of rapidly decreasing heat signatures holds for the *baby* dataset, and consequently the matching performance that results from the IE scheme outperforms these of the analytical solution, as already shown in Figure 5.23.

Remark 5.21. *The analytical* (5.56) *and numerical* (5.54) *solution can be rewritten as*

$$\boldsymbol{z}_r(t+\tau) = e^{\Lambda_r \tau} \boldsymbol{z}_r(t) \tag{5.79}$$

as well as

$$\boldsymbol{z}_r(t_{k+1}) = (I_r - \tau\Lambda_r)^{-1} \boldsymbol{z}_r(t_k) \tag{5.80}$$

respectively, where $t+\tau$ corresponds to t_{k+1}. A comparison of the coefficients of (5.79) *and* (5.80)*, i.e. $e^{\Lambda_r\tau}$ and $(I_r - \tau\Lambda_r)^{-1}$, verifies the damping effect of the IE scheme when we explicitly consider the following inequality with $x = -\tau\lambda_r$ as $\lambda_r < 0$:*

$$e^{-x} \leq \frac{1}{1+x}, \quad \forall x \geq 0 \tag{5.81}$$

For this reason, the IE scheme provides a more suitable feature descriptor, especially when the corresponding analytical heat signature changes more rapidly at small times t.

Remark 5.22. *In relation* (5.79) *the term*

$$\boldsymbol{z}_r(t+\tau) = e^{\Lambda_r(t+\tau)} \boldsymbol{z}_r(0) = e^{\Lambda_r t} e^{\Lambda_r \tau} \boldsymbol{z}_r(0) = e^{\Lambda_r \tau} \boldsymbol{z}_r(t) \tag{5.82}$$

with $\boldsymbol{z}_r(t) = e^{\Lambda_r t} \boldsymbol{z}_r(0)$ is used.

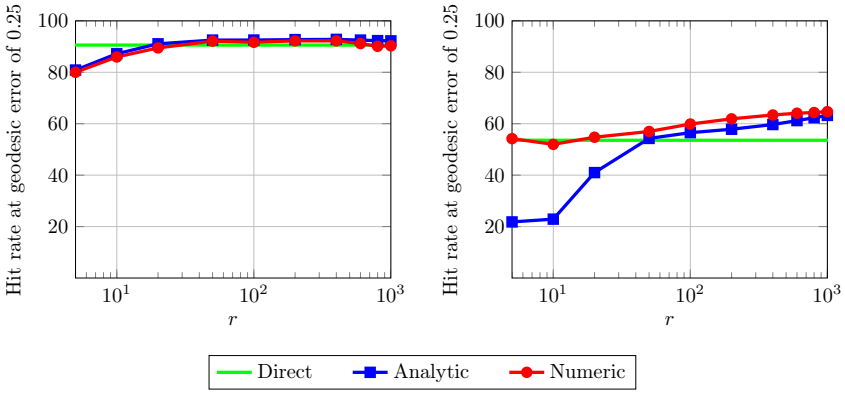


Figure 5.23: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the geometric heat equation. We compare the geodesic error at 0.25 between the *analytical* (5.56) and the *numerical* (5.54) solution computed by the optimised MCR technique for different number of modes $r \in [5, 1000]$. The numerical solution is based on the IE scheme and achieves a significantly higher matching performance for the *baby* dataset.

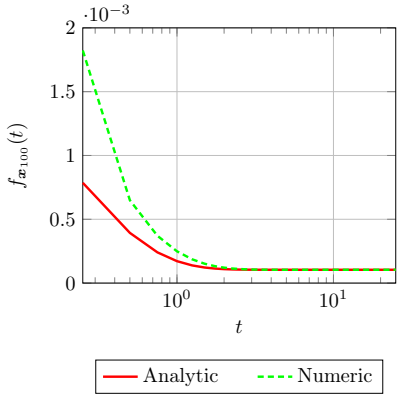


Figure 5.24: Results for the *baby* dataset using the geometric heat equation. We compare the discrete feature descriptor $f_{\boldsymbol{x}_{100}}(t)$ obtained by the *analytical* (5.56) and the *numerical* (5.54) solution using the MCR technique for $r = 100$. The numerical solution is based on the IE scheme. For a better comparison the time axis is scaled logarithmically. Obviously, the heat signature linked to the numerical solution is damped.

MCR Wave Signature The approach for computing the solution with regard to the geometric wave equation is discussed analogously to the above procedure. As previously stated, the analytical solution of the decoupled reduced model is given by

$$\boldsymbol{z}_r(t) = \cos(\sqrt{-\Lambda_r}t)\boldsymbol{z}_r(0) \tag{5.83}$$

In contrast to this the numerical solution, cf. (5.30)-(5.31), reads as

$$\boldsymbol{z}_r(t_{k+1}) = (I_r - \tau^2\Lambda_r)^{-1}\left(2\boldsymbol{z}_r(t_k) - \boldsymbol{z}_r(t_{k-1})\right) \tag{5.84}$$

On closer examination, the exact solution (5.83) reflects a pure oscillating behaviour, whereas the IE scheme once again adds an artificial damping to the numerical solution (5.84). The corresponding comparison of both kinds of solution in terms of matching performance is illustrated in Figure 5.25. Contrary to expectations, the (damped) numerical solution clearly achieves a higher geodesic error accuracy for both datasets. Usually, we would have expected that the (undamped) exact solution is the better procedure, as a more unique wave signature is obtained. However, this presumption can be invalidated for the discrete shape matching application as follows: in general, it can be assumed that the mesh arises from the point cloud is slightly shifted and altered due to the almost isometric transformation. A function that lives on the shifted/changed mesh is sampled differently. Comparing the same function on the slightly different grids, may lead to a deviation in the amplitude and also produce a phase shift. For this reason, the point-to-point correspondence of two points, that actually belong together, can fail as a consequence of the phase shift when using the analytical solution (5.83). Contrary to this, a damped wave signature can be beneficial so that a higher matching probability is generally achieved.

The latter justification is described in more detail using a test example at hand of the points $\boldsymbol{x}_{1611} \in \mathcal{M}$ and $\widetilde{\boldsymbol{x}}_{1611}, \widetilde{\boldsymbol{x}}_{795} \in \widetilde{\mathcal{M}}$ for the *wolf* dataset. It should be noted that the points $\boldsymbol{x}_{1611}$ and $\widetilde{\boldsymbol{x}}_{1611}$ belong together. The feature descriptors $f_{\boldsymbol{x}_{1611}}(t)$, $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$, $f_{\widetilde{\boldsymbol{x}}_{795}}(t)$ obtained by (5.83) and (5.84) for $r = 100$ are presented in Figure 5.26. On the one hand, the wave signatures $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$ which are computed by (5.83) produce a phase shift for $t > 10$. On the other hand, $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{795}}(t)$ generally match up to an amplitude for all times t. Interestingly, the wave signatures $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$ obtained via (5.84) are almost fit together on the considered temporal domain.

To illustrate the latter results more comparable, we use the L_1-error (along the temporal axis) defined as

$$\widetilde{d}_{f_{(i,j)}}(t) = \left| f_{\boldsymbol{x}_i}(t) - f_{\widetilde{\boldsymbol{x}}_j}(t) \right| \quad \boldsymbol{x}_i \in \mathcal{M},\, \widetilde{\boldsymbol{x}}_j \in \widetilde{\mathcal{M}} \tag{5.85}$$

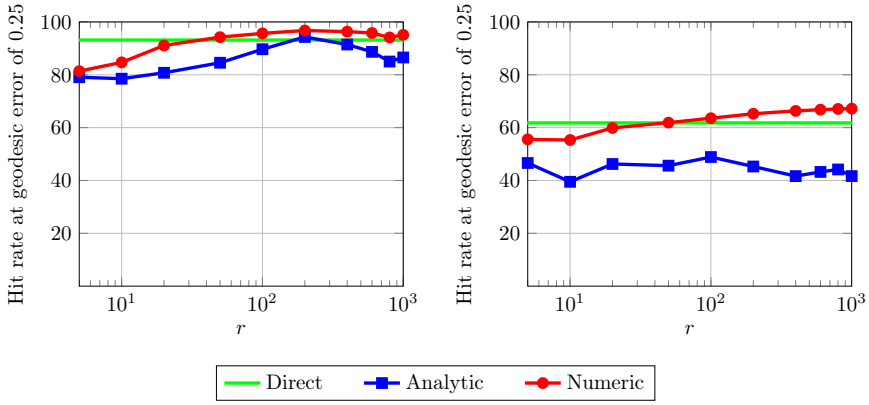


Figure 5.25: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the geometric wave equation. We compare the geodesic error at 0.25 between the *analytical* (5.83) and the *numerical* (5.84) solution computed by the optimised MCR technique for different number of modes $r \in [5, 1000]$. The numerical solution is based on the IE scheme and achieves a significantly higher matching performance, especially for the *baby* dataset.

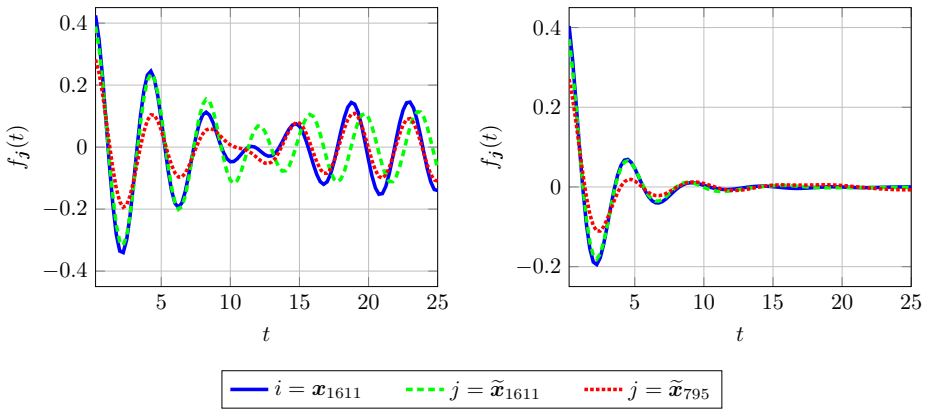


Figure 5.26: Results for the *wolf* dataset using the geometric wave equation. We compare the discrete feature descriptor between $f_{\boldsymbol{x}_{1611}}(t) \in \mathcal{M}$ and $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$, $f_{\widetilde{\boldsymbol{x}}_{795}}(t) \in \widetilde{\mathcal{M}}$ obtained by the *analytical* (5.83) (**left**) and the *numerical* (5.84) (**right**) solution using the MCR technique for $r = 100$. Note that the points $\boldsymbol{x}_{1611}$ and $\widetilde{\boldsymbol{x}}_{1611}$ belong together. Obviously, the wave signatures linked to (5.83) are damped-free. Furthermore, it can also be seen that $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$ generally match for $t < 10$, otherwise there is a *phase shift* between both signatures. In contrast, $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{795}}(t)$ have the same function profile with a generally decreasing amplitude. Apart from that, the signatures $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$ constructed by (5.84) are relatively equal on the considered temporal domain due to the dampened effect of the IE scheme. The corresponding L_1-errors (5.85) are shown in Figure 5.27.

The corresponding L_1-error between $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$, $f_{\widetilde{\boldsymbol{x}}_{795}}(t)$ is shown in Figure 5.27. Obviously, the phase shift leads to a high accumulated L_1-error and thus to a wrong point-to-point correspondence. In contrast, the correct matching is preserved when the damped IE scheme is used, which prevents an unfavourable phase shift for this test example.

The last experiment demonstrates the beneficial use of the IE scheme for the MCR wave signature. Overall, we have shown through various tests that the IE scheme (for both geometric PDEs) is superior compared to the analytical solution due to its higher geodesic error accuracy. Therefore, we will apply the optimised MCR method based on the IE schemes (5.54) and (5.84) for all further experiments.

Remark 5.23. *A damped wave signature can be achieved in different ways. Another possibility is to apply an appropriate weighting within* (5.8)*, however, we prefer some kind of weighting depending on the considered point of a given shape. And that is, in fact, exactly what the IE scheme provides. Another possibility would be to remodel the continuous geometric wave equation by adding a desired damping term, for example via a diffusion process. The latter option models in particular an analytical damping instead of a numerical (artificial) damping and is also an interesting issue, but this is beyond the scope of this work.*

5.6.5 Implementation of the Optimised MCR Method

We now briefly describe the simple implementation of the optimised MCR signature which makes it a candidate for practical shape analysis purposes.

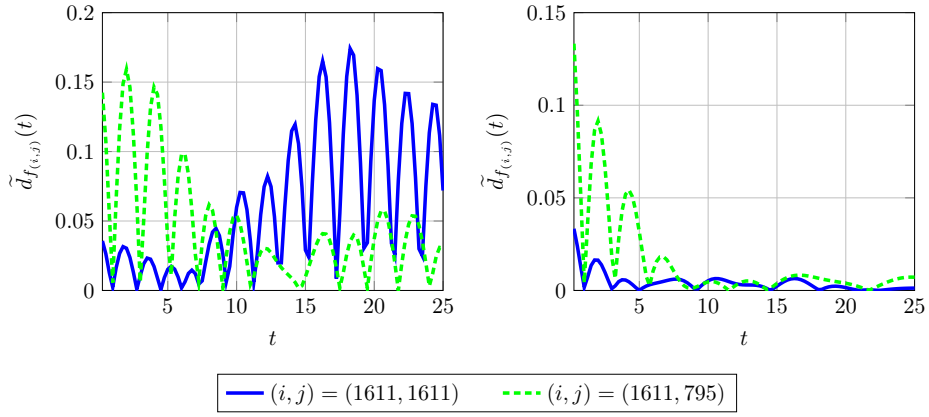


Figure 5.27: Results for the *wolf* dataset using the geometric wave equation. We compare the L_1-error (5.85) between the discrete feature descriptors $f_{\boldsymbol{x}_{1611}}(t)$ and $f_{\widetilde{\boldsymbol{x}}_{1611}}(t)$, $f_{\widetilde{\boldsymbol{x}}_{795}}(t)$ obtained by the *analytical* (5.83) (**left**) and the *numerical* (5.84) (**right**) solution using the MCR technique for $r = 100$. Although the points $\boldsymbol{x}_{1611}$ and $\widetilde{\boldsymbol{x}}_{1611}$ belong together, its accumulated L_1-error of $\widetilde{d}_{f_{(1611,1611)}}(t)$ constructed via (5.83) is much larger than using (5.84). Thus, the point-to-point correspondence is preserved (for this test example) when the damped IE scheme is used. The results of this experimental investigation should demonstrate why the use of the analytical solution is not beneficial and consequently leads to worse matching performances compared to the use of (5.84), cf. Figure 5.25.

As already mentioned, the feature descriptors have to be computed for each point and on each shape. Therefore, two implementation strategies exist. First, the signatures can be calculated separately for each point combined with parallel computing[7] to speed up the CPU time. Second, the computation can be done simultaneously for all locations $\boldsymbol{x}_i \in \mathcal{M}$ on a given shape by putting it into the format of a matrix-matrix multiplication. In addition, a parallelisation on GPUs[8] can be used, which further improves the computational efficiency. The CPU times for both strategies based on the geometric heat equation are illustrated in Figure 5.28. Since the GPU computation is more beneficial in our setting, we explain the implementation procedure based on the second strategy.

Optimised MCR Heat Signature Solving the decoupled and reduced geometric heat equation numerically using the IE scheme (5.54) including the modified initial conditions (5.78) can be rewritten into the matrix-matrix multiplication format as

$$\begin{aligned} Z_r^{k+1} &= P Z_r^k \quad \text{and} \quad U^{k+1} \approx U_r^{k+1} = \Phi_r Z_r^{k+1}, \quad k = 0, \dots, F-1 \\ Z_r^0 &= \Phi_r^\top D U^0 = \Phi_r^\top D \end{aligned} \tag{5.86}$$

with $\Phi_r \in \mathbb{R}^{N \times r}$, $Z_r \in \mathbb{R}^{r \times N}$, $U^0 = I$ and $P = (I_r - \tau \Lambda_r)^{-1}$. In particular, Z_r and U_r correspond to the reduced solutions in modal and physical coordinates, respectively. The complete algorithm for computing the optimised MCR heat signature is described in Figure

[7] The class *parfor* from MATLAB 2018b is employed. The computations are conducted with six cores here.
[8] The class *gpuArray* from MATLAB 2018b is employed.

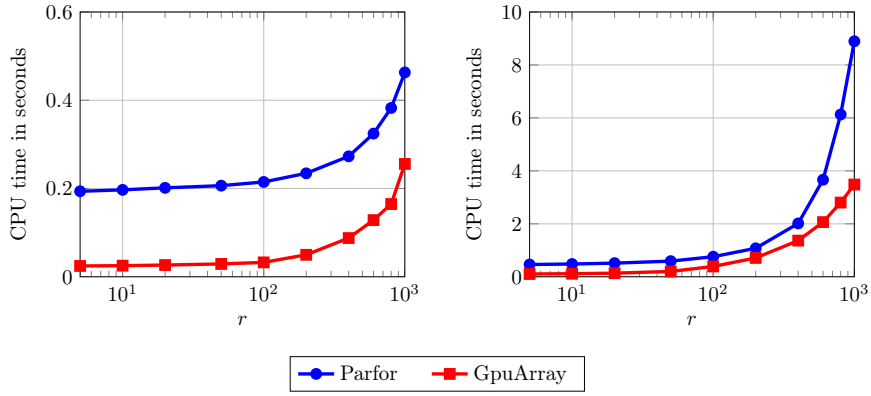


Figure 5.28: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the geometric heat equation. We compare the CPU time between the class *parfor* and the class *gpuArray* computed by the optimised MCR technique for different number of modes $r \in [5, 1000]$. The CPU times indicate the pure computational costs for solving the reduced systems without the costs for generating eigenvalues/eigenvectors. The computations are conducted on six cores or with NVIDIA GeForce GTX 690. The use of *gpuArray* is in our setting significantly faster than using *parfor*. However, we stress again that almost all computational costs incurred when solving the eigenvalue problem.

5.29. It should be stressed that the extraction of the feature descriptors $f_{\boldsymbol{x}_i}(t)$ in step 3c) of the algorithm uses the Hadamard product. Finally, the correspondence quality can be evaluated on the selected measure.

Remark 5.24. *Incidentally, the computation of the temporal domain* $[0, t_r^\star]$ *and the time increment* τ *in step 2.) of the Algorithm 5.1 is only done for the reference shape. This ensures that a correct matching is made by comparing the feature descriptors for different locations* $\boldsymbol{x}_i \in \mathcal{M}$ *and* $\widetilde{\boldsymbol{x}}_j \in \widetilde{\mathcal{M}}$ *on the time scale. This is another difference to the HKS method, in which signatures for different shapes are computed at different time samples.*

Remark 5.25. *Another interesting issue would be the use of a nonuniform time sampling. This may lead to a potential improvement if more information is captured, especially at small times* t*.*

Optimised MCR Wave Signature The above aspects can analogously be used to numerically solve the geometric wave equation by the IE scheme (5.30)-(5.31). The implementation procedure for computing the optimised MCR wave signature, see Figure 5.30, remains identical to the Algorithm 5.1, except for the calculation of $t_r^\star$ in step 2b) and the numerical solution scheme in step 3b)-3d).

Finally, a comparison of the optimised MCR method using the geometric heat and wave equation is illustrated in Figure 5.31. Overall, the geometric wave equation achieves approximately 3% higher matching performance on average than using the heat equation for the two considered datasets. The more accurate matching of the wave equation is primarily due to the fact that wave signatures capture more information of shape variations and thus give better separation of the signatures.

Algorithm 5.1 Optimised MCR heat signature

Input: Matrices D^{-1}, C; reduction order r; amount of time levels $F+1$; stopping time t_F

Output: $f_{\boldsymbol{x}_i}(t)$

Reference Shape

1.) Computation of $\Lambda_r \in \mathbb{R}^{r\times r}$ and $\Phi_r \in \mathbb{R}^{N\times r}$

a) Compute r dominant eigenpairs $(\lambda, \boldsymbol{\phi})$ of $B = D^{-\frac{1}{2}} C D^{-\frac{1}{2}}$ by solving $B\boldsymbol{\phi} = \lambda\boldsymbol{\phi}$

b) Calculation of $\left(\lambda, \widetilde{\boldsymbol{\phi}}\right)$ of L with $\widetilde{\boldsymbol{\phi}} = D^{-\frac{1}{2}}\boldsymbol{\phi}$

2.) Computation of $[0, t_r^\star]$ using (5.72)

a) Compute fastest mode λ_N by solving $B\boldsymbol{\phi} = \lambda\boldsymbol{\phi}$

b) $t_r^\star = \dfrac{t_F\sqrt{|\lambda_N|}}{\sqrt{|\lambda_r|}}, \quad \tau = \dfrac{t_r^\star}{F}$

3.) Solving reduced system (5.86)

a) $Z_r^0 = \Phi_r^\top D$

b) $P = (I_r - \tau\Lambda_r)^{-1}$

c) Compute $Z_r^{k+1} = PZ_r^k$ and Extract $f_{\boldsymbol{x}_i}(t)$ by

$$f_{\boldsymbol{x}_i}(t_{k+1}) = \sum_{j=1}^{r}\left[\left(\Phi_r \circ \left(Z_r^{k+1}\right)^\top\right)\right]_{ij} \text{ for } k = 0, \ldots, F-1$$

Transformed Shape(s)

Repeat computations of Steps 1.) and 3.)

Figure 5.29: Algorithm to compute the optimised MCR signature by solving the geometric heat equation. The implementation is structured into three parts. First, the eigenvalues and eigenvectors are computed by solving SEP. In a second step, the adapted integration domain $[0, t_r^\star]$ is calculated via (5.72). The last step includes solving the decoupled and reduced ODE system (5.86) based on the IE scheme and extracting the pointwise feature descriptors.

Algorithm 5.2 Optimised MCR wave signature

Input: Matrices D^{-1}, C; reduction order r; amount of time levels $F+1$; stopping time t_F

Output: $f_{\boldsymbol{x}_i}(t)$

Reference Shape

1.) Computation of $\Lambda_r \in \mathbb{R}^{r\times r}$ and $\Phi_r \in \mathbb{R}^{N\times r}$

a) Compute r dominant eigenpairs $(\lambda, \boldsymbol{\phi})$ of $B = D^{-\frac{1}{2}} C D^{-\frac{1}{2}}$ by solving $B\boldsymbol{\phi} = \lambda\boldsymbol{\phi}$

b) Calculation of $\left(\lambda, \widetilde{\boldsymbol{\phi}}\right)$ of L with $\widetilde{\boldsymbol{\phi}} = D^{-\frac{1}{2}}\boldsymbol{\phi}$

2.) Computation of $[0, t_r^\star]$ using (5.77)

a) Compute fastest mode λ_N by solving $B\boldsymbol{\phi} = \lambda\boldsymbol{\phi}$

b) $t_r^\star = \dfrac{t_F \sqrt[4]{|\lambda_N|}}{\sqrt[4]{|\lambda_r|}} \quad \tau = \dfrac{t_r^\star}{F}$

3.) Solving reduced system based on (5.30)-(5.31)

a) $Z_r^0 = \Phi_r^\top D$

b) $P = \left(I_r - \tau^2 \Lambda_r\right)^{-1}$

c) Compute $Z_r^1 = P Z_r^0$ and Extract $f_{\boldsymbol{x}_i}(t_1) = \sum_{j=1}^{r} \left[\left(\Phi_r \circ \left(Z_r^1\right)^\top\right)\right]_{ij}$

d) Compute $Z_r^{k+1} = P(2Z_r^k - Z_r^{k-1})$ and Extract $f_{\boldsymbol{x}_i}(t)$ by

$$f_{\boldsymbol{x}_i}(t_{k+1}) = \sum_{j=1}^{r} \left[\left(\Phi_r \circ \left(Z_r^{k+1}\right)^\top\right)\right]_{ij} \quad \text{for } k = 1, \ldots, F-1$$

Transformed Shape(s)

Repeat computations of Steps 1.) and 3.)

Figure 5.30: Algorithm to compute the optimised MCR signature by solving the geometric wave equation. The implementation procedure is identical to the Algorithm 5.1 in Figure 5.29, except for the calculation of $t_r^\star$ in step 2b) and the numerical solution scheme in step 3b) - 3d).

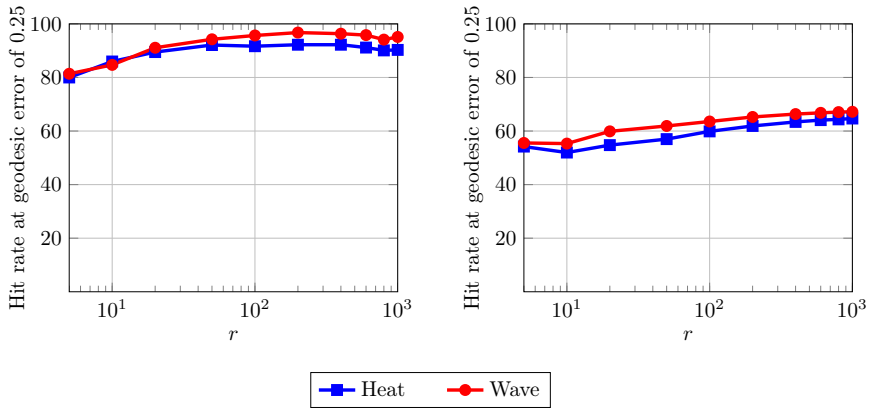


Figure 5.31: Results for the dataset *wolf* (**left**) and *baby* (**right**) by applying the optimised MCR technique. We compare the geodesic error at 0.25 between using the geometric heat and wave equation for different number of modes $r \in [5, 1000]$. The wave equation achieves a slightly higher geodesic error accuracy.

5.7 Reference Models

In this section we describe in more detail those approaches that are explicitly based on the analytical solution of the underlying geometric PDEs. These models are called *kernel-based methods* (or *spectral methods*) and relates to the matrix decomposition methods (eigendecomposition methods) to approximate the matrix exponential of the discrete Laplacian. Apart from that, we will shortly discuss the rational approximants and the Krylov subspace methods for approximating the matrix exponential.

Let us note here that also the procedure described in [179] is technically related to the approaches as it is a time-evolution method and is based on diffusion. However, as noted in [179], the technique is computationally very demanding compared to HKS and WKS construction. We therefore refrain from comparing to this method, as one of our main aims is computational efficiency.

5.7.1 Kernel-Based Methods

The spectral methods HKS and WKS that we employ for comparison with our MCR approach are based on the geometric heat equation [267] and the geometric Schrödinger equation [13,14], respectively. As already indicated, the kernel-based methods build on the analytical solutions of the underlying geometric PDEs. In particular, the descriptor class is based on the spectral decomposition of the Laplace-Beltrami operator and yields a series expression of its eigenfunctions and eigenvalues. The contributions in such a series are ordered in a way that the first terms contain the low frequency components describing the macroscopic (global) properties of a shape. Taking into account a corresponding part of the spectral components thus leads to a feature descriptor that is robust to local errors such as (high frequent) noise, but vulnerable to global distortions such as e.g. changes in shape topology.

For the construction of spectral descriptors, it is assumed that a function of the form $u(\boldsymbol{x}, t) = \phi(\boldsymbol{x})\theta(t)$ will be the solution of the geometric heat and Schrödinger equation which is true if the underlying PDEs are linear and homogeneous. The approach of separation is

built on the fact that if the product of two functions ϕ and θ that depend on different sets of variables is a constant, then each function must separately be a constant. Therefore, one may separate the equations to get a function of only t and $\boldsymbol{x}$:

$$\frac{\theta_t(t)}{\kappa\theta(t)} = \frac{\Delta_{\mathcal{M}}\phi(\boldsymbol{x})}{\phi(\boldsymbol{x})} = \text{const} = \lambda \tag{5.87}$$

where κ summarises both equations, namely $\kappa = 1$ for geometric heat equation or $\kappa = i$ for geometric Schrödinger equation, and λ is called the (arbitrary) separation constant. This leaves us with two new equations, namely an ODE for the temporal component

$$\theta_t(t) = \kappa\lambda\theta(t), \quad t \in [0, T] \tag{5.88}$$

and the spatial part takes the form of the Helmholtz equation

$$\begin{cases} \Delta_{\mathcal{M}}\phi(\boldsymbol{x}) = \lambda\phi(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{M} \\ \langle \nabla_{\mathcal{M}}\phi, \boldsymbol{n} \rangle = 0, & \boldsymbol{x} \in \partial\mathcal{M} \end{cases} \tag{5.89}$$

where the constant λ has here the meaning of the operator's eigenvalue.

For a compact domain $\mathcal{M}$, which is the case here, the Laplace-Beltrami operator is a self-adjoint operator on the space $L_2(\mathcal{M})$. This implies that the Helmholtz equation has an infinite number of nontrivial solutions ϕ_k for certain eigenvalues λ_k and corresponding eigenfunctions, which is a result of the spectral theorem [245]. Consequently, (5.89) takes the following format:

$$\begin{cases} \Delta_{\mathcal{M}}\phi_k(\boldsymbol{x}) = \lambda_k\phi_k(\boldsymbol{x}), & \boldsymbol{x} \in \mathcal{M} \\ \langle \nabla_{\mathcal{M}}\phi_k, \boldsymbol{n} \rangle = 0, & \boldsymbol{x} \in \partial\mathcal{M} \end{cases} \quad k = 1, 2, \ldots \tag{5.90}$$

In particular, the ordered spectrum of eigenvalues $\{0 = \lambda_1 > \lambda_2 \geq \ldots\}$ and the corresponding eigenfunctions $\{\phi_1, \phi_2, \ldots\}$ form an orthonormal basis of $L_2(\mathcal{M})$.

Remark 5.26. *With Neumann boundary conditions or without shape boundaries, constant functions are solutions of the Helmholtz equation for the first eigenvalue with zero value. Thus, only one eigenvalue vanishes with the constant function as the corresponding eigenfunction.*

It is well known that the eigenfunctions are a natural generalisation of the classical Fourier basis for working with functions on shapes. In this context, the physical interpretation of the Helmholtz equation can be understood as follows. The shape of a three-dimensional object can be viewed as a vibrating membrane, the ϕ_k can be interpreted as its vibration modes, while λ_k have the meaning of the corresponding vibration frequencies, sorted from low to high magnitude.

Thus, for each index k one obtains an ODE depending on t such as (5.17) which can be solved by indefinite integration:

$$\int \frac{\mathrm{d}\theta(t)}{\theta(t)} = \int \kappa\lambda_k \,\mathrm{d}t \quad \Longrightarrow \quad \theta(t) = \alpha_k e^{\kappa\lambda_k t} \tag{5.91}$$

where the integration constant α_k should satisfy the initial condition of the k-th eigenfunction.

Consequently, the product solution reads as

$$u_k(\boldsymbol{x}, t) = \alpha_k e^{\kappa \lambda_k t} \phi_k(\boldsymbol{x}) \tag{5.92}$$

Finally, for a linear homogeneous differential equation the principle of superposition can be used so that the analytical solution can be expressed as the sum (linear combination) of all individual solutions. Therefore, a closed-form solution of the geometric heat equation is expressed in terms of a series with

$$u(\boldsymbol{x}, t) = \sum_{k=1}^{\infty} \alpha_k e^{\lambda_k t} \phi_k(\boldsymbol{x}) \tag{5.93}$$

and the solution of the geometric Schrödinger equation is given via

$$u(\boldsymbol{x}, t) = \sum_{k=1}^{\infty} \alpha_k e^{i \lambda_k t} \phi_k(\boldsymbol{x}) \tag{5.94}$$

where the coefficients α_k are chosen to fulfil the initial condition.

Heat Kernel Signature The coefficients α_k of the series expansion (5.93) can be computed using the L_2-inner product

$$\alpha_k = \langle u_0, \phi_k \rangle_{L_2(\mathcal{M})} = \int_{\mathcal{M}} u_0(\boldsymbol{y}) \phi_k(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \tag{5.95}$$

which gives

$$u(\boldsymbol{x}, t) = \sum_{k=1}^{\infty} \left(\int_{\mathcal{M}} u_0(\boldsymbol{y}) \phi_k(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \right) e^{\lambda_k t} \phi_k(\boldsymbol{x}) = \int_{\mathcal{M}} u_0(\boldsymbol{y}) \underbrace{\left(\sum_{k=1}^{\infty} e^{\lambda_k t} \phi_k(\boldsymbol{y}) \phi_k(\boldsymbol{x}) \right)}_{=K(\boldsymbol{x}, \boldsymbol{y}, t)} \mathrm{d}\boldsymbol{y} \tag{5.96}$$

The *heat kernel* $K(\boldsymbol{x}, \boldsymbol{y}, t)$ describes the amount of heat that is transferred from $\boldsymbol{x}$ to $\boldsymbol{y}$ in time t. By setting the initial condition to be a delta heat distribution with $u_0(\boldsymbol{y}) = \delta_{\boldsymbol{x}}(\boldsymbol{y})$ and $\int_{\mathcal{M}} \delta_{\boldsymbol{x}}(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} = 1$ at the position $\boldsymbol{y}$, we obtain according to [267] the HKS feature descriptor

$$\mathrm{HKS}(\boldsymbol{x}, t) = \sum_{k=1}^{\infty} e^{\lambda_k t} \left| \phi_k(\boldsymbol{x}) \right|^2 \tag{5.97}$$

where the shifting property of the delta distribution $f(\boldsymbol{x}) = \int_{\mathcal{M}} f(\boldsymbol{y}) \delta_{\boldsymbol{x}}(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y}$ is applied. Accordingly, the term $\mathrm{HKS}(\boldsymbol{x}, t)$ describes the amount of heat at point $\boldsymbol{x}$ at the time t. Furthermore, the discrete HKS constructs a feature descriptor for a point $\boldsymbol{x}_i$ on a given shape $\mathcal{M}$ via

$$\mathrm{HKS}(\boldsymbol{x}_i, t) = \sum_{k=1}^{N} e^{\lambda_k t} \left| \phi_k(\boldsymbol{x}_i) \right|^2 \tag{5.98}$$

and is the corresponding equivalent of the MCR heat signature.

Remark 5.27. *As already notified, the discrete HKS signature* (5.98) *is interpreted formally as the raw version and corresponds to the analytical solution* (5.57) *using the MCR technique.*

In order to enhance the discrete signatures for shape matching purposes the authors in [267] apply two heuristic modifications. First, for comparing two computed signatures on different time intervals a normalised distance based on the L_2-norm is used as follows:

$$d_{\mathrm{HKS}}(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j) = \left(\int_{t_a}^{t_b} \left(\frac{\left| \mathrm{HKS}(\boldsymbol{x}_i, t) - \mathrm{HKS}(\widetilde{\boldsymbol{x}}_j, t) \right|}{\int\limits_{\mathcal{M}} \mathrm{HKS}(\boldsymbol{x}, t)} \right)^2 \mathrm{d}\log t \right)^{\frac{1}{2}} \tag{5.99}$$

Second, uniform time samples $t_a := t_1 < t_2 < \cdots < t_F =: t_b$ over the logarithmic scaled temporal domain $[t_a, t_b]$ with $t_a = \frac{4\ln(10)}{|\lambda_r|}$ and $t_b = \frac{4\ln(10)}{|\lambda_2|}$ are used (validated merely by experiments), to overcome the issue which arises for larger times t in which the heat distribution converges to a less informative constant temperature. Consequently, the nonlogarithmic time sampling is actually a nonuniform discretisation.

Remark 5.28. *In practice the dominance of small eigenvalues at larger times t is compensated by introducing a logarithmic time sampling.*

Wave Kernel Signature The authors in [14] define the WKS as the time-averaged probability of detecting a particle of a certain energy distribution at the point $\boldsymbol{x}$ in the form of

$$\mathrm{WKS}(\boldsymbol{x}, e) = \lim_{T\to\infty} \frac{1}{T} \int_0^T \left| u(\boldsymbol{x}, t) \right|^2 \mathrm{d}t = \sum_{k=1}^{\infty} \left| \alpha(e_k) \right|^2 \left| \phi_k(\boldsymbol{x}) \right|^2 \tag{5.100}$$

whereby taken in advantage the fact that the functions $e^{i\lambda_k t}$ are orthogonal for the L_2-norm. Moreover, the time variable is replaced by an energy which is directly related to the eigenvalues of the Laplace-Beltrami operator.

Remark 5.29. *Replacing the time variable generally highlights the suggestion to use the Schrödinger equation, since the constructed feature descriptor then forms a function of the energy of a particle. In this way, WKS constructs more intuitive and discriminative signatures.*

Thus, $\alpha(e_k)$ in (5.100) becomes a function of the energy distribution e_k of the quantum mechanical particle and can be chosen as a log-normal distribution, i.e.

$$\left| \alpha(e_k) \right|^2 = \exp\left(\frac{-\left(e - \log|\lambda_k|\right)^2}{2\sigma^2} \right) \tag{5.101}$$

where σ denoted the variance of the energy distribution, cf. [13, 14] for more details. Finally, the WKS is obtained in the form

$$\mathrm{WKS}(\boldsymbol{x}, e) = C_e \sum_{k=1}^{\infty} \phi_k^2(\boldsymbol{x}) \exp\left(\frac{-\left(e - \log|\lambda_k|\right)^2}{2\sigma^2} \right) \tag{5.102}$$

and in discrete sense

$$\mathrm{WKS}(\boldsymbol{x}_i, e) = C_e \sum_{k=1}^{N} \phi_k^2(\boldsymbol{x}_i) \exp\left(\frac{-\left(e - \log|\lambda_k|\right)^2}{2\sigma^2}\right) \tag{5.103}$$

with a normalisation factor C_e. To compare two discrete WKS signatures a normalised distance based on the L_1-norm is employed

$$d_{\mathrm{WKS}}(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j) = \int_{e_a}^{e_b} \left| \frac{\mathrm{WKS}(\boldsymbol{x}_i, e) - \mathrm{WKS}(\widetilde{\boldsymbol{x}}_j, e)}{\mathrm{WKS}(\boldsymbol{x}_i, e) + \mathrm{WKS}(\widetilde{\boldsymbol{x}}_j, e)} \right| \mathrm{d}e \tag{5.104}$$

where $e_a = \log(|\lambda_2|) + 2\sigma, e_b = \log(|\lambda_r|) - 2\sigma$ and the uniform time increment is fixed to $\tau = \frac{e_b - e_a}{F}$ as described in [14].

Remark 5.30. *The latter heuristic of scaling the energy domain (logarithmic energy scale) is employed in order to enhance the practicability of the WKS techniques.*

Computational Aspects In contrast to the present cotangent weight scheme [184], the kernel-based methods rely on the mesh Laplace operator [31] as discretisation of the Laplace-Beltrami operator on triangular meshes. In addition, the eigenfunctions and eigenvalues of the discrete Laplacian are computed by solving the GEP. Let us recall that the eigenvectors of $L = D^{-1}C$ with respect to the inner product

$$\langle \boldsymbol{f}, \boldsymbol{g} \rangle_D = \boldsymbol{f}^\top D \boldsymbol{g} \tag{5.105}$$

are D-orthogonal with $\boldsymbol{\phi}_i^\top D \boldsymbol{\phi}_j = \delta_{ij}$. Thus, in the discrete setting, the HKS signature for a given initial vector $\boldsymbol{f} : \mathcal{M} \to \mathbb{R}^N$ can be specified as

$$\mathrm{HKS}(\cdot, t) = \sum_{k=1}^{N} e^{\lambda_k t} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle_D \, \boldsymbol{\phi}_k \tag{5.106}$$

The WKS signature is handled analogously.

Remark 5.31. *The latter representation can be expressed as the matrix exponential of the discrete Laplacian. First, the presented heat kernel* (5.106) *can be rewritten in matrix form as*

$$\mathrm{HKS}(\cdot, t) = \sum_{k=1}^{N} e^{\lambda_k t} \langle \boldsymbol{f}, \boldsymbol{\phi}_k \rangle_D \, \boldsymbol{\phi}_k = \Phi e^{\Lambda t} \Phi^\top D \boldsymbol{f} = K_t \boldsymbol{f} \tag{5.107}$$

with $K_t = \Phi e^{\Lambda t} \Phi^\top D$. *Using* (2.148) *and the equalities* (5.35) *implies*

$$K_t = \Phi \sum_{k=1}^{N} \frac{(t\Lambda)^k}{k!} \Phi^\top D = \sum_{k=1}^{N} \frac{(\Phi t \Lambda \Phi^\top D)^k}{k!} = \sum_{k=1}^{N} \frac{(tL)^k}{k!} = e^{tL} \tag{5.108}$$

Hence, for a given $\boldsymbol{f}$*, the heat kernel is expressed by the exponential matrix of the Laplacian*

$$\mathrm{HKS}(\cdot, t) = e^{tL} \boldsymbol{f} \tag{5.109}$$

This clarifies again that HKS and WKS are eigendecomposition methods and approximate the matrix exponential (5.109) *when* $r < N$ *dominant eigenvalues and eigenvectors are used.*

For the sake of completeness, we deal with the proposed improvements (eigenvalue computation, modified initial condition) with regard to the shape signatures which are based on the kernel-based methods.

First, the matching performances of HKS and WKS in relation to the use of GEP and SEP for the dataset *wolf* and *baby* are shown in Figure 5.32. Obviously, the matching results of both kernel-based methods are independent of the specifically computed eigendecomposition. This can be explained experimentally, cf. Figure 5.33, using a test example by means of the HKS technique as follows: the contribution of slightly disturbed eigenvalues especially affects the feature descriptors at very small times t, since the heat signature decays exponentially as t increases and thus converges to a constant temperature. In contrast to the temporal domain $[0, t_r^\star]$ as used within the MCR approach, HKS signatures are built on the scale $[t_a, t_b]$. However, applying this scale neglects exactly those small time samples so that unstable eigenvalue computations have no significant influence on the HKS signature, as shown Figure in 5.33. Although robust techniques are usually considered to be beneficial in practice, this might come at the cost of the matching performance for the PSC application. For similar reasons, the WKS signature is also robust towards unstable eigenvalue computations.

In a second experiment, the results of the modified initial condition (5.78) are displayed in Figure 5.34. Compared to MCR, this modification is not a useful tool in order to improve the kernel-based methods.

To conclude the present discussion, we give an overview of all essential differences between the MCR technique and HKS/WKS in Table 5.1.

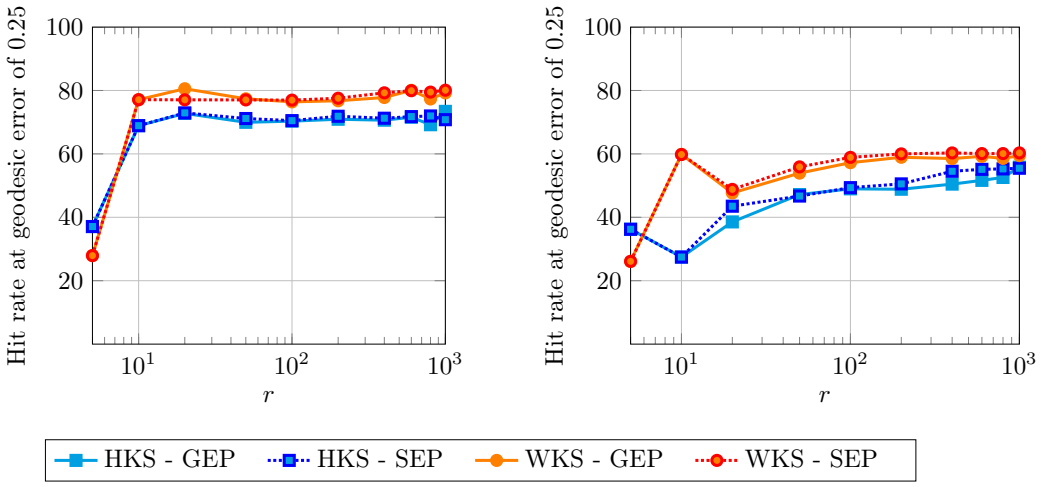


Figure 5.32: Results for the dataset *wolf* (**left**) and *baby* (**right**) using the kernel-based methods. We compare the geodesic error at 0.25 between HKS and WKS for different number of modes $r \in [5, 1000]$. The eigenvalues and eigenvectors are computed by solving the GEP (5.33) and the SEP (5.60). The geodesic error accuracy is generally independent of the underlying eigendecomposition.

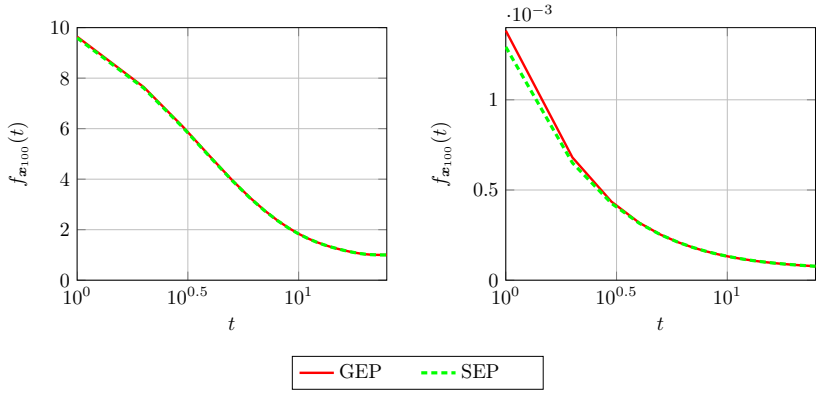


Figure 5.33: Results for the *wolf* dataset using the geometric heat equation. We compare the discrete feature descriptor $f_{\boldsymbol{x}_{100}}(t)$ obtained by the HKS method (**left**) and the optimised MCR technique (**right**) for $r = 100$. The signatures are computed by GEP as well as SEP. For a better comparison the time axes are scaled logarithmically. Obviously, the type of eigendecomposition generally only affects the MCR heat signature.

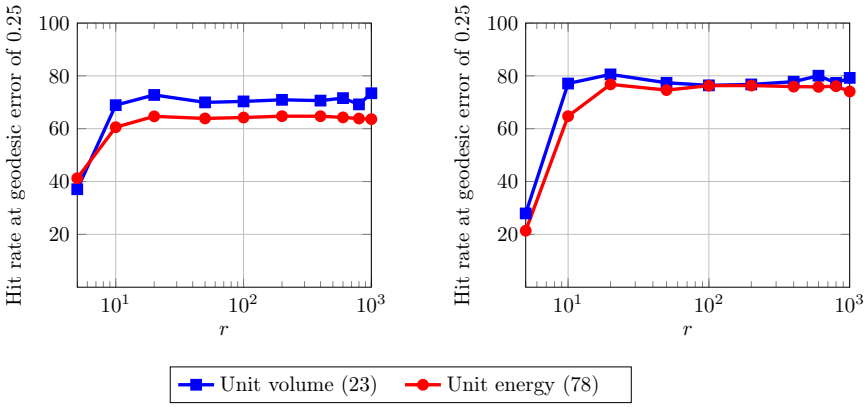


Figure 5.34: Results for the *wolf* dataset using HKS (**left**) and WKS (**right**). We compare the geodesic error at 0.25 between the *original* (5.24) and the *modified* (5.78) initial condition computed by the kernel-based methods for different number of modes $r \in [5, 1000]$. The eigenvalues and eigenvectors are computed by solving the GEP. This type of modification yield worse matching results, especially for HKS.

Table 5.1: Technical differences between MCR and the kernel-based techniques.

Specification	**MCR**	**HKS**	**WKS**
Eigenvalue problem	SEP	SEP/GEP	SEP/GEP
Temporal sampling	uniform	nonuniform	uniform
Time scales on shapes	identical	various	various
Initial condition	unit energy	unit volume	unit volume

5.7.2 Rational Approximants and Krylov Exponential Approximations

Finally, we consider methods for approximating the matrix exponential that belong to the class of spectrum-free computations. As mentioned in Section 2.3.2 popular methods for computing e^{tL} such as the Padé approximation, the scaling and squaring method or the Chebyshev polynomials are generally only effective in computing exponentials of small matrices. Therefore, these methods are usually not suitable for discrete Laplacians with larger resolutions.

In order to tackle this issue and to use the scaling and squaring method efficiently, the authors [285] apply a multi-resolution approach. As a result, this technique can achieve fast approximations, but this approach, along with some theoretical limitations, is much more cumbersome to implement.

In contrast, Patané [208] proposes for applications in shape comparison to compute the wFEM heat kernel via the generalised Chebyshev approximation, in which the matrix exponential is replaced by

$$K_t \boldsymbol{f} \approx \alpha_0 \boldsymbol{f} - \sum_{i=1}^{r} \alpha_i (tL + \Theta_i D)^{-1} D \boldsymbol{f} \tag{5.110}$$

with the poles $\{\Theta_i\}_{i=1}^r$ and the coefficients $\{\alpha_i\}_{i=1}^r$. Then the approximate solution (5.110) is computed by solving sparse linear systems using an iterative solver for $r = 5$ or $r = 7$. Obviously, the Chebyshev method leads to high computational costs in the PSC application, since (5.110) has to be solved for all points on the shape. On closer inspection, this approach represents a higher order scheme and leads to higher computational costs than the proposed first order IE scheme.

In fact, only the product $e^{tL}\boldsymbol{f}$ is required instead of the full exponential matrix. In this situation the Krylov subspace methods introduced in Section 2.3.2 are one of the most efficient methods as used, e.g. for image smoothing [301]. However, as indicated earlier, the convergence of this method depends on the norm of $\|tL\|_2$. In connection with the fact that L is characterised through a wide spectrum of eigenvalues, it is evident that Krylov-based matrix exponential approximations are not practical for the present application. The use of a preconditioner according to [278] is also practically unsuitable. Although the dimension m of the Krylov subspace is often still quite small compared to the basic method, the subspaces have to be constructed on each time level. Contrarily, the KSMOR technique constructs the Krylov subspaces only once.

We therefore consider the proposed KSMOR technique, as described in Section 5.4.3, to be the superior method in the class of spectrum-free computation methods.

5.8 Evaluation of Optimised MCR and Kernel-Based Methods

In this section we give a qualitative evaluation between the developed optimised MCR technique, relying on both geometric heat and wave equation, compared to the kernel-based methods HKS and WKS. To this end, we benchmark the methods at hand of the complete TOSCA dataset [50] whose shapes are almost isometric.

It should be noted again that the comparison of the optimised MCR wave signature and WKS is not caused on the same PDE model, although the notion “wave kernel signature”

suggests this, as the latter method is based on the Schrödinger equation. Concerning this point, let us note first that a recent work [67] has shown that the numerical descriptor[9] based on the geometric wave equation gains better results than the numerical descriptor constructed on the geometric Schrödinger equation. Second, within the class of kernel-based methods the WKS can be considered as a competitive descriptor. Thus, we compare the optimised MCR wave signature in its relation to the WKS due to their distinct role in their respective class.

Evaluation Measure. For the evaluation of the correspondence quality, we use again the *geodesic error*.

Technical Remarks. The TOSCA dataset [50] we investigate includes several classes of almost isometric shapes. In detail, it contains 76 shapes (without gorilla shapes) which are divided into 8 classes (humans and animals) of varying resolution (4K to 53K vertices). Furthermore, for an introductory experiment the datasets *wolf* and *baby* that have already been used are evaluated.

The experimental comparison basically considers the implementations of the kernel-based methods and its parameter settings as described in [14, 267]. For all methods we compute the feature descriptors sampled at 100 points. On this basis, the adapted temporal domain $[0, t_r^\star]$ and the uniform time increment τ are calculated using $t_F = 25$ and $F = 100$.

All experiments were done in MATLAB R2018b with an Intel Xeon(R) CPU E5-2609 v3 CPU. The eigenvalues and eigenvectors are computed by the MATLAB internal function *eigs*. Moreover, the optimised MCR technique and the kernel-based methods are built on solving SEP and GEP, respectively.

5.8.1 Evaluation of the Geodesic Error

The following evaluation is subdivided into two parts. First, we compare the correspondence quality of the optimised MCR technique and the kernel-based methods using the specified datasets *wolf* and *baby*. Second, all methods are benchmarked at hand of the complete TOSCA dataset. Let us emphasise at this point that the numerical advances in the use of the MCR technique as a time integration solver here enable for the first time the evaluation of the entire TOSCA dataset.

Besides the qualitative evaluation, we also shed light on the number of eigenvalues that should be employed within all of the techniques mentioned. This is still an important practical issue for the methods discussed in the PSC application.

Evaluation on Dataset Wolf and Baby The results presented in Figure 5.35 show a significantly higher matching performance using the optimised MCR technique. The MCR signatures outperform their competitive methods HKS and WKS for both datasets. Another interesting aspect is that the MCR heat signature achieves even better results than WKS. In addition, the depicted curves in Figure 5.35 show a saturation behaviour with respect to the number of modes used. This means in particular that the correspondence quality can no longer be significantly improved after a certain number of modes have been used. For the considered two datasets the point of saturation is achieved approximately for $r = 100$ modes.

[9] In this work the underlying PDEs are solved numerically by discretisation in space and time.

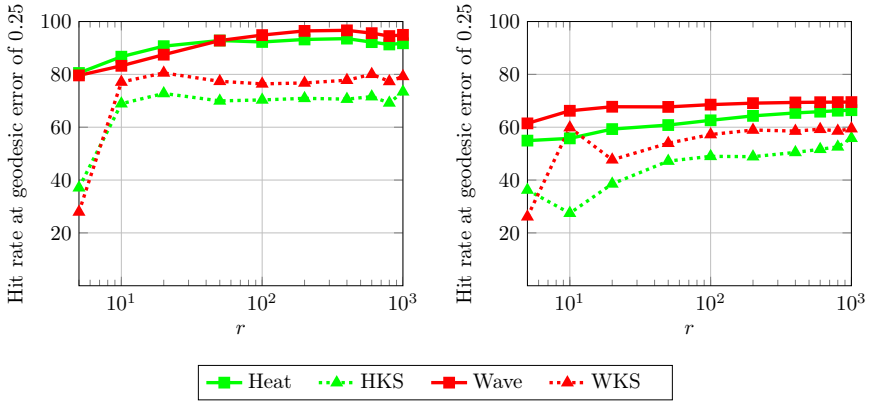


Figure 5.35: Results for the dataset *wolf* (**left**) and *baby* (**right**). Comparison of the geodesic error at 0.25 between the optimised MCR technique and the kernel-based methods for different number of modes $r \in [5, 1000]$. The eigenvalues and eigenvectors for MCR and the kernel-based methods are computed by solving SEP and GEP, respectively. Obviously, our proposed MCR technique clearly outperforms HKS and WKS. It is also interesting that the MCR heat signature gives better results than WKS.

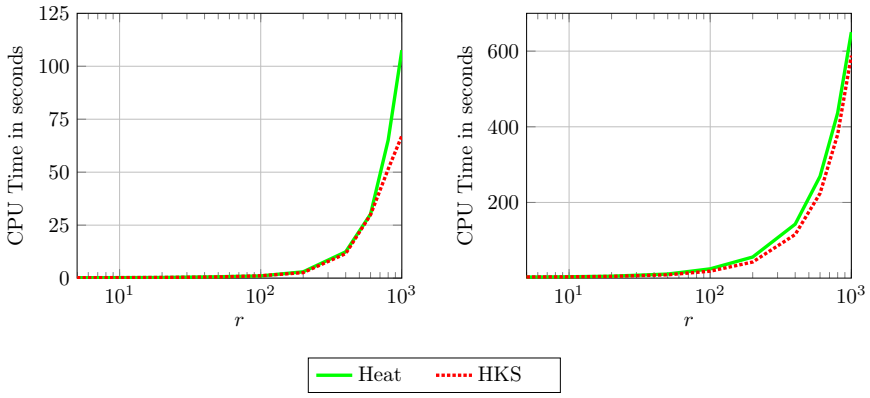


Figure 5.36: Results for the datasets *wolf* (**left**) and *baby* (**right**) using the geometric heat equation. Comparison of the CPU time required between the optimised MCR technique and the kernel-based methods for a varying number of modes $r \in [5, 1000]$. The CPU time of the kernel-based methods is slightly faster, especially for $r > 100$ modes, but the MCR technique is obviously competitive. Using the geometric wave equation has the same costs.

For the sake of completeness, we also examine the CPU time required for the methods used on the basis of the *wolf* and *baby* dataset. As shown in Figure 5.36, both techniques achieve almost equally fast CPU times.

Evaluation on TOSCA Dataset First, let us discuss here the number of the ordered modes for the methods used. In many publications on the kernel-based methods this number is manually set to a fixed value, e.g. $r = 300$ is often used. Because of their influence on computational efficacy, we were motivated to consider the examined measurements and the amount of eigenvalues used for the TOSCA dataset, see the results illustrated in Figure 5.37.

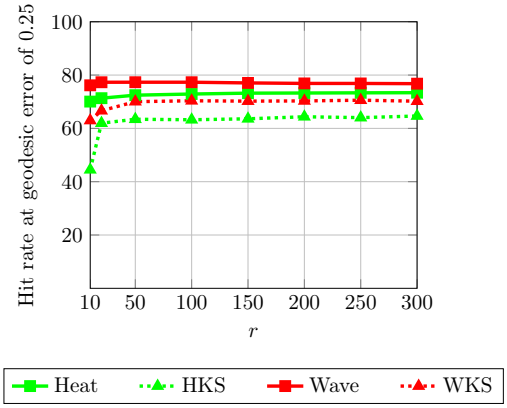


Figure 5.37: Results for the TOSCA dataset. Comparison of the geodesic error at 0.25 between the optimised MCR technique and the kernel-based methods for a varying number of modes r. The eigenvalues and eigenvectors for MCR and the kernel-based methods are computed by solving SEP and GEP, respectively. Again, the proposed MCR technique achieves a better performance than HKS and WKS.

Over the whole TOSCA dataset, we obtain qualitatively identical results as for the examples *wolf* and *baby*. The proposed MCR technique outperforms the kernel-based methods in terms of the geodesic error. More precisely, for both PDEs the optimised MCR method provides around 5-10% higher correspondence quality than the corresponding kernel-based approach. Again, the MCR heat signature generally slightly outperforms WKS. It should also be emphasised at this point that the MCR signatures produce a high matching performance compared to the kernel-based methods even for a small number of $r = 10$ modes used.

Moreover, the experiment additionally indicates that a saturation behaviour is achieved at a small spectrum of $r \approx 50$ modes for all methods. In total, by using a small spectrum the computational effort can be notably reduced without losing performance, which is particularly important in the case of highly resolved meshes. Of course, fast approximations of the Laplace-Beltrami eigenproblem are useful when considering high resolutions. In fact, a recent work [191] has been introduced an efficient approximation for the lowest part of the Laplacian spectrum which perfectly fits into the spectrum-based concept.

For the sake of documentation, we finally evaluate the geodesic error within the range $[0, 0.25]$ for all methods using $r = 50$ modes as visualised in Figure 5.38. As expected, the outcomes are the same to those as before, in particular the MCR wave signature is superior to all other methods.

KSMOR Technique on TOSCA Dataset As already noted, we consider the KSMOR technique to be superior in the class of the spectrum-free computation methods for the PSC application. First tests in Section 5.5 have shown that KSMOR achieves a high matching performance linked with a fast CPU time. Therefore, it is essential to give an evaluation of the KSMOR method using the complete TOSCA dataset. The corresponding results are visualised in Figure 5.39. Compared to the spectrum-based methods, the matching performance on the basis of the geodesic error at 0.25 is slightly lower. In contrast, KSMOR achieves a higher accuracy in terms of the geodesic error up to around 0.05.

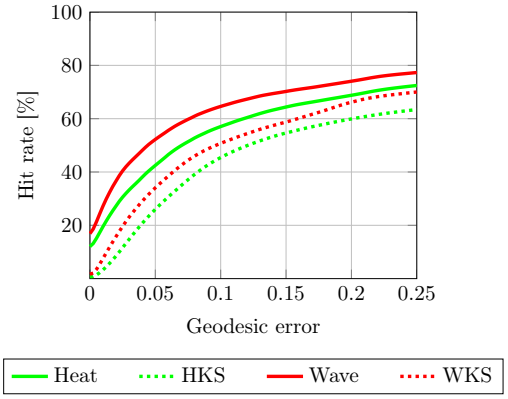


Figure 5.38: Results for the TOSCA dataset. Comparison of the geodesic error for the interval $[0, 0.25]$ between the optimised MCR technique and the kernel-based methods for $r = 50$ modes. The proposed MCR technique achieves a higher geodesic error accuracy than HKS and WKS.

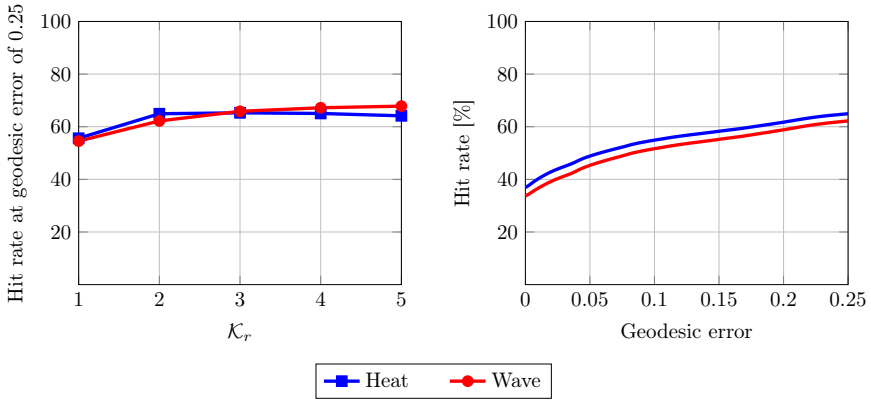


Figure 5.39: Results for the TOSCA dataset. Comparison between the numerical heat and wave signatures based on the KSMOR technique with $\sigma = 0.1$. **Left:** Comparison of the geodesic error at 0.25 for a varying number of Krylov subspaces $\mathcal{K}_r$. **Right:** Comparison of the geodesic error for the interval $[0, 0.25]$ using $r = 2$ subspaces. On the basis of KSMOR, both geometric PDEs achieves almost the same matching performance.

5.8.2 Evaluation Based on Mapping Indicator Functions

All previous results were based on the geodesic error, especially at the hit rate of 0.25, but without some kind of a visual matching comparison or an explicit rating with respect to the point-to-point[10] correspondence quality. However, this is a very important matter and demonstrate the practical usability of a method in the context of the PSC application. To this end, we additionally compare all techniques using mapping indicator functions that are

[10] Even nowadays, the correspondence of two nonisometric shapes as an assignment problem, which can be interpreted as the matching of pointwise descriptors, is a highly interesting area of research, see e.g. [288] and the references therein. We do not address this issue in this thesis, as we are more interested in the efficient computation of precise feature descriptors than in the assignment problem.

defined for a specific region of a shape. A brief description of the procedure is given below.

Let us recall the correspondence map S which was essentially introduced in Section 5.2.3. For two discrete shapes $\mathcal{M}$, $\widetilde{\mathcal{M}}$ with associated point clouds $P = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$, $\widetilde{P} = \{\widetilde{\boldsymbol{x}}_1, \dots, \widetilde{\boldsymbol{x}}_{\widetilde{N}}\}$, the matrix representation of the correspondence map $S_{\{0,1\}} : \widetilde{P} \times P \to \{0, 1\}$ can be encoded by using

$$S_{\{0,1\}_{ji}} = \begin{cases} 1, & \text{if } d_f(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j) \leq d_f(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_k), \quad k = 1, \dots, \widetilde{N} \\ 0, & \text{else} \end{cases} \tag{5.111}$$

where $S_{\{0,1\}}$ is a binary assignment matrix of size $\widetilde{N} \times N$. For finding a single corresponding counterpart, e.g. of $\boldsymbol{x}_i \in P$ on $\widetilde{\mathcal{M}}$, we construct a N-dimensional vector-valued indicator function $\boldsymbol{h} : P \to \{0, 1\}$ with

$$\boldsymbol{h}_i(\boldsymbol{x}_k) = \begin{cases} 1, & \text{if } i = k \\ 0, & \text{else} \end{cases} \tag{5.112}$$

By performing the matrix-vector multiplication $\widetilde{\boldsymbol{h}}_j := S_{\{0,1\}}\boldsymbol{h}_i$, we obtain the indicator vector $\widetilde{\boldsymbol{h}}_j$ for the case the tuple $(\boldsymbol{x}_i, \widetilde{\boldsymbol{x}}_j) \in P \times \widetilde{P}$ is a corresponding pair. However, due to strong elastic deformations, noisy shapes or intrinsic symmetries (i.e. inherent ambiguities) the construction (5.111) may result in a misleading matchings. Therefore, the correspondence map $S_{\{0,1\}}$ is in practice neither injective nor surjective, even if $N = \widetilde{N}$.

By allowing values between 0 and 1, the correspondence map can be cast as a *soft correspondence map* $S_{[0,1]} : \widetilde{P} \times P \to [0, 1]$, as used for example in [83], which can be interpreted as a correspondence probability using normalisation

$$\sum_{i=1}^{N} S_{[0,1]_{ji}} = 1, \quad j = 1, \dots, \widetilde{N} \tag{5.113}$$

For a certain point $\boldsymbol{x}_i \in P$ on the reference shape, the map encodes the transition probabilities to the points on $\widetilde{P}$ based on the feature distances, whereby high probabilities correspond to low feature distances and vice versa. The arising soft correspondence map allows us to express the probability transition of all points from a reference shape onto a target shape. Therefore, this framework may enable the detection of specific geometric regions. The key idea is that intrinsic feature descriptors have almost identical low feature distances for whole regions where the intrinsic geometry is similar. Using the soft correspondence map, we compare the methods when dealing with problems such as the correspondence of shape segments. To this end, a discrete indicator function $\boldsymbol{h}$ is constructed, e.g. for the tail of the *wolf* shape with

$$\boldsymbol{h}(\boldsymbol{x}_i) = \begin{cases} 1, & \text{if } \boldsymbol{x}_i \in \text{tail} \subset \mathcal{M} \\ 0, & \text{else} \end{cases} \tag{5.114}$$

By performing $\widetilde{\boldsymbol{h}} := S_{[0,1]}\boldsymbol{h}$, the entries of $\widetilde{\boldsymbol{h}}$ contain the probability of the matched indicator function on the transformed shape, which is exemplarily presented in Figure 5.40.

In what follows we consider for a qualitative comparison in particular the tail, the nose, the front left paw and the left ear of the *wolf*. The corresponding indicator functions are visualised in Figure 5.41.

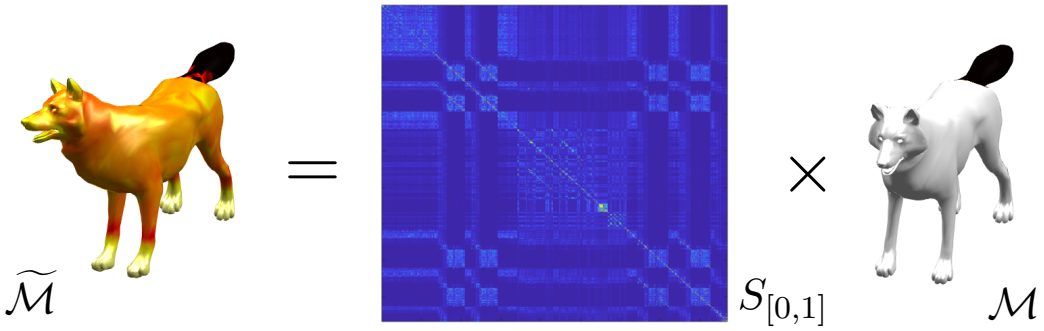


Figure 5.40: Idea of mapping indicator functions: an indicator function (visualised by a colour map on the shape) $\boldsymbol{h} \in \mathcal{M}$ is defined such that it is one at the tail of the dog and zero else, cf. (5.114). Using the soft correspondence matrix, we exemplarily define the probability of the matched indicator function $\widetilde{\boldsymbol{h}} \in \widetilde{\mathcal{M}}$ by performing $\widetilde{\boldsymbol{h}} := S_{[0,1]}\boldsymbol{h}$. The probability of matching the indicator function on the tail of the transformed *wolf* $\widetilde{\mathcal{M}}$ is very high, while the colour encodes the probability ranging from white (almost zero probability) to black (high probability). Since the ground truth is the identical labelling (i, i), the soft correspondence matrix has a diagonally dominant structure.

Figure 5.41: Mapping of indicator functions defined on certain regions of the wolf shape. From left to right the indicator function are triggered (to be 1 - black colour) at the wolf's tail, nose, front left paw and left ear. Let us note that the other colours (except black and white) are due to the plotting properties of graphics in MATLAB.

Results on Shape Segments of Wolf Dataset The correspondence quality is measured built on the soft correspondence matrix $S_{[0,1]}$. The realisation of $S_{[0,1]}$ can easily be done in the following way: the computed feature distances can be normalised and reformulated as probabilities, so that low distances correspond to high probabilities.

In a first experiment, we evaluate the correspondence quality based on the geometric heat equation using the direct solver, see Figure 5.42. The same graphic also shows the matching results with respect to the KSMOR solver with $\sigma = 0.1$ and using $r = 2$ Krylov subspaces. Obviously, both methods perform very well and capture the sought region on the transformed

shape (indicator functions $\boldsymbol{h}$ and $\widetilde{\boldsymbol{h}}$ are very similar). On closer examination, the KSMOR technique indicates a minor correspondence quality compared to the direct solver for the indicator function belonging to the ear region. Nevertheless, KSMOR is much more efficient in computing the numerical signatures and can save around 95% of the CPU time in relation to the direct solver, as we have shown in Section 5.5. Let us mention that the same quality outcome is obtained for the geometric wave equation.

The latter experiment is also performed for the optimised MCR technique and the kernel-based methods using $r = 50$ modes. First, the results for the geometric heat equation are illustrated in Figure 5.43. Compared to the direct solver and KSMOR, the spectrum-based methods achieve in part significantly different results. The matching probability of the indicator functions of the transformed *wolf* is lower and, in general, more scattered over the whole shape. However, it is evident that the MCR technique realises a much higher correspondence quality than HKS, especially for the indicator function related to the tail, nose and paw. An interesting aspect is that the MCR approach has fewer problems with intrinsic symmetries than HKS, cf. the front left paw. Because the HKS method, in contrast, identifies all four paws with a high degree of probability (darker colour).

The correspondence quality of MCR and WKS based on the geometric wave and the Schrödinger equation, respectively, are visualised in Figure 5.44. Once again the MCR technique achieves a much higher correspondence quality than WKS, especially for the indicator function with respect to the tail and nose. In contrast to the geometric heat equation, the wave signatures show a less precise matching performance for the detection of the front left paw.

In total, the last experiments using mapping indicator functions have clearly shown that the KSMOR technique can provide a high correspondence quality. In contrast, the results of the spectrum-based methods are less good due to the use of low-frequency modes, since small details (which are described by high-frequency) are generally more difficult to capture. Nonetheless, the use of soft correspondence maps in combination with the MCR technique may provide very beneficial results compared to HKS and WKS.

Future Work on Soft Correspondence The soft correspondence map $S_{[0,1]}$ is a dense matrix of size $\widetilde{N} \times N$ that becomes increasingly cumbersome for large $\widetilde{N}$ and N. In particular, storing $S_{[0,1]}$ might be computationally expensive for large shapes. In addition, the soft correspondence map could be used to extract the ideal binary assignment matrix $S_{\{0,1\}}$ from itself using a more sophisticated (optimisation) approach, but this task is not trivial in practice. For both issues it would be desirable to reduce the information contained in the dense soft correspondence matrix and therefore to give $S_{[0,1]}$ ideally a sparse structure while keeping its useful meaning for PSC. The sparse structure appears to be a reasonable and computationally efficient compromise between the dense soft correspondence matrix and the ideal binary assignment matrix.

A simple strategy to increase the sparsity could be for instance to set iteratively small entries of $S_{[0,1]}$ (with low probability) in ascending order to zero until a specific tolerance is reached. However, the essential building block for a possible extraction of the binary assignment matrix $S_{\{0,1\}}$ is to develop a method that could make full use of the soft correspondence information, which may result in a quite powerful approach. Obviously, this is not trivial and beyond the scope of this work.

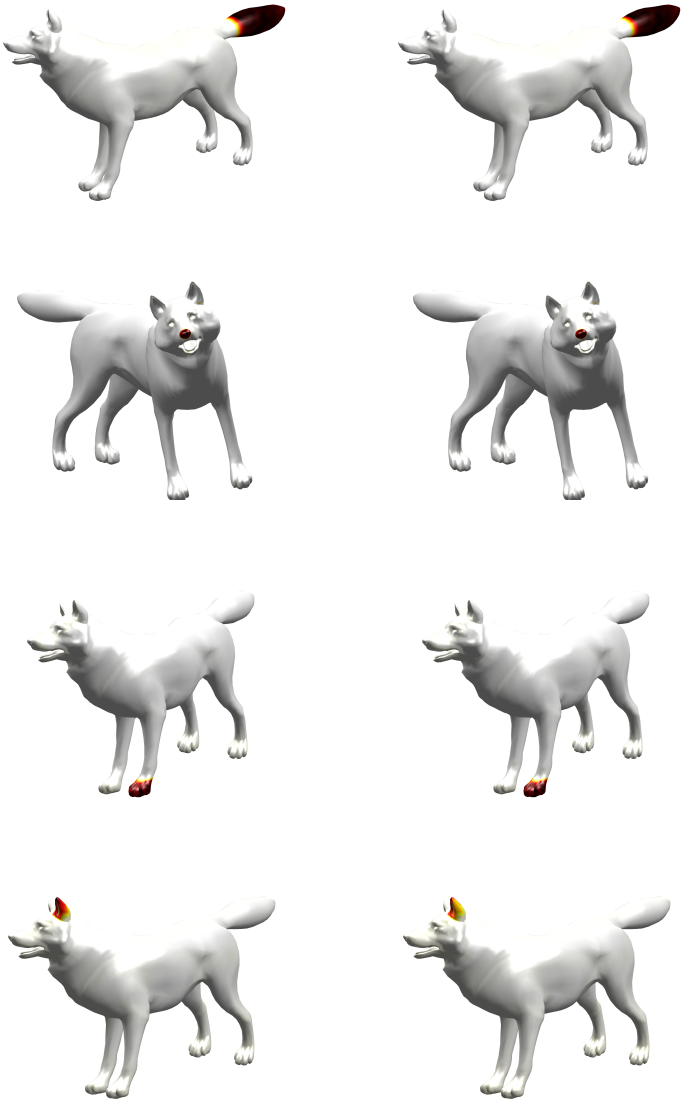


Figure 5.42: Results for the wolf dataset using the geometric heat equation. Comparison of the mapping of the indicator function defined on a certain region of the wolf shape (from **top** to **bottom**: tail, nose, front left paw, left ear) to a transformed version of it between the direct solver (**left**) and the KSMOR technique (**right**) with $\sigma = 0.1$ and $r = 2$. For reasons of clarity, the shapes are rotated for detecting the specific shape segments. The colour encodes the probability of the matching, being black for a high probability and white for almost zero probability. Both methods perform very well and generally capture the sought region on the transformed shape. However, KSMOR is much more efficient in computing the numerical signatures. The same quality results are obtained using the geometric wave equation.

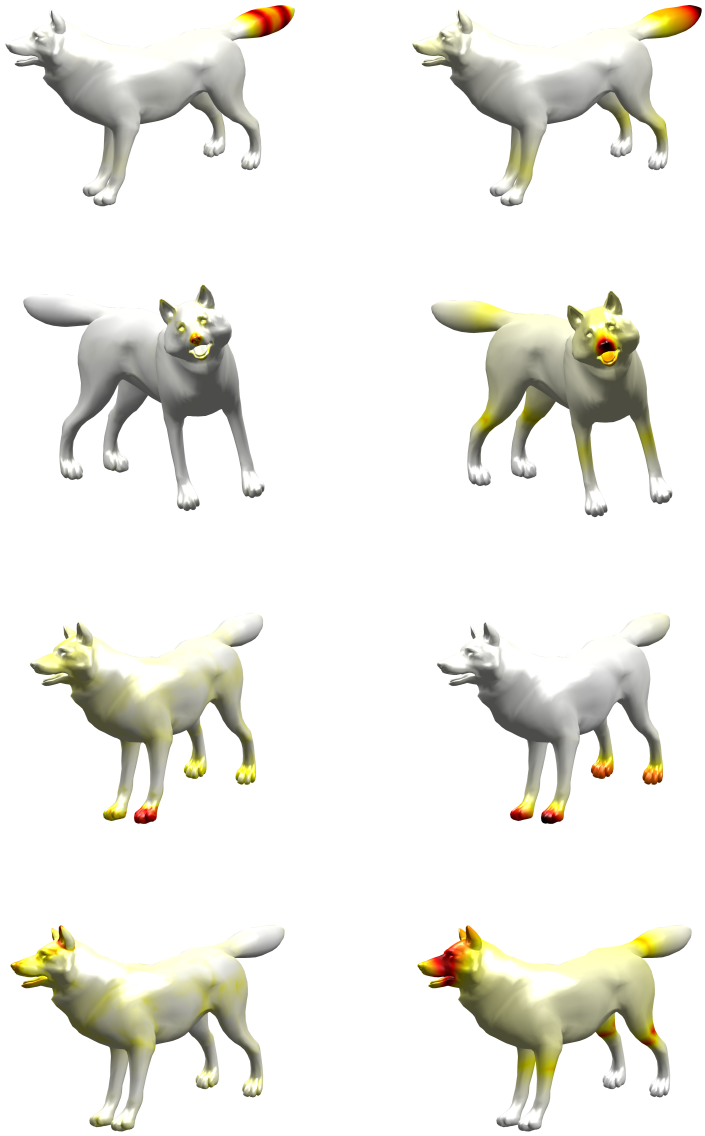


Figure 5.43: Results for the wolf dataset using the geometric heat equation. Comparison of the mapping of the indicator function defined on a certain region of the wolf shape (from **top** to **bottom**: tail, nose, front left paw, left ear) to a transformed version of it between the optimised MCR technique (**left**) and HKS (**right**) using $r = 50$ modes. The colour encodes the probability of the matching, being black for a high probability and white for almost zero probability. For reasons of clarity, the shapes are rotated for detecting the specific shape segments. Obviously, MCR achieves better matching performance, especially for the regions - tail, nose and paw.

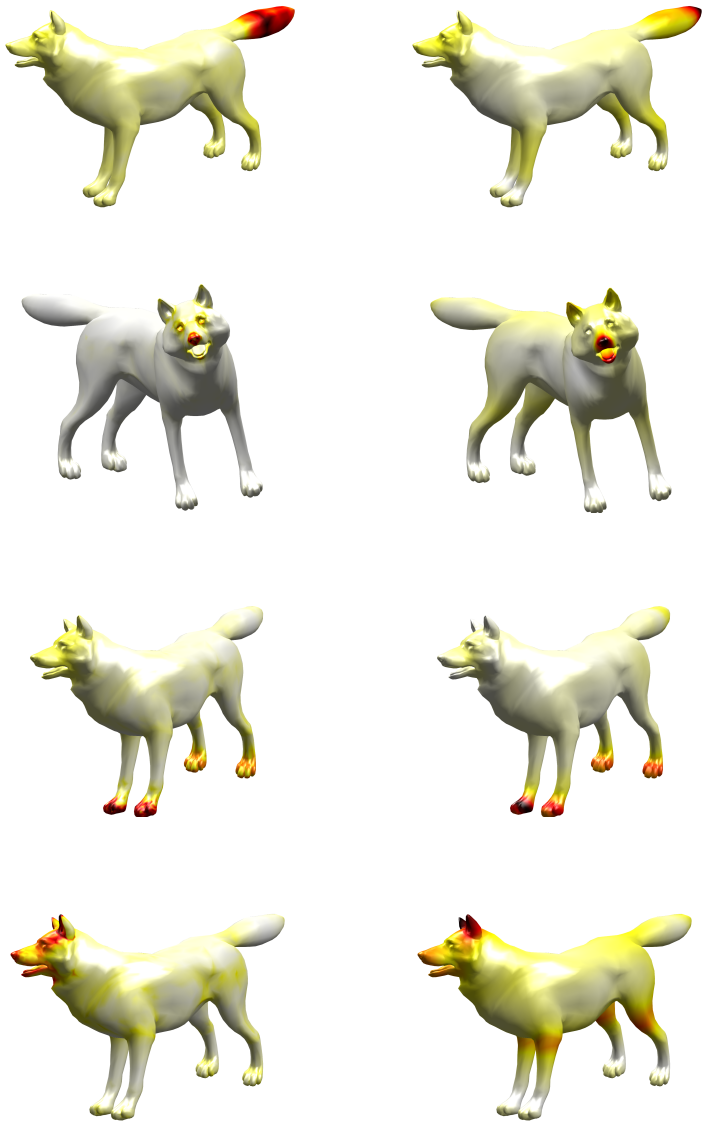


Figure 5.44: Results for the wolf dataset using the geometric wave equation. Comparison of the mapping of the indicator function defined on a certain region of the wolf shape (from **top** to **bottom**: tail, nose, front left paw, left ear) to a transformed version of it between the optimised MCR technique (**left**) and WKS (**right**) using $r = 50$ modes. The colour encodes the probability of the matching, being black for a high probability and white for almost zero probability. For reasons of clarity, the shapes are rotated for detecting the specific shape segments. In general, MCR achieves better matching performance, especially for the regions - tail and nose.

5.9 Summary

We have extended the numerical framework that has been presented in earlier literature [67,68], and in our opinion, we have tweaked with this chapter the MOR approach very close to its limit with regard to its use in the PSC application. On the basis of numerical time integration, we have developed the spectrum-free KSMOR technique and the spectrum-based optimised MCR approach, which are of beneficial use compared to the state-of-the-art solvers in the class of time-evolution method. We have demonstrated that the efficient KSMOR method can achieve a high correspondence quality using mapping indicator functions. In contrast, the optimised MCR technique is highly efficient for applications with high resolutions and clearly outperforms their direct counterparts HKS and WKS with respect to the geometric error, but also for the detection of specific geometric regions. To achieve this, we have combined several numerical techniques related to MCR with benefit that result in an algorithm that is ultimately easy to implement.

Let us stress that our approaches are nearly free of parameters. One can even conclude for the remaining few parameters like e.g. r the number of eigenvalues or Krylov subspaces that we have shown experimentally how to choose them in applications, so that in practice our approach can be considered as parameter-free, which is of high practical value.

We think that the use of KSMOR and MCR for particular tasks in shape analysis such as shape detection could be a promising subject of future research. Furthermore, the use of soft correspondence maps as introduced here appears to be promising. However, for the practical use of a sparse soft correspondence matrix a sophisticated method has to be developed that could make full use of this sparse information.

We also emphasise that as part of the functional maps pipeline mentioned in related work, feature descriptors are used to compute a coarse correspondence between points. Providing accurate initial correspondences based on feature descriptors improve the performance and CPU time of dense shape correspondence algorithms relying on functional maps, as reported in [179]. We conjecture that our methods may be useful in this framework as well, and therefore this topic will be of interest in the future.

Chapter 6

Efficient Long-Term Simulation of a Geothermal Energy Storage

In this chapter we discuss particular challenges of numerical methods arising in connection with long-term simulation of a *geothermal energy storage (GES)*. Long-term evolutions of parabolic PDEs such as the heat equation are the subject of interest in many applications. There are several numerical solvers marking the state of the art in diverse scientific fields that may be employed with benefit for the simulation of such long-term scenarios. However, long-term simulation of technical models often requires numerical methods that are specifically designed for the intended purpose.

The main goal of this chapter is to investigate which numerical methods are suitable for long-term simulations as appearing in real-world applications as considered here, and how they need to be used for the fundamental problem of heat evolution with internal and external boundary conditions as well as source terms. This problem arises in GES, for which we provide here a comprehensive analytical and numerical model. In order to provide an efficient and accurate enough simulation, we give a thorough discussion of the various numerical solvers along with many technical details and adaptations. In this context, let us emphasise that the methods we rely on already exist in previous literature, but we show how to use them in order to obtain efficient schemes for the GES application.

In our investigation, we focus on two largely competitive approaches, namely the FEDRK method originating in image processing and the KSMOR technique. Considering our GES application, we will precisely elaborate the complete continuous model and the corresponding discretisation which is an important component for the numerical realisation. In particular, the matching conditions at the interfaces that occur leading to a large-scale input, which is why special care is required when using the KSMOR technique.

Under these circumstances, it is essential to adapt the original KSMOR technique for practical use. We validate our numerical findings at the hand of two experiments using synthetic and real-world data, and show that one can obtain fast and accurate long-term simulations of typical GES facilities.

Since the long-term simulation of a three-dimensional GES is linked to extreme computational costs, it is of high relevance if the model dimension itself could be reduced for computational purposes. We will demonstrate that the application of a two-dimensional linear heat equation is absolutely sufficient for the long-term simulation considered in our purpose. The latter is validated on real-world data using temperature probes of a real three-dimensional test field. The simplified and dimensionally reduced model can then be used in practice instead of the real (nonlinear) model either for simulation purposes or for parameter optimisation.

Let us note that a part of this chapter was already presented in our work [17][1]. To improve the performance of the FED method, which has shown some promising first results, the FEDRK scheme is employed. In doing so, we show how to use the FEDRK method for parabolic problems including sources/sinks, which demonstrates the wide applicability of this technique also in connection with engineering problems. In addition, we modify the original KSMOR technique for systems with a large number of inputs in a similar manner as proposed in [232]. In this framework, we provide here to our best knowledge the first very detailed exposition and comparison of this kind of approach.

In total, we give a detailed overview of relevant and modern numerical solvers that can be helpful for tackling long-term heat evolution in many engineering fields.

Chapter Organisation First, a brief introduction to the GES application is given in Section 6.1. The Section 6.2 contains the continuous model description including the modelling of the external and internal boundary conditions, and we also inform about the generation of the initial heat distribution. Then we recall the numerical realisation by spatial and temporal discretisation in Section 6.3. In Section 6.4 a detailed overview of the numerical solvers is given. To be more precise, we discuss the methods: FED, FEDRK, linear system solvers, KSMOR and its adapted variant KSMOR*. The experimental evaluation presented in Section 6.5 focuses on the simulation quality and efficiency of the numerical solvers by comparing at hand of two GES experiments. This chapter is concluded with a summary.

6.1 Introduction

Alternatives to fossil fuel resources are becoming increasingly important. Apart from an efficient energy generation it is also important to store it, ideally with minimal losses over long periods of time. The recent GES technology represents a potentially very attractive approach to energy storage. The GES is implemented in natural underground sites, for instance using large soil tanks. Such tanks are partially surrounded by insulating walls and, depending on their depth, soils with different heat conduction and capacity properties. It is a very cost effective technology that can be used in both new construction and refurbishment.

In contrast to classic energy storage approaches, in which tanks are employed that are fully closed and do not interact with their environment, the technical realisation of GES is characterised by a downwardly open heat tank. This is one of the aspects that makes the technology highly cost effective as it is not required to excavate the ground to a large degree. It is sufficient to excavate a relatively small area that will represent the tank, while staying thereby close to the surface. In the resulting pit the heating pipes are installed, and one has to insulate the walls (excepted the bottom side) of the pit for instance by use of styrofoam. Afterwards, the pit may typically simply be filled with the original soil. The second aspect that makes the GES technology cost effective compared to traditional heat storage in a closed tank is given by considering the dimensions that are needed to provide sufficient capacity for storing energy in practice. The advantage of the open tank is that the heat energy is effectively stored by the earth in and below the tank, making its capacity extremely high in

[1] In our previous four-page conference paper, we have shown some preliminary results encompassing a simple synthetic experiment, thereby comparing the performances of the IE method and the FED scheme.

real applications and providing in practice a multiple of the capacity that is making up the actual tank.

We are particularly interested in the potential of GES to store excess energy generated during the summer, e.g. by the use of solar cells, for heating in winter. This becomes a very important issue when the entire heating system is considered over a year or even several years, depending on weather conditions, day, night and weekly rhythm. Even though already working GES exist, especially for single-family homes and smaller office buildings, well-founded evidence or simulations are required to assess the profitability of a GES. This is of great importance for the optimal dimensioning of the heat tank, which is the most expensive design factor. Apart from that, this will also be very important in order to adopt this technique for large office complexes.

To study the reliability of such systems it is important to know how they behave over long time spans for several months or years. The problem can be formulated in terms of a parabolic PDE model given by the heat equation related to space and time. In order to describe the heat transfer in the GES set-up adequately enough, the mathematical model including contact and boundary conditions must first be described. Then the simulation of heat evolution is run as long as the user requires. At this point, it should be emphasised that the simulation may not be performed offline, as the planning of a GES typically requires communication between engineers, architects and the customer, often directly on construction sites. The simulation can therefore ideally be performed online in a few minutes at most during the discussion.

The long-term heat evolution in a geothermal setting can often be modelled on the basis of assuming a homogeneous and isotropic setup, corresponding to a linear PDE. The simulation of linear heat transfer is of fundamental importance in diverse scientific fields. However, even nowadays it is still a challenging task to specify a numerical method that combines reasonable accuracy and computational efficiency. In particular, the computing power of a solver is a key requirement when working with multi-dimensional problems. The long-term integration that needs to be performed in order to simulate seasonal energy storage represents an additional challenge. Standard methods are not devised for combining high efficiency and accuracy in time, so that an analysis of time integration methods for the heat equation in the context of GES is absolutely essential.

There exist countless methods for solving the linear heat equation. As examples, we mention finite difference, finite volume or finite element methods that discretise the spatial dimensions into ODEs. After the spatial discretisation, the temporal integration can either be done explicitly or implicitly. Explicit schemes are based on simple sparse matrix-vector multiplications in which the allowed time step size has a rather small upper bound, rendering the explicit strategy unsuitable for long-term evaluations. On the other hand, the use of implicit schemes leads to the task of solving a system of linear equations in each time step, whereby the number of variables related to the multi-dimensional GES may extend to several hundreds of thousands or perhaps millions. The implicit schemes do not suffer from restrictions on the time step size in theory, but a fast solver for large sparse systems of linear equations is needed. Another numerical aspect that has to be kept in mind when using implicit methods for applications with source terms is that the contributions of the sources must be updated at relatively small time intervals for obtaining an accurate simulation.

The classic solvers such as explicit schemes, sparse direct or sparse iterative solvers are relatively simple to implement or are based on existing sophisticated software packages. As

is known, their efficiency is related to severe time step size restrictions in the case of explicit methods, or high effort for solving large systems of linear equations (implicit methods). For making these approaches efficient enough and to tackle our application, two popular techniques in their respective scientific fields, FEDRK and KSMOR, can be helpful to reduce the computational effort significantly compared to conventional methods.

The FEDRK approach, originating from image processing and belongs to the class of fast explicit methods, combines the advantage of an explicit evaluation with the possibility of achieving high integration times in just a few steps of evaluation. As a result, FEDRK is much more efficient than the usual explicit scheme and simultaneously is based on cheap matrix-vector multiplications. The MOR methods represent another possible approach to reduce the computational complexity of heat simulations. Such techniques can be applied to approximate the underlying ODE system by a significantly reduced system, that is much faster to solve due to its reduced dimension. To solve large-scale problems, the powerful KSMOR methods are most frequently used in this area [29, 123, 140, 150, 169, 232, 261] due to their superior numerical efficiency. Although the basic aspects of the KSMOR technique are well understood, it is not easy to devise it in a way that yields an efficient scheme for resolving heat transfer if internal boundary conditions with a high number of inputs are involved, as in our case.

We note again that there exist other popular strategies to speed up diffusion processes such as operator splitting methods, e.g. the ADI method. The main idea of these techniques is to split the multi-dimensional problem into several one-dimensional problems which can then be efficiently solved using tridiagonal matrix algorithms. However, this approach induces a splitting error that increases with the magnitude of the time step size τ. Moreover, external and internal boundary conditions have to be treated very carefully which makes it difficult to apply these methods in our setting.

6.2 Continuous-Scale Mathematical Model

As indicated, the GES concept is specially designed for seasonal heat storage, so that long-term simulation is a particular challenge. The heat transfer in the near-surface ground is mainly characterised through heat conduction, heat radiation, advection, convection, evaporation, seepage water and geothermal heat. The most important transport mechanism in the ground is usually heat conduction. The associated PDE, called the heat equation, is a classic and thoroughly studied equation. A complete setup that covers a realistic scenario also includes heat sources and sinks, multiple boundary conditions and other physical properties such as the heat dissipation, which may vary in space and time. The basic model equation that we consider is therefore given by

$$\rho\, c\, \partial_t u(\boldsymbol{x}, t) = \operatorname{div}\left(\lambda \nabla u(\boldsymbol{x}, t)\right) + f(\boldsymbol{x}, t), \quad (\boldsymbol{x}, t) \in \Omega \times [0, t_F] \tag{6.1}$$

with thermal conductivity λ, density ρ, specific heat capacity c and heat source/sink f. In general, the physical quantities λ, ρ, c may vary in space and time, and may depend on the current temperature u in this equation. Furthermore, various boundary conditions including Dirichlet-, Neumann- and Robin boundary conditions are required. Indeed, the different regions depicted in Figure 6.1 consist of different media and thus their heat transfer properties are different. This implies that interface boundary conditions are needed in our model.

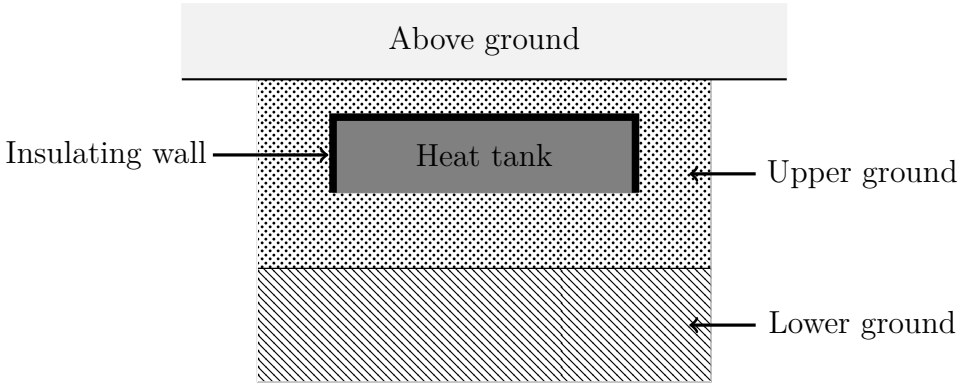


Figure 6.1: Cross section as a schematic representation of a 3D-GES. The upper ground often has different properties than the lower ground. The heat tank is filled with upper ground, contains a heat source/sink and is downwardly open. Accordingly, the heat flows into the tank from below or vice versa. Each interface exhibits different transition properties that need to be modelled into the heat flow equations.

6.2.1 Basic Model for Describing the Geothermal Energy Storage

As already mentioned, we aim to tackle the long-term evolution of heat in a GES as sketched in Figure 6.1. In the following it is assumed that the physical variables λ, ρ and c are constant and nonzero. These assumptions are reasonable in our near ground scenario even though the physical quantities are not perfectly constant. However, the temperature fluctuations are so small, in the regime we are interested in, that they may hardly cause any difference in the solutions of the PDE. In addition, we assume the surrounding soil to be nonporous and that seeping rainwater has no influence on the long-term evolution of the GES. The latter influencing factor has only a short-term effect on the underground temperature, see [253].

Lastly, an important condition within the GES model is related to the groundwater, which may have a strong impact on the temperature distribution in the underground. In this work, it is assumed that the flowing groundwater has a large distance to the heat tank, so that one can specify isothermal boundary conditions for the lower ground boundary in our three-dimensional model. The latter assumption not only enables the consideration of a pure heat equation without a convection term, but on this basis we may also neglect the percentage of water in the soil for specifying of heat conductivity.

These assumptions ensure that the surrounding soils can be considered as homogeneous isotropic media. With this setup we define the thermal diffusivity $a := \frac{\lambda}{\rho c}$ and (6.1) can be rewritten into the following linear model PDE:

$$\partial_t u(\boldsymbol{x}, t) = a\Delta u(\boldsymbol{x}, t) + \frac{f(\boldsymbol{x}, t)}{\rho c}, \quad (\boldsymbol{x}, t) \in \Omega \times [0, t_F] \tag{6.2}$$

which has to be considered for each region sketched in Figure 6.1. The function f represents various heat sources and sinks. Furthermore, an initial heat distribution $u(\boldsymbol{x}, 0)$ and boundary conditions (interface, border) are required as explained below.

6.2.2 Modelling of Interface Conditions

Concerning the interface conditions set-up, we follow the general modelling framework as described for instance in [15]. The interface $\boldsymbol{x} = \boldsymbol{x}_i$ between two solids, where no heat is lost when considering the flux between them, can be modelled by means of the following boundary conditions:

$$\lambda^{(k)} \left(\partial_{\boldsymbol{n}} u^{(k)} \right) \Big|_{\boldsymbol{x}_i} = \lambda^{(l)} \left(\partial_{\boldsymbol{n}} u^{(l)} \right) \Big|_{\boldsymbol{x}_i} \tag{6.3}$$

Here, $u^{(k)}$ and $u^{(l)}$ are the solutions for the individual regions and where $\partial_{\boldsymbol{n}}$ denotes the derivative in the outer normal direction.

With special attention to the contact between the upper and the lower ground, we assume that the contact is perfect and the heat transfer is continuous, so that it additionally follows at the interface

$$u^{(k)}(\boldsymbol{x}_i, t) = u^{(l)}(\boldsymbol{x}_i, t) \tag{6.4}$$

In contrast, the interface in which the insulating walls are involved is subjected to thermal contact resistance, which can generally lead to a discontinuity in heat transfer, and thus yields the condition

$$\lambda^{(k)} \left(\partial_{\boldsymbol{n}} u^{(k)} \right) \Big|_{\boldsymbol{x}_i} = \alpha_c \left(u^{(l)} - u^{(k)} \right), \quad \lambda^{(l)} \left(\partial_{\boldsymbol{n}} u^{(l)} \right) \Big|_{\boldsymbol{x}_i} = \alpha_c \left(u^{(l)} - u^{(k)} \right) \tag{6.5}$$

with a contact heat transfer coefficient $\alpha_c \geq 0$. The latter equation implies two properties. For $\alpha_c = 0$, homogeneous boundary conditions are obtained whereby no exchange of heat exists between two solids. In contrast, for $\alpha_c \to \infty$, the temperature profile between two solids is continuously being.

Consequently, the interaction between different types of soils at interface $\boldsymbol{x}_i$ can be modelled by the equations (6.3)-(6.4). Instead, the interface between soil and insulation is modelled via (6.3) and (6.5).

6.2.3 Modelling of Boundary Conditions

External boundary conditions must also be specified within the model. Therefore, we have to fix conditions situated on $\partial\Omega$, actually at the top, at the bottom and at the sides.

The upper domain boundaries (i.e. between upper and above ground, see Figure 6.1) are characterised such that the soil of the upper ground is not covered by other structures. Thus, time-dependent Robin boundary conditions

$$-\lambda^{(k)} \left(\partial_{\boldsymbol{n}} u^{(k)} \right) \Big|_{\boldsymbol{x}_i} = \alpha_A \left(u^{(k)} - T_A(t) \right) \tag{6.6}$$

are considered at the interface between the topmost layer of ground and the air above, where $T_A(t)$ is the ambient temperature on the earth's surface and α_A the heat transfer coefficient. The coefficient α_A can either be assumed to be constant or dependent on current weather conditions. If the wind speed v is known, the coefficient α_A can be specified, cf. [30], via

$$\alpha_A(v) = \begin{cases} 1.8 + 4.1v, & v \leq 5\text{m/s} \\ 7.3v^{0.73}, & v > 5\text{m/s} \end{cases} \tag{6.7}$$

The lateral domain boundaries on $\partial\Omega$ are generally unknown and may be influenced by various factors such as for instance border-near basements. Ideally, we assume that there exist no anthropogenic influences. Therefore, within the model, time-dependent Dirichlet boundary conditions are specified in the form of undisturbed ground temperatures $T_g(t,x)$, which also depend on space (i.e. on depth in the ground, in meters) in the following form:

$$T_g(t,x) = \bar{\theta} - \delta\theta \exp\left(-\frac{x}{\delta_g}\right) \cos\left[2\pi\frac{(t-\bar{t})}{t_h} - \frac{x}{\delta_g}\right] + G_t x \tag{6.8}$$

with average ambient temperature $\bar{\theta}$, amplitude of monthly fluctuations in ambient temperature $\delta\theta$, measure of lagging ambient temperature in depth δ_g, number of hours in year t_h, time lag between the time of the lowest annual temperature $\bar{t}$ and geothermal gradient G_t, where δ_g is given by

$$\delta_g = \sqrt{\frac{3600 t_h \lambda}{\pi\rho c}} \tag{6.9}$$

In particular, the lateral boundary condition (6.8) depends on the location of the installed GES due to the location-dependent parameters $\bar{\theta}, \delta\theta, \delta_g$ and G_t.

Finally, the lower domain boundaries are given by time-dependent Dirichlet boundary conditions in the form of groundwater temperatures

$$T_{gw}(t) = b_1 \cos\left(\frac{2\pi t}{t_h 3600}\right) + b_2 \tag{6.10}$$

with a measure of fluctuation intensity b_1 and the average groundwater temperature b_2. The latter condition models a seasonal shift, as the thermal energy spreads down into the depths with a time delay, cf. [202].

Remark 6.1. *The modelling of other boundary conditions can be done without restrictions. For instance, if the GES is located below a base plate or lateral to a basement, the boundary conditions only need to be adapted to fit the model.*

6.2.4 Generating the Initial Heat Distribution

The initial condition of the GES model problem, which corresponds to the initial temperatures at time $t = 0$, is generally not known. In practice, initial temperatures may only be determined by mounting temperature sensors $\widetilde{s}$ at some grid points. In the presence of sensor data, the initial temperature is known in some places $\boldsymbol{x}_j \in \Omega_K \subset \Omega$, but most of the initial heat distribution $u(\boldsymbol{x}, 0)$ remains unknown. One possibility is to estimate the unknown temperatures by interpolation using the given data. In doing so, the interpolation task can be realised by PDE-based *image inpainting*, also known as Laplace interpolation, see e.g. [97,250]. More precisely, image inpainting is a process in order to reconstruct or fill-in missing parts in the inpainting domain $\Omega\backslash\Omega_K$ in a way that is undetectable to the casual observer.

To this end, in turn, the heat equation can be used. In the simplest case of applying linear diffusion, the *initial heat distribution* is obtained as the steady state solution, i.e.

$$u(\boldsymbol{x}, 0) := \lim_{t\to\infty} \widetilde{u}(\boldsymbol{x}, t) \tag{6.11}$$

of the heat evolution that is described by

$$\begin{aligned} \partial_t \widetilde{u}(\boldsymbol{x},t) &= \Delta \widetilde{u}(\boldsymbol{x},t), && \forall \boldsymbol{x} \in \Omega \backslash \Omega_K \\ \widetilde{u}(\boldsymbol{x},0) &= 0, && \forall \boldsymbol{x} \in \Omega \backslash \Omega_K \\ \widetilde{u}(\boldsymbol{x},t) &= \widetilde{s}(\boldsymbol{x}), && \forall \boldsymbol{x} \in \Omega_K \end{aligned} \tag{6.12}$$

with suitable boundary conditions on $\partial\Omega$. In other words, the temperature sensors on Ω_K are interpreted as thermostats that are kept at a fixed temperature at any time, and are thus modelled as Dirichlet boundary conditions.

6.3 Discretisation of the Continuous-Scale Model

In this section we provide the basic discretisation of the underlying continuous GES model, characterised via linear heat equation (6.2), different interior (6.3)-(6.5) and exterior (6.6)-(6.8), (6.10) boundary conditions and initial heat distribution (6.11). In doing so, we describe the discretisation aspects in space and time in detail in the following subsections.

The underlying computational domain (as well as the heat tank), cf. Figure 6.1, is given by a cuboid type form, so that we apply standard finite difference schemes for the discretisation of the continuous model, which will be sufficient for the application in this chapter.

6.3.1 Discretisation in Space

For reasons of simplicity we consider in the following the two-dimensional rectangular domain $(x,y) = [x_1, x_n] \times [y_1, y_m] \in \Omega$ with an equidistant mesh size $h = \Delta x = \Delta y > 0$ in x- and y-direction, where $u_{i,j}(t)$ denotes an approximation of the unknown function u at grid point (x_i, y_j) and time t. For convenience only, we use the abbreviation $u_{i,j} := u_{i,j}(t)$.

The approximation of the spatial partial derivatives u_{xx} and u_{yy} in (6.2) using standard central differences leads to

$$\frac{\mathrm{d}u_{i,j}}{\mathrm{d}t} = a\frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}}{h^2} + \frac{f_{i,j}}{\rho c} \tag{6.13}$$

where $f_{i,j}$ is the discretised heat source/sink. It should be noted that (6.13) is only valid for inner points of the computational domain that are not located at internal interfaces. The spatial discretisation at the interfaces can be explained as follows.

In particular, we assume that no grid point is located exactly at the interface $\boldsymbol{x}_i$. Exemplarily, we define the interface as $\boldsymbol{x}_i := (x_{i+\frac{1}{2}}, y_j)$, which is centred located between the grid points (x_i, y_j) and (x_{i+1}, y_j). As stated before, the interaction between different types of soils is modelled by the equations (6.3)-(6.4). Discretising (6.3) at the interface $\boldsymbol{x}_i$ between two layers, denoted here as "k" and "l", using the forward and backward difference results in

$$\lambda^{(k)} \frac{u^{(k)}_{i+1,j} - u^{(k)}_{i+\frac{1}{2},j}}{\frac{h}{2}} = \lambda^{(l)} \frac{u^{(l)}_{i+\frac{1}{2},j} - u^{(l)}_{i,j}}{\frac{h}{2}} \tag{6.14}$$

Due to (6.4) we have $u^{(k)}_{i+\frac{1}{2},j} = u^{(l)}_{i+\frac{1}{2},j} =: u_I$ for the fictitious value u_I at the interface, which

can then be calculated via

$$u_I = \frac{\lambda^{(k)} u_{i+1,j}^{(k)} + \lambda^{(l)} u_{i,j}^{(l)}}{\lambda^{(k)} + \lambda^{(l)}} \tag{6.15}$$

The latter scheme is visualised on the left in Figure 6.2. At this point it should be mentioned that the discretisation can also be done without using a fictitious value, if one assumes that a grid point lies on the interface, see [129].

In contrast, when modelling the relation between soil and insulation, condition (6.4) must be replaced by (6.5). The discretisation of (6.5) at $\boldsymbol{x}_i$ using the forward and backward differences gives

$$\lambda^{(k)} \frac{u_{i+1,j}^{(k)} - u_{i+\frac{1}{2},j}^{(k)}}{\frac{h}{2}} = \alpha_c \left(u_{i+\frac{1}{2},j}^{(l)} - u_{i+\frac{1}{2},j}^{(k)} \right) \tag{6.16}$$

which can be rewritten as

$$\left(\frac{2\lambda^{(k)}}{h} - \alpha_c \right) u_{i+\frac{1}{2},j}^{(k)} + \alpha_c u_{i+\frac{1}{2},j}^{(l)} = \frac{2\lambda^{(k)}}{h} u_{i+1,j}^{(k)} \tag{6.17}$$

Moreover, the equation (6.14) can be transformed into

$$u_{i+\frac{1}{2},j}^{(k)} + \frac{\lambda^{(l)}}{\lambda^{(k)}} u_{i+\frac{1}{2},j}^{(l)} = u_{i+1,j}^{(k)} + \frac{\lambda^{(l)}}{\lambda^{(k)}} u_{i,j}^{(l)} \tag{6.18}$$

The equations (6.17)-(6.18) form a system of linear equations for the two unknowns $u_{i+\frac{1}{2},j}^{(k)}$

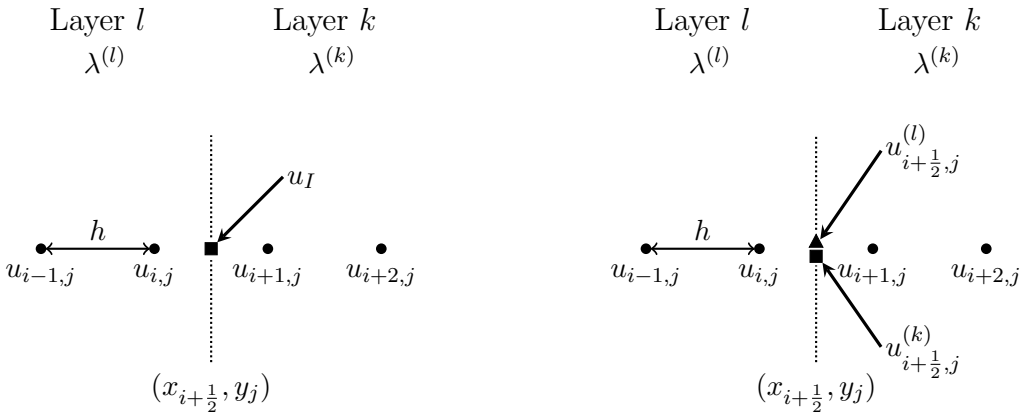


Figure 6.2: Schematic sketch at the interface $\boldsymbol{x}_i = (x_{i+\frac{1}{2}}, y_j)$ between the layers k and l using the proposed finite difference schemes at grid points (dots) and equidistant mesh size h. **Left:** The discretisation of the interaction between different types of soils leads to fulfilling the matching conditions (6.3)-(6.4) around the fictitious interface point, and defining the value u_I using the formula (6.15). **Right:** The discretisation of the interaction between soil and insulation leads to fulfilling the matching conditions (6.3) and (6.5) around the fictitious interface points, and defining left and right values $u_{i+\frac{1}{2},j}^{(k)}$ and $u_{i+\frac{1}{2},j}^{(l)}$ using the system of linear equations (6.19).

and $u^{(l)}_{i+\frac{1}{2},j}$ by means of

$$\begin{pmatrix} 1 & \frac{\lambda^{(l)}}{\lambda^{(k)}} \\ \frac{2\lambda^{(k)}}{h} - \alpha_c & \alpha_c \end{pmatrix} \begin{pmatrix} u^{(k)}_{i+\frac{1}{2},j} \\ u^{(l)}_{i+\frac{1}{2},j} \end{pmatrix} = \begin{pmatrix} u^{(k)}_{i+1,j} + \frac{\lambda^{(l)}}{\lambda^{(k)}} u^{(l)}_{i,j} \\ \frac{2\lambda^{(k)}}{h} u^{(k)}_{i+1,j} \end{pmatrix} \tag{6.19}$$

The following proposition verifies that the latter system has a unique solution for $\alpha_c \neq \frac{2\lambda^{(k)}\lambda^{(l)}}{h(\lambda^{(k)}+\lambda^{(l)})}$ and can be solved with Cramer's rule [65]:

Proposition 6.1. *The linear system* (6.19) *has a unique solution for*

$$\alpha_c \neq \frac{2\lambda^{(k)}\lambda^{(l)}}{h\left(\lambda^{(k)} + \lambda^{(l)}\right)} \tag{6.20}$$

Proof. The determinant of (6.19) is given by

$$\det \begin{vmatrix} 1 & \frac{\lambda^{(l)}}{\lambda^{(k)}} \\ \frac{2\lambda^{(k)}}{h} - \alpha_c & \alpha_c \end{vmatrix} = \alpha_c \left(1 + \frac{\lambda^{(l)}}{\lambda^{(k)}}\right) - \frac{2\lambda^{(l)}}{h} =: D \tag{6.21}$$

Due to the requirements $h > 0$ and $\lambda^{(k)} > 0$, the determinant D exists and is well-defined. For $\alpha_c = 0$ is $D \neq 0$, since $\lambda^{(l)} > 0$. In contrast, for $\alpha_c \neq 0$, the equation (6.21) implies that the determinant D is zero if $\alpha_c = \frac{2\lambda^{(k)}\lambda^{(l)}}{h(\lambda^{(k)}+\lambda^{(l)})}$ holds. □

The used scheme at the interface including a jump condition is shown on the right in Figure 6.2. The case for $\boldsymbol{x}_j := (x_i, y_{j+\frac{1}{2}})$ is handled analogously. We mention that another possible discretisation is presented in [129].

Finally, the exterior boundary conditions have to be discretised. In case of the upper domain boundaries at the topmost layer of ground, again interfaces $\boldsymbol{x}_i = (x_i, y_m)$ and fictitious values $u^{(k)}_{i,m+1}$ for $i = 1, \ldots, n$ are incorporated. Using the standard first order spatial discretisation

$$\left(\partial_y u^{(k)}\right)\Big|_{(x_i,y_m)} = \frac{u^{(k)}_{i,m+1} - u^{(k)}_{i,m}}{h} + \mathcal{O}(h) \tag{6.22}$$

linked to the discretised condition (6.6) as

$$-\lambda^{(k)} \left(\partial_y u^{(k)}\right)\Big|_{(x_i,y_m)} = \alpha_A \left(u^{(k)}_{i,m} - T_A(t)\right) \tag{6.23}$$

leads to

$$u^{(k)}_{i,m+1} = u^{(k)}_{i,m} - \frac{h\alpha_A}{\lambda^{(k)}} \left(u^{(k)}_{i,m} - T_A(t)\right) \tag{6.24}$$

The remaining conditions concerning the lateral and lower domain boundaries are fixed via

$$u_{0,j}^{(k)} = u_{n+1,j}^{(k)} = T_g(t, d_j) = T_{g_j}, \quad j = 1, \ldots, m \tag{6.25}$$

$$u_{i,0}^{(k)} = T_{gw}(t), \quad i = 1, \ldots, n \tag{6.26}$$

where d_j is the depth of the j-th grid layer. Let us mention that the three-dimensional case can be handled analogously.

Remark 6.2. *When using first order discretisations, the accuracy at the boundary is formally by one order worse than the error of the boundary-free discrete equation. However, it is a matter of definition of boundary location to understand the same finite difference expression as a central discretisation of the derivative inbetween the considered points, which is again of second order. This means we do not have to expect any error deterioration. In addition, an important aspect of the proposed first order discretisation is the symmetry preservation of the underlying Laplacian matrix L. The symmetry is of great importance for the application of numerical solvers.*

6.3.2 Arising System of Ordinary Differential Equations

Lastly, we summarise the components of the proposed two-dimensional discretisation (6.13), (6.15), (6.19) and (6.24)-(6.26) which end up in a semi-discrete system. In particular, a function defined on all grid points can now be represented as an N-dimensional vector

$$\boldsymbol{u}(t) = \big(u_1(t), \ldots, u_N(t)\big)^\top \tag{6.27}$$

where N is the total number of all grid points with linear grid point numbering from top left to bottom right.

The spatial discretisation of the GES by applying finite differences on a regular grid with constant grid size h results into an ODE system, one ODE for each grid point, which can be represented as follows:

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + K_1\boldsymbol{u}_1(t) + K_2\boldsymbol{u}_2(t) + K_A T_A(t) + K_{gw} T_{gw}(t) + K_g \boldsymbol{T}_g(t) + K_f \boldsymbol{f}(t) \tag{6.28}$$

with temperature vector $\boldsymbol{u} \in \mathbb{R}^N$, temperature vectors $\boldsymbol{u}_1 \in \mathbb{R}^n$ (continuity condition) and $\boldsymbol{u}_2 \in \mathbb{R}^{\widetilde{n}}$ (discontinuity condition) for fictitious points at the two different types of interfaces, ambient temperature $T_A \in \mathbb{R}$, groundwater temperature $T_{gw} \in \mathbb{R}$, undisturbed ground temperature vector

$$\boldsymbol{T}_g(t) = \big(T_{g_1}(t), \ldots, T_{g_m}(t)\big)^\top \in \mathbb{R}^m \tag{6.29}$$

and source/sink vector $\boldsymbol{f} \in \mathbb{R}^{\widetilde{m}}$. At this point it should be mentioned that the values for $\widetilde{n}$ and $\widetilde{m}$ depend on the user-defined size setting of the insulating walls and geometry setting of the considered source/sink, respectively.

Remark 6.3. *The conditions at the lateral boundaries on Ω considered here are identical, therefore it is sufficient that the input $\boldsymbol{T}_g$ specified in* (6.29) *is of size m.*

The matrices $L \in \mathbb{R}^{N\times N}$, $K_1 \in \mathbb{R}^{N\times n}$, $K_2 \in \mathbb{R}^{N\times\widetilde{n}}$ and $K_f \in \mathbb{R}^{N\times\widetilde{m}}$ collect the terms of the basic discretisation and are not shown in detail. Nevertheless, we give an exemplary

illustration of the technical construction for the Laplacian matrix L based on the continuity condition in Figure 6.3. The remaining components, which correspond to the ambient temperature, the groundwater temperature and the undisturbed ground temperature, have the following simple structure:

$$K_A = \frac{\alpha_A}{\rho c h} \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ \hline 1 \\ 0 \\ \vdots \\ 0 \\ \hline \vdots \end{pmatrix}, \quad K_{gw} = \frac{a}{h^2} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ \hline 0 \\ \vdots \\ 0 \\ 1 \\ \hline \vdots \end{pmatrix}, \quad K_g = \frac{a}{h^2} \begin{pmatrix} I_{m,m} \\ \hline 0_{N-2m,m} \\ \hline I_{m,m} \end{pmatrix} \tag{6.30}$$

with $K_A, K_{gw} \in \mathbb{R}^N$, $K_g \in \mathbb{R}^{N \times m}$, the identity matrix $I_{m,m} \in \mathbb{R}^{m \times m}$, the null matrix $0_{N-2m,m} \in \mathbb{R}^{(N-2m) \times m}$, and where a, ρ, c depend on material parameters. The mark "—" within (6.30) indicates that the discretisation points of the underlying rectangular computational domain are considered row by row. Finally, the discrete initial condition is given by (6.11) with $\boldsymbol{u}^0 := \boldsymbol{u}(\boldsymbol{x}, 0)$.

Remark 6.4. *Apart from the exemplary illustration for assembling the Laplacian matrix L, the Figure 6.3 additionally indicates the symmetry preservation of the matrix structure.*

6.3.3 Time Integration

The application of the developed spatial discretisations leads to the ODE system (6.28) which can be represented as

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + K\boldsymbol{w}(t), \quad t \in (0, t_F], \, \boldsymbol{u}(0) = \boldsymbol{u}^0, \, \boldsymbol{w}(0) = \boldsymbol{w}^0 \tag{6.31}$$

with input matrix and inputs

$$K = \Big[K_1, K_2, K_A, K_{gw}, K_g, K_f\Big], \quad \boldsymbol{w}(t) = \big(\boldsymbol{u}_1(t), \boldsymbol{u}_2(t), T_A(t), T_{gw}(t), \boldsymbol{T}_g(t), \boldsymbol{f}(t)\big)^\top \tag{6.32}$$

In doing so, we specify the input matrix $K \in \mathbb{R}^{N \times p}$ with $p = n + \widetilde{n} + 2 + m + \widetilde{m}$ and make use of stacked vectors for defining $\boldsymbol{w}(t) \in \mathbb{R}^p$. Let us note once more that the time-dependent large-scale input $\boldsymbol{w}(t)$ controls the model by boundary conditions and heat sources/sinks. Furthermore, the Laplacian matrix L is symmetric and negative definite which is large, sparse and structured. The definiteness follows directly from its strictly diagonal dominance and negative diagonal entries according to the Gershgorin's circle theorem [100].

Remark 6.5. *Although the matching conditions at the interfaces corresponding to $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are not user-defined controls, one has to handle these as indirect inputs.*

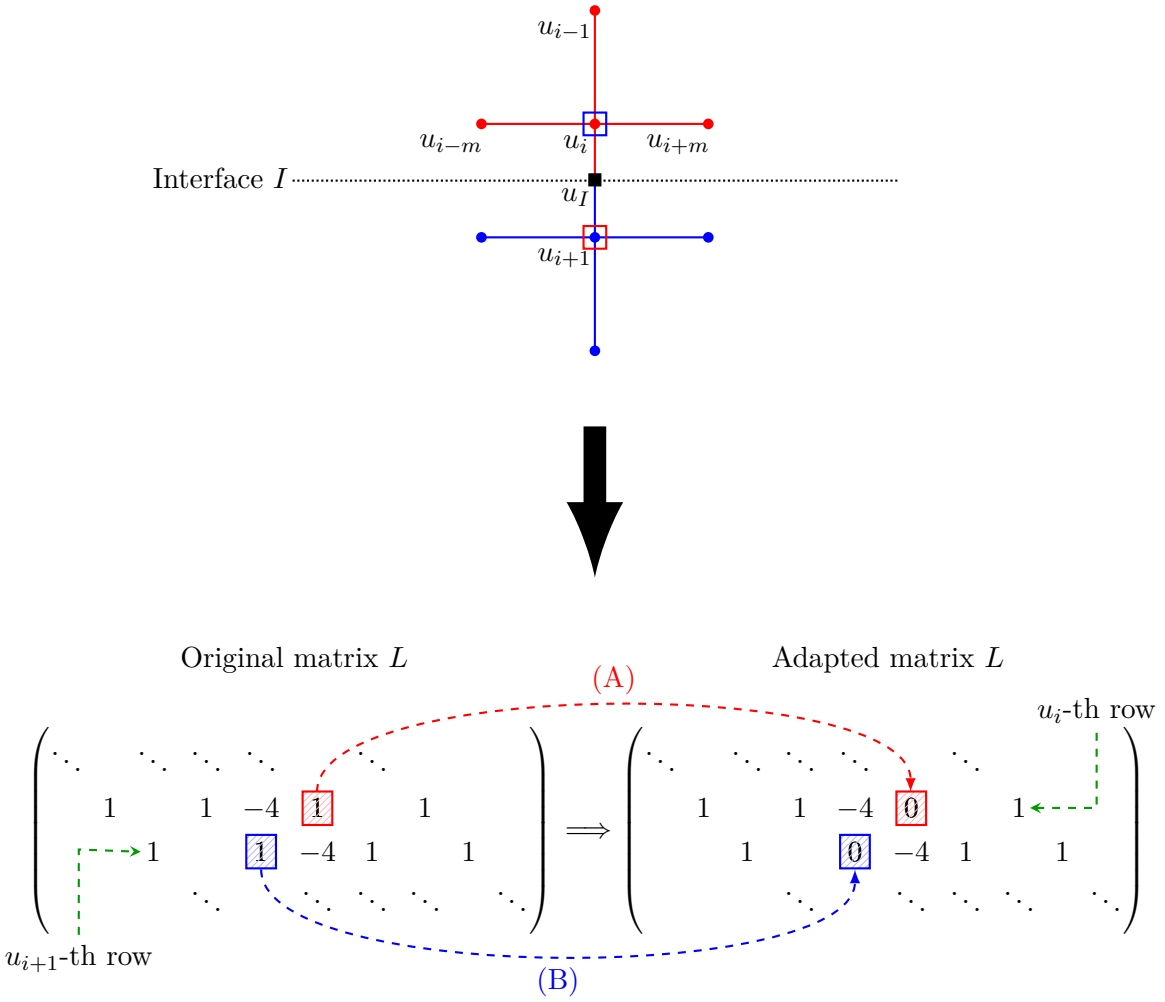


Figure 6.3: Exemplary illustration of the technical construction for the system matrix L including the mentioned matching conditions in 2D. **Top:** The sketch can be interpreted as a (red, blue) five-point stencil of the two-dimensional discrete Laplace operator, which is used to represent finite difference approximations of derivatives at grid points. The stencil considered here is made up of the point itself together with its four "neighbours". This means, for $h = 1$ the original red stencil yields $\Delta u_i \approx u_{i+1} + u_{i-1} + u_{i+m} + u_{i-m} - 4u_i$, where m denotes the number of rows. **Bottom:** The corresponding rows of the original and adapted system matrix L for the grid points u_i and u_{i+1} at the interface I, as already illustrated in the upper sketch. **(A)** To compute Δu_i (red stencil), u_{i+1} (red square) is substituted via u_I (black square). More precisely, we replace 1 inside the original matrix L with 0 into the adapted matrix (red dashed line). **(B)** To compute Δu_{i+1} (blue stencil), we substitute u_i (blue square) via u_I (black square). Again, 1 is replaced with 0 (blue dashed line). In this case, the fictitious value u_I is defined by the formula (6.15). This proceeding is performed analogously for all grid points at the interfaces. In the same manner, the second kind of matching conditions is also being handled. This implementation implies that the adapted system matrix L remains symmetric. Let us stress that the changed coefficients "1" do not disappear. Since these coefficients then define the matrices K_1 and K_2 for the corresponding temperature vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$, respectively.

Discrete time integration of ODEs can then be achieved using the standard time stepping methods, introduced in Section 2.1. Besides the EE method we consider here the IE and CN scheme for the numerical solution of the underlying model problem. For time discretisation, we define uniform time intervals $I_k = [t_k, t_{k+1}]$ and set $t_0 = 0$ for subdividing the complete integration time $[0, t_F]$.

Explicit Euler Method Applying the fundamental theorem of calculus and the left-hand rectangle method for the left-hand side and right-hand side, respectively, of (6.31) including the uniform time step size $\tau = t_{k+1} - t_k$ and using the notations $\boldsymbol{u}(t_k) = \boldsymbol{u}^k$, $\boldsymbol{w}(t_k) = \boldsymbol{w}^k$ results in the EE scheme

$$\boldsymbol{u}^{k+1} = (I + \tau L)\, \boldsymbol{u}^k + \tau K \boldsymbol{w}^k \tag{6.33}$$

with $k \in \{0, \ldots, F-1\}$, the identity matrix $I \in \mathbb{R}^{N \times N}$ and given data $\boldsymbol{u}^0 = \boldsymbol{u}(0)$, $\boldsymbol{w}^0 = \boldsymbol{w}(0)$. Due to the fact that the values $\boldsymbol{u}^k$ and $\boldsymbol{w}^k$ are known, the computation of the corresponding values $\boldsymbol{u}^{k+1}$ at time level $k+1$ can be done by simple matrix-vector multiplication.

From Section 2.1.2 it is known that the stability properties of the EE scheme remain unrestricted if $\boldsymbol{w}$ is linearly dependent on the temperature, which is the case here. The following time step size restriction on τ holds:

Proposition 6.2. *The numerical stability of the EE method* (6.33) *is guaranteed for*

(i) *in **2D:***
$$\tau \leq \frac{2h^2}{\max(a) \max\left(8, 6 + \frac{\alpha_A h}{\widetilde{\lambda}}\right)} =: \tau_{\max,2D}$$

(ii) *in **3D:***
$$\tau \leq \frac{2h^2}{\max(a) \max\left(12, 10 + \frac{\alpha_A h}{\widetilde{\lambda}}\right)} =: \tau_{\max,3D}$$

where a is the thermal diffusivity of the materials (varies on location), α_A the heat transfer coefficient, h the uniform grid size and $\widetilde{\lambda}$ the thermal conductivity of the upper ground.

Proof. Let us consider the two-dimensional model problem. First, we recall the *adapted* Laplacian matrix L, extended from Figure 6.3. Using the mentioned first order spatial discretisation (6.24) one obtains

$$L = \frac{1}{h^2} \left(\begin{array}{cccccccccccc}
za & a & & & & a & & & & & & \\
a & -4a & a & & & & a & & & & & \\
 & a & -4a & a & & & & a & & & & \\
 & & \ddots & \ddots & \ddots & & & & \ddots & & & \\
 & & & a & -4a & 0 & & & & a & & \\
\hline
a & & & & 0 & za & a & & & & a & \\
 & a & & & & a & -4a & a & & & & a \\
 & & \ddots & & & & \ddots & \ddots & \ddots & & & \ddots
\end{array}\right) \quad \text{with} \quad z = -3 - \frac{\alpha_A h}{\widetilde{\lambda}} \tag{6.34}$$

It should be noted that the thermal diffusivity a is not constant, in particular, the value depends on material properties and varies in relation to the grid points. According to the Gershgorin's circle theorem [100] we have

$$\begin{aligned}\varrho(L) \subseteq \Bigg\{ \lambda \in \mathbb{C} : \left|\lambda + \left(3 + \frac{\alpha_A h}{\widetilde{\lambda}}\right)\frac{a}{h^2}\right| \leq \frac{2a}{h^2} \quad \vee \quad \left|\lambda + \left(3 + \frac{\alpha_A h}{\widetilde{\mu}}\right)\frac{a}{h^2}\right| \leq \frac{3a}{h^2} \\ \vee \quad \left|\lambda + \frac{4a}{h^2}\right| \leq \frac{2a}{h^2} \quad \vee \quad \left|\lambda + \frac{4a}{h^2}\right| \leq \frac{3a}{h^2} \quad \vee \quad \left|\lambda + \frac{4a}{h^2}\right| \leq \frac{4a}{h^2} \Bigg\} \\ = \left[\min\left(-\frac{8a}{h^2}, \left(-6 - \frac{\alpha_A h}{\widetilde{\lambda}}\right)\frac{a}{h^2}\right), 0 \right] \end{aligned} \tag{6.35}$$

where ϱ denotes the spectrum of L. Due to the symmetry of L the eigenvalues are real and their Euclidean norm corresponds to the largest eigenvalue in magnitude. To ensure stability the following condition for τ must hold:

$$\left|1 + \tfrac{\tau}{h^2} \min\left(-8a, -6a - \tfrac{\alpha_A h}{\widetilde{\lambda}} a\right)\right| \leq 1 \quad \Longrightarrow \quad \tau \leq \underbrace{\frac{2h^2}{\max(a)\max\left(8, 6 + \frac{\alpha_A h}{\widetilde{\lambda}}\right)}}_{=:\tau_{\max,2\mathrm{D}}} \tag{6.36}$$

The three-dimensional case is handled analogously. □

Implicit Methods Applying the fundamental theorem of calculus, including the use of the right-hand rectangle method or the trapezoidal rule for the integral approximation of the right-hand side of (6.31), we obtain the implicit methods

$$\text{(IE)}: \qquad (I - \tau L)\,\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \tau K \boldsymbol{w}^{k+1} \tag{6.37}$$

$$\text{(CN)}: \qquad \left(I - \tfrac{\tau}{2} L\right)\boldsymbol{u}^{k+1} = \left(I + \tfrac{\tau}{2} L\right)\boldsymbol{u}^k + \tfrac{\tau}{2} K\left(\boldsymbol{w}^{k+1} + \boldsymbol{w}^k\right) \tag{6.38}$$

where $\boldsymbol{w}^k, \boldsymbol{w}^{k+1}$ is given from data. As is known, both schemes are by theory unconditionally stable and require the solution of a linear system for the values at the time level $k+1$. The CN method is slightly more intensive than IE, but provides second order convergence in time.

6.4 Numerical Methods

In this section we provide some details in order to properly apply the FEDRK scheme and the KSMOR technique for the fundamental problem of heat evolution with internal and external boundary conditions as well as source terms. First, some relevant properties of the FEDRK method are highlighted that overcome the disadvantages of the FED method. Second, we introduce the KSMOR methods in connection with the GES application for the first time. Due to a large number of inputs (6.32) the use of KSMOR leads to new challenges. For this reason, we follow a recent approach [232] to enhance the efficiency of the original technique. Finally, the viability of the modified KSMOR method is demonstrated using a test example.

6.4.1 Fast Explicit Methods

As presented in Chapter 3 fast explicit methods are highly efficient schemes that are well-suited for parallel GPU processing and are simple to implement. In this class of methods, FEDRK and RKL are of beneficial use because no additional damping is required. Both techniques, which basically differ from a theoretical point of view, can be deemed as equally competitive, so we focus on the FEDRK method which is solely applied in image processing.

The FED and FEDRK method are mainly used for image processing tasks, especially in connection with homogeneous nonlinear and anisotropic diffusion filtering [6,108,116,118,294] or image registration including force vectors [171,172]. However, the scheme can be applied to many parabolic problems, including sources/sinks, also in an engineering context as demonstrated in this chapter.

Fast Explicit Diffusion The FED method described in Section 3.4 can easily be used for problems including sources as well as internal and external boundary conditions. Applying FED to (6.31) yields a cycle of n explicit linear diffusion steps given by

$$\begin{aligned} \boldsymbol{u}^{k+1,0} &= \boldsymbol{u}^k, \\ \boldsymbol{u}^{k+1,i+1} &= (I + \widetilde{\tau}_i L)\, \boldsymbol{u}^{k+1,i} + \widetilde{\tau}_i K \boldsymbol{w}^k, \quad i = 0, 1, \ldots, n-1 \\ \boldsymbol{u}^{k+1} &= \boldsymbol{u}^{k+1,n} \end{aligned} \tag{6.39}$$

with $\widetilde{\tau}_i := c\tau_i, c := \frac{4}{h^2 \lambda_{max}}$, and where the time step sizes are defined by

$$\widetilde{\tau}_i := \tau_{\max} \frac{1}{2\cos^2\left(\pi \frac{2i+1}{4n+2}\right)}, \quad i = 0, 1, \ldots, n-1 \tag{6.40}$$

The corresponding upper stability bound $\tau_{\max}$ is specified in the Proposition 6.2 above. It should also be noted that the input $\boldsymbol{w}^k$ is constant within the full cycle (6.39). Therefore, the FED scheme inherits the same properties as the original method without existing an input as shown in the following:

Proposition 6.3. *Let L be the introduced adapted discrete Laplacian and let the time step limit $\tau_{\max}$ satisfies the stability condition given in Proposition 6.2, then the FED scheme* (6.39) *is stable with respect to the Euclidean norm.*

Proof. One FED cycle can be represented recursively by

$$\boldsymbol{u}^{k+1,1} = (I + \widetilde{\tau}_0 L)\, \boldsymbol{u}^{k+1,0} + \widetilde{\tau}_0 K \boldsymbol{w}^k \tag{6.41}$$

$$\begin{aligned} \boldsymbol{u}^{k+1,2} &= (I + \widetilde{\tau}_1 L)\, \boldsymbol{u}^{k+1,1} + \widetilde{\tau}_1 K \boldsymbol{w}^k \\ &= (I + \widetilde{\tau}_1 L) \left[(I + \widetilde{\tau}_0 L)\, \boldsymbol{u}^{k+1,0} + \widetilde{\tau}_0 K \boldsymbol{w}^k \right] + \widetilde{\tau}_1 K \boldsymbol{w}^k \\ &= (I + \widetilde{\tau}_1 L)\, (I + \widetilde{\tau}_0 L)\, \boldsymbol{u}^{k+1,0} + (I + \widetilde{\tau}_1 L)\, \widetilde{\tau}_0 K \boldsymbol{w}^k + \widetilde{\tau}_1 K \boldsymbol{w}^k \end{aligned} \tag{6.42}$$

and analogously for $i = 2, \ldots, n-1$ this finally leads to

$$\boldsymbol{u}^{k+1,n} = \left(\prod_{i=0}^{n-1} (I + \widetilde{\tau}_i L) \right) \boldsymbol{u}^{k+1,0} + \left(\prod_{i=1}^{n-1} (I + \widetilde{\tau}_i L) \right) \boldsymbol{g}^0 + \left(\prod_{i=2}^{n-1} (I + \widetilde{\tau}_i L) \right) \boldsymbol{g}^1 + \cdots \tag{6.43}$$

with $\boldsymbol{g}^i = \widetilde{\tau}_i K \boldsymbol{w}^k$. Using a perturbed vector $\boldsymbol{u}_*^{k+1,0}$ of the initial values $\boldsymbol{u}^{k+1,0}$, the perturbed solution is given as

$$\boldsymbol{u}_*^{k+1,n} = \left(\prod_{i=0}^{n-1}(I+\widetilde{\tau}_i L)\right)\boldsymbol{u}_*^{k+1,0} + \left(\prod_{i=1}^{n-1}(I+\widetilde{\tau}_i L)\right)\boldsymbol{g}^0 + \left(\prod_{i=2}^{n-1}(I+\widetilde{\tau}_i L)\right)\boldsymbol{g}^1 + \cdots \quad (6.44)$$

The combination of equations (6.43) and (6.44) with the perturbation error vector defined as $\boldsymbol{e} = \boldsymbol{u}_* - \boldsymbol{u}$ yields

$$\boldsymbol{e}^n = \boldsymbol{u}_*^{k+1,n} - \boldsymbol{u}^{k+1,n} = \left(\prod_{i=0}^{n-1}(I+\widetilde{\tau}_i L)\right)\left(\boldsymbol{u}_*^{k+1,0} - \boldsymbol{u}^{k+1,0}\right) = \left(\prod_{i=0}^{n-1}(I+\widetilde{\tau}_i L)\right)\boldsymbol{e}^0 \quad (6.45)$$

with a perturbation $\boldsymbol{e}^0 := (\boldsymbol{u}_*^{k+1,0} - \boldsymbol{u}^{k+1,0})$. Consequently, the FED scheme is stable in the Euclidean norm following (3.146). □

In theory, the only requirement for the FED scheme is that the underlying matrix L must be symmetric and negative semi-definite. Nevertheless, the FED scheme has two major drawbacks from a numerical point of view. Although the internal stability of the scheme is fulfilled when using the natural sequence τ_i and thus stable intermediate solutions are enabled, FED is highly sensitive with respect to numerical rounding errors. Thus, a rearrangement of varying time steps is required in order to avoid serious accumulation of rounding errors. As a consequence, such a rearrangement can yield highly unstable intermediate solutions and therefore the input $\boldsymbol{w}^k$ should be kept constant within one cycle (6.39) (updating after a full cycle). For this reason, the well-performing FEDRK scheme is better suited for solving GES.

Fast Explicit Diffusion Runge-Kutta In order to reduce numerical rounding errors and simultaneously to increase the approximation quality we apply the FEDRK method, cf. Section 3.4.7. Another advantage over FED is that FEDRK causes less computational effort due to the use of nonvarying time step sizes, since $I + \tau L$ must be computed only once. For time-independent inputs, i.e. $\boldsymbol{w}(t) \equiv \boldsymbol{w}$, both methods obviously give identical results. In contrast to FED, the FEDRK scheme only uses stable time steps and ensures internal stability so that strong numerical rounding errors can be avoided. As a result, the time step sizes do not require a rearrangement and the input $\boldsymbol{w}^k$ can be updated within a cycle. Based on these properties, the proposed FEDRK scheme is highly beneficial. To use FEDRK with inner updates of the input, we recall the recursive form (3.196). Finally, the cyclic FEDRK scheme for the m-th cycle with cycle length n is given by

$$\begin{aligned}
&\boldsymbol{u}^{m,k+1} = \alpha_k\Big[(I+\tau L)\,\boldsymbol{u}^{m,k} + \tau K \boldsymbol{w}^{m,k}\Big] + (1-\alpha_k)\,\boldsymbol{u}^{m,k-1}, \quad k = 0,\ldots,n-1 \\
&n = \left\lceil \sqrt{\frac{3t_F}{\tau_{\max} M} + \frac{1}{4}} - \frac{1}{2} \right\rceil, \quad \tau = \frac{3t_F}{Mn(n+1)}, \quad \alpha_k = \frac{4k+2}{2k+3}, \quad \boldsymbol{u}^{m,-1} := \boldsymbol{u}^{m,0} \\
&\boldsymbol{w}^{m,k} := \boldsymbol{w}^m(c_k), \quad c_{k+1} = \alpha_k\left(c_k + \frac{3t_F}{Mn(n+1)}\right) + (1-\alpha_k)c_{k-1}, \quad c_0 = c_{-1} = 0
\end{aligned} \quad (6.46)$$

where M is the number of outer FEDRK cycles, t_F the stopping time and $\boldsymbol{w}^{m,k}$ is the input at the increment parameter c_k of the m-th cycle. Considering the GES application, the FEDRK scheme (6.46) is inherently stable in the Euclidean norm when the EE scheme is stable. Let us note again that increasing the number of cycles, whereby n becomes smaller, improves the accuracy of this method.

We point out that the fast explicit methods such as FED and FEDRK have a natural weakness. The methods perform inefficiently for highly nonuniform meshes, especially when very small grid widths arise. This is based on the fact that the (minimum) spatial mesh width and the allowed time step size of explicit methods are coupled. In this chapter, however, it is sufficient to consider (relatively) uniform grids in the context of the GES application.

6.4.2 Implicit Methods

The IE and CN method are implicit schemes and result in a system of linear equations. More precisely, (6.37) and (6.38) can be expressed as a linear system

$$A\boldsymbol{x} = \boldsymbol{b} \tag{6.47}$$

using the notations $A = I - \tau L$, $\boldsymbol{b} = \boldsymbol{u}^k + \tau K \boldsymbol{w}^{k+1}$ for the IE method as well as $A = I - \frac{\tau}{2} L$, $\boldsymbol{b} = (I + \frac{\tau}{2} L)\boldsymbol{u}^k + \frac{\tau}{2} K(\boldsymbol{w}^{k+1} + \boldsymbol{w}^k)$ for the CN scheme. The underlying constant matrix $A \in \mathbb{R}^{N \times N}$ is symmetric, positive definite and also large, sparse and structured.

The linear system (6.47) can be solved with sparse direct and sparse iterative solvers, see Section 2.2. In doing so, it will be interesting which of the two solvers is more efficient for the GES application considered here. In case of iterative solvers such as CG and PCG, the optimal selection of the parameters, more precisely the relative residual ε and the drop tolerance γ, must be investigated experimentally. Another interesting aspect from a numerical point of view is the effect on the iterative solver's performance when the current heat distribution is used as the start initialisation, in simple terms by $\boldsymbol{x}_0 = \boldsymbol{u}^k$, within CG and PCG instead of the common choice $\boldsymbol{x}_0 = \boldsymbol{0}$. In principle, the latter initialisation appears logical if two successive approximations $\boldsymbol{u}^k$ and $\boldsymbol{u}^{k+1}$ do not differ strongly, which is to be expected for linear heat equations for smaller time step sizes τ.

Other techniques such as MG methods may also be used for solving (6.47) efficiently. As already mentioned, their implementation is a rather delicate and quite elaborate, and basically such methods are typically superior for discretised elliptic boundary value problems. Otherwise, due to the large variety of different MG methods it is beyond the scope of this work to investigate the use of those methods here.

6.4.3 Adapted KSMOR Technique

The implicit methods have to handle large sparse systems, in which the computational costs directly depend on the data size. A potential alternative to implicit solvers are MOR techniques as already demonstrated for the PSC application in Chapter 5. Since the underlying ODE system (6.31) is large, the use of the BT method is not preferable for reasons of efficiency. In addition, due to the inclusion of internal and external boundary conditions in connection with extensive computational costs for solving an eigenvalue problem, the MCR technique is also impractical and will not be considered further here.

As stated in Section 6.3.2 and 6.3.3, the original system is built on a large-scale input. Unfortunately, a large number of inputs is a major challenge when using the KSMOR technique. To overcome this problem, we propose an adapted KSMOR* method that is used in a similar way to the recent SVD-based method [232]. Within the scope of this work, this issue is addressed very detailed in this section, with the practicability is tested at hand an example. Using this test example, we will compare both KSMOR variants with the POD method (introduced in Section 4.6) for dealing with dynamical systems with a large number of inputs.

Krylov Subspace Model Order Reduction As explained in Section 4.5, moment matching techniques approximate the moments of the system's transfer function and assuming zero initial condition $\boldsymbol{u}(0) = \mathbf{0}$. Therefore, by use of the coordinate transformation $\widetilde{\boldsymbol{u}}(t) = \boldsymbol{u}(t) - \boldsymbol{u}^0$ the original MICO[2] system (6.31) can be translated into

$$\begin{cases} \dot{\widetilde{\boldsymbol{u}}}(t) = L\widetilde{\boldsymbol{u}}(t) + L\boldsymbol{u}^0 + K\boldsymbol{w}(t) \\ \widetilde{\boldsymbol{y}}(t) = \widetilde{\boldsymbol{u}}(t) + \boldsymbol{u}^0, \quad \widetilde{\boldsymbol{u}}(0) = \mathbf{0} \end{cases} \tag{6.48}$$

where the nonzero initial condition now appears on the right-hand side of the ODE system. The transformed system (6.48) equipped with zero initial conditions can be rewritten as

$$\begin{cases} \dot{\widetilde{\boldsymbol{u}}}(t) = L\widetilde{\boldsymbol{u}}(t) + \widetilde{K}\widetilde{\boldsymbol{w}}(t) \\ \widetilde{\boldsymbol{y}}(t) = \widetilde{\boldsymbol{u}}(t) + \boldsymbol{u}^0, \quad \widetilde{\boldsymbol{u}}(0) = \mathbf{0} \end{cases} \tag{6.49}$$

with $\widetilde{K} = [K, L\boldsymbol{u}^0]$ and $\widetilde{\boldsymbol{w}}(t) = [\boldsymbol{w}(t), 1]^\top$. Applying the Laplace transform to the system (6.49) and assuming that the inverse $(sI - L)^{-1}$ exists, we obtain

$$\begin{aligned} \widetilde{\boldsymbol{U}}(s) &= (sI - L)^{-1}\widetilde{K}\widetilde{\boldsymbol{W}}(s) \\ \widetilde{\boldsymbol{Y}}(s) &= (sI - L)^{-1}\widetilde{K}\widetilde{\boldsymbol{W}}(s) + \tfrac{\boldsymbol{u}^0}{s} = \boldsymbol{H}(s)\widetilde{\boldsymbol{W}}(s) + \tfrac{\boldsymbol{u}^0}{s} \end{aligned} \tag{6.50}$$

where the transfer function $\boldsymbol{H}(s) = (sI - L)^{-1}\widetilde{K}$ is expressed by the input $\widetilde{\boldsymbol{W}}$ and the output $\widetilde{\boldsymbol{Y}}$ in the frequency domain. The key idea of KSMOR is based on moment matching, whereby the k-th moment of the transfer function is given here via $m_k(\sigma) = (L - \sigma I)^{-(k+1)}\widetilde{K}$. In order to avoid numerical problems the projection matrices are chosen as biorthogonal matrices $W^\top V = I$. In doing so, the one-sided Arnoldi approach is applied by means of $W = V$, which amounts to construct an orthogonal basis using the input block Krylov subspace

$$V = \operatorname{span}\left((L - \sigma I)^{-1}\widetilde{K}, (L - \sigma I)^{-2}\widetilde{K}, \ldots, (L - \sigma I)^{-q}\widetilde{K}\right) \tag{6.51}$$

Due to the fact that $\widetilde{K} \in \mathbb{R}^{N\times(p+1)}$ is a matrix, the projection matrix $V \in \mathbb{R}^{N\times r}$ with dimension $r = (p+1)q$ is computed by the block Arnoldi algorithm [238], and the reduced system leads to

$$\begin{cases} \dot{\widetilde{\boldsymbol{u}}}_r(t) = L_r\widetilde{\boldsymbol{u}}_r(t) + \widetilde{K}_r\widetilde{\boldsymbol{w}}(t) \\ \widetilde{\boldsymbol{y}}_r(t) = V\widetilde{\boldsymbol{u}}_r(t) + \boldsymbol{u}^0, \quad \widetilde{\boldsymbol{u}}_r(0) = \mathbf{0} \end{cases} \tag{6.52}$$

[2] In this work, we are generally interested in the complete (temperature) distribution so that the output vector $\boldsymbol{y}(t)$ is of full dimension, which is a difference to many other works found in the literature.

where $L_r = V^\top LV \in \mathbb{R}^{r\times r}$, $\widetilde{K}_r = V^\top \widetilde{K} \in \mathbb{R}^{r\times(p+1)}$. Then the reduced system can be efficiently solved by using an implicit time integration scheme in combination with a direct solver and an LU factorisation. After computing the solution of the reduced model (6.52) the original solution can be recovered via $\widetilde{\boldsymbol{u}} \approx V\widetilde{\boldsymbol{u}}_r$ and $\boldsymbol{u}(t) = \widetilde{\boldsymbol{u}}(t) + \boldsymbol{u}^0$.

Remark 6.6. *As a result of the underlying stable dynamical system built on the negative definite matrix L, the one-sided KSMOR method, i.e. $W = V$, preserves the stability of the reduced system, cf. Lemma 4.1.*

A parameter still to be determined is the choice of the expansion point σ. The underlying heat evolution is characterised by a rather slow dynamic, so approximating the system at the frequency $\sigma = 0$ is a natural choice. In particular, the inverse $(L - \sigma I)^{-1}$ does not exist for $\sigma = \lambda_i$, where λ_i corresponds to an eigenvalue of L.

Block Arnoldi For the construction of the Krylov subspace

$$\operatorname{range}(V) = \mathcal{K}_q\left((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\widetilde{K}\right) \tag{6.53}$$

the included inverse $(L - \sigma I)^{-1}$ leads anew to the task of solving large sparse systems of linear equations. This requires the application of sparse direct or sparse iterative solvers as previously stated. In this chapter we focus on the MOR technique itself, therefore we apply the sparse direct solver as this gives the most accurate representation. Analogous to the SISO system, see Section 4.5, the Arnoldi algorithm with $W = V$ computes an orthonormal basis using the (modified) Gram-Schmidt method. For MIMO systems the input Krylov subspace is simply replaced by the input block Krylov subspace with multi-dimensional input matrix $\widetilde{K}$. The corresponding algorithm of the one-sided Krylov method for MIMO systems using the input block Krylov subspace is shown in Figure 6.4. The underlying construction of V has to deal with orthogonalisation processes. Typically, there can be a significant loss of orthogonality due to round-off errors as an orthogonalisation algorithm progresses. In order to address this orthogonality problem and to achieve a stable procedure that improves the moment matching accuracy, a reorthogonalisation of a new computed block with respect to all previous blocks is performed, see step 2e) in Algorithm 6.1.

Dealing with MIMO systems has the consequence that the dimension of the reduced order model is significantly larger compared to SISO systems, since the Krylov subspace (6.53) has to contain all the information from the individual Krylov subspaces corresponding to the columns generated by $\widetilde{K}$. Consequently, the computational costs of the KSMOR method increase both for the offline as well as for the online phase.

Remarks on the Reduced Solution Based on the interior GES matching conditions and their associated calculations of fictitious values at the interfaces, the reduced solution $\widetilde{\boldsymbol{u}}_r$ of (6.52) must be re-enlarged in each time step into the original dimension by means of

$$\widetilde{\boldsymbol{u}} \approx V\widetilde{\boldsymbol{u}}_r \tag{6.54}$$

Because the projection matrix V is not sparse the matrix-vector multiplication (6.54) increases the computational costs significantly in each iteration. For this reason, we strive to avoid large dimensional projection matrices for the GES application.

Algorithm 6.1 One-sided Krylov subspace method for MIMO systems using input matrix

Input: Matrix L; input matrix K; Krylov subspace order q; expansion point σ

Output: $V = [V_1, \ldots, V_q]$

1.) Set $i = 1$

a) $V_1 = (L - \sigma I)^{-1} K$

b) QR factorisation: $Q_1 R_1 = V_1$

c) $V = [Q_1]$

2.) Iterate for $i = 2, \ldots, q$

a) $V_i = (L - \sigma I)^{-1} Q_{i-1}$

b) $V_i = V_i - V \left(V^\top V_i \right)$

c) QR factorisation: $Q_i R_i = V_i$

d) $V := [V, Q_i]$

e) Reorthogonalise V

Figure 6.4: The one-sided Krylov subspace method for constructing the projection matrix V considering MIMO systems using the input block Krylov subspace. QR factorisations are used within the algorithm that reflect the modified Gram-Schmidt process. Assuming the input matrix contains p inputs, the reduced model is of order $r = pq$.

In order to overcome this issue more efficiently, it could be useful to use approaches such as the discrete empirical interpolation method [60] that deal with such problems in the nonlinear regime. However, this is not investigated in this chapter.

Treatment of a Large Number of Inputs A significant weakness of KSMOR arises when dealing with dynamical systems with large-scale inputs. To retain high computational efficiency the KSMOR techniques typically require a relatively small number of inputs. The efficiency limitations caused by the large-scale input can be explained as follows: first, a large number of inputs leads to higher computational costs when constructing the Krylov subspace (6.51) within the offline phase. Second, every input eventually needs to be expanded in terms of several moments, so that the order of the reduced system grows significantly with the number of inputs and thus the online costs increase.

In general it is assumed that the number of inputs is quite small and usually independent of the fineness of the grid. However, the GES application discussed in this chapter cannot guarantee this feature. In particular, the input size depends on the user-defined dimensions of the heat tank and the incorporated insulating walls, which have to fulfil the matching conditions described in Section 6.2.2. Because of these considerations, the KSMOR method should not be applied directly and a change in method for dealing with large-scale inputs is absolutely necessary.

At present, there exists no universal procedure to tackle the problem of a large input dimension, and how to deal with this issue also generally depends on the model problem. Some investigations [38, 41, 88, 90, 140, 164, 232] have been done in the past. The proposed methods [41,88,90,164] are based on the approximation of the input matrix using the dominant singular vectors of the transfer function at the steady state. Usually, these approaches are limited to systems with a high correlation between the various inputs and outputs, which is not the case for thermal systems as considered here. Another method is based on the superposition principle to linear systems introduced in [38]. The reduction is performed separately using each column of the input matrix so that the original system is approximated by the summation of the output responses of each of the single-input reduced systems. As a result, the original MIMO system is decoupled into separate single-input systems. The superposition approach leads to the fact that the same number of moments is matched and a reduced system of the same order is built. Consequently, in order to attain the same accuracy this technique reduces the computational complexity for systems with a large number of inputs compared to the standard KSMOR method. Nonetheless, the superposition principle for systems with large-scale inputs remains computationally intensive for practical purposes.

The recently proposed approach by [232] is similar to ours. The basic idea is to use an input matrix reduction based on a snapshot matrix linked with computing the dominant singular vectors. We also make use of snapshots here, but in order to reduce the computational costs we neglect the SVD and use the snapshot matrix directly. The idea of using the snapshot matrix in a direct manner is applied to a nonlinear-input problem in [140], where the authors proposed to use the weighted average of these snapshots to form a single input. However, this is not an appropriate approach for this application.

Modified Krylov Subspace Model Order Reduction (KSMOR*) Let us now propose a heuristic procedure to resolve the problem of a large number of inputs which incorporates some technical novelties. In doing so, we only adapt the input matrix explicitly for the subspace construction procedure. For the description let us return to the ODE-system (6.31), in particular to the part of the control term. For the sake of simplicity, we represent the p-dimensional input $\boldsymbol{w}$ and the corresponding input matrix $K \in \mathbb{R}^{N\times p}$ by

$$\boldsymbol{w}(t) = \big(w_1(t), \ldots, w_p(t)\big)^\top, \quad K = \begin{pmatrix} | & | & & | \\ K_1 & K_2 & \cdots & K_p \\ | & | & & | \end{pmatrix} \tag{6.55}$$

with $K_i \in \mathbb{R}^N$ for $i = 1, \ldots, p$. As is known, the costs of building the Krylov subspace $V = \mathcal{K}_q((L-\sigma I)^{-1}, (L-\sigma I)^{-1}K)$ for large values p are immense. For motivation, we assume that the input is time-independent, i.e. $\boldsymbol{w}(t) \equiv \boldsymbol{w}$, then the input matrix K from (6.55) could be assembled via

$$K\boldsymbol{w} =: \widehat{\boldsymbol{K}} \in \mathbb{R}^N \tag{6.56}$$

which would be accompanied by very low computational costs for generating

$$\widehat{V} = \mathcal{K}_q\Big((L-\sigma I)^{-1}, (L-\sigma I)^{-1}\widehat{\boldsymbol{K}}\Big) \tag{6.57}$$

Before proceeding with our approach, we state the relations (6.55)-(6.57) in an exemplary manner below.

Example 6.1. *Let us explain the formulation* (6.56) *in more detail using the following simple example: for instance, the underlying dynamical system has the form*

$$\begin{aligned} \dot{\boldsymbol{u}}_1(t) &= \boldsymbol{u}_1(t) + \boldsymbol{w}_1(t) \\ \dot{\boldsymbol{u}}_2(t) &= \boldsymbol{u}_2(t) + \boldsymbol{w}_2(t) \\ \dot{\boldsymbol{u}}_3(t) &= \boldsymbol{u}_3(t) + \boldsymbol{w}_3(t) \\ \dot{\boldsymbol{u}}_4(t) &= \boldsymbol{u}_4(t) \end{aligned} \quad \Longrightarrow \quad \dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + K\boldsymbol{w}(t) \tag{6.58}$$

with the matrices and the vectors

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad K = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}, \quad \boldsymbol{u}(t) = \begin{pmatrix} \boldsymbol{u}_1(t) \\ \boldsymbol{u}_2(t) \\ \boldsymbol{u}_3(t) \\ \boldsymbol{u}_4(t) \end{pmatrix}, \quad \boldsymbol{w}(t) = \begin{pmatrix} \boldsymbol{w}_1(t) \\ \boldsymbol{w}_2(t) \\ \boldsymbol{w}_3(t) \end{pmatrix} \tag{6.59}$$

Assuming the input is time-independent, i.e. $\boldsymbol{w}(t) \equiv \boldsymbol{w}$*, the system* (6.58) *can be expressed as*

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \\ \boldsymbol{w}_3 \end{pmatrix} = L\boldsymbol{u}(t) + \begin{pmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \\ \boldsymbol{w}_3 \\ 0 \end{pmatrix} = L\boldsymbol{u}(t) + \widehat{\boldsymbol{K}}\widehat{w} \tag{6.60}$$

with the adapted input

$$\widehat{\boldsymbol{K}} = \begin{pmatrix} \boldsymbol{w}_1 \\ \boldsymbol{w}_2 \\ \boldsymbol{w}_3 \\ 0 \end{pmatrix} \in \mathbb{R}^4, \quad \widehat{w} = 1 \tag{6.61}$$

This implies that the input of the reformulated system (6.60) *is accordingly one-dimensional instead of three-dimensional as before. Consequently, the costs of the subspace construction* (6.57) *can be reduced within the KSMOR method, which are directly depending on the size of the underlying input matrix.*

However, when executing the latter approach the discrepancy between the subspaces V and $\widehat{V}$ may be large due to the dimensional difference between the matrix K and the vector $\widehat{\boldsymbol{K}}$. This means, that the dimensional difference can be comprehended as a kind of information loss and affects the moment matching process.

Therefore, we modify this idea for simplification by building a compromise between V and $\widehat{V}$, and construct a new input matrix $\overline{K} \in \mathbb{R}^{N \times s}$, where the columns lie in a known

s-dimensional subspace of K with $s \ll p$. In doing so, the realisation of this procedure is based solely on *snapshots* from a numerical simulation of the dynamical system in a similar manner to the POD technique. More precisely, using snapshot data

$$W = [\boldsymbol{w}(t_1), \boldsymbol{w}(t_2), \ldots, \boldsymbol{w}(t_s)] = \begin{pmatrix} w_1(t_1) & w_1(t_2) & \cdots & w_1(t_s) \\ w_2(t_1) & w_2(t_2) & \cdots & w_2(t_s) \\ w_3(t_1) & w_3(t_2) & \cdots & w_3(t_s) \\ \vdots & \vdots & & \vdots \\ w_p(t_1) & w_p(t_2) & \cdots & w_p(t_s) \end{pmatrix} \in \mathbb{R}^{p \times s} \tag{6.62}$$

generated by a simulated model, we can construct the snapshot-based input matrix $\overline{K}$ and the subspace $\overline{V}$ via $\overline{K} = KW$ and $\overline{V} = \mathcal{K}_q((L - \sigma I)^{-1}, (L - \sigma I)^{-1}\overline{K})$, respectively. The original KSMOR method is then continued with K and $\boldsymbol{w}(t)$. This means, the methods KSMOR and KSMOR* differ only in the computation of the subspaces V and $\overline{V}$, which are based on the corresponding input matrices. The structure of the algorithm for the proposed KSMOR* method is sketched in Figure 6.5.

Remark 6.7. *For the proposed approach a numerical simulation is necessary, since the temperature vectors $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ within* (6.32) *are not known a priori.*

Remark 6.8. *The viability of our proposed method is only justified experimentally in this section. We will show this, first on a simple test example and second with a synthetic as well as real-world GES experiment. In doing so, the presented results are analysed both at the steady state solution and during the transient simulation.*

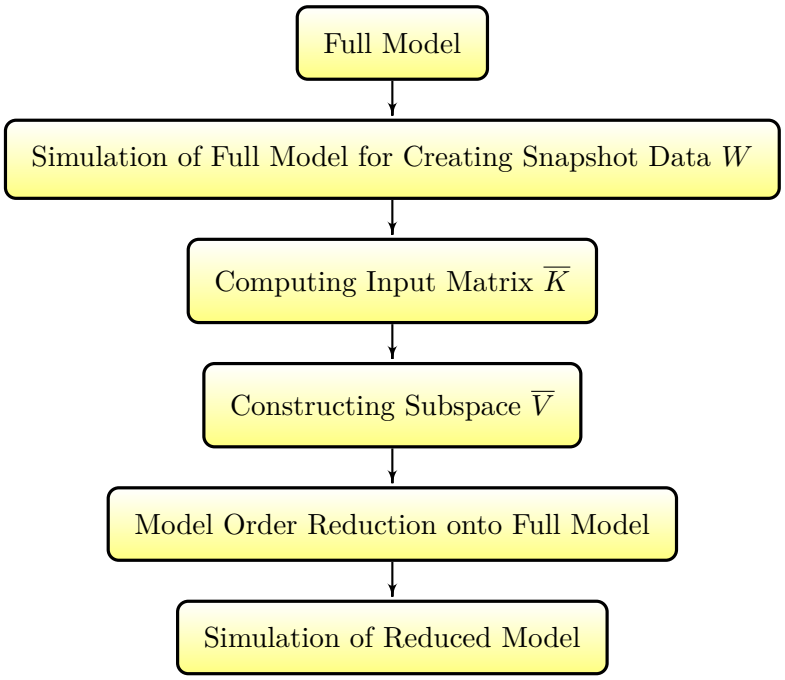


Figure 6.5: Structure of the algorithm for the KSMOR* method.

For the sake of completeness, let us briefly describe the differences to the approach performed in [232] in our GES setting. The authors also use the snapshot matrix (6.62), but in an indirect manner. In doing so, l singular vectors belonging to the largest singular values are calculated, which subsequently assembling the appropriate columns of the input matrix. Thus, the eigenvalues λ_i and eigenvectors $\boldsymbol{\phi}_i$ of

$$Y = W^\top W \in \mathbb{R}^{s\times s} \tag{6.63}$$

are computed, and the corresponding singular vectors are given by

$$\overline{\boldsymbol{w}}_i = \frac{W\boldsymbol{\phi}_i}{\sqrt{\lambda_i}}, \quad i = 1, \ldots, s \tag{6.64}$$

Finally, the dominant l modes with $l \leq s$ form the matrix

$$\overline{W} = [\overline{\boldsymbol{w}}_1, \overline{\boldsymbol{w}}_2, \ldots, \overline{\boldsymbol{w}}_l] \tag{6.65}$$

so that the snapshot-based input matrix is constructed by means of $\overline{K} = K\overline{W} \in \mathbb{R}^{N\times l}$. It should be noted that the latter ansatz of computing the input matrix based on snapshots is not performed in exactly the same way in [232], since they have a different model problem as considered here. In the following we will abbreviate the approach as KSMOR-SVD.

In contrast to KSMOR* the latter approach has two drawbacks, mainly in terms of computational costs and degrees of freedom. On the one hand, the additional solution of an eigenvalue problem of size $s \times s$ increases the computational costs. On the other hand, the number of dominant singular values used is not evident and is set manually depending on the problem being solved. Therefore, a good trade-off between the number of snapshots selected and the dominant singular values is essential.

Let us end this section with a numerical evaluation on a test example. In doing so, we compare the numerical solutions that are generated by the methods KSMOR* and KSMOR-SVD. In addition, we also study the performance of the POD technique, which can be easily employed without further modifications for problems with many inputs.

Example 6.2. *Apart from the numerical comparison, we describe here more detailed the realisation of KSMOR* at hand on a selected numerical experiment. In particular, we demonstrate the general viability of our approach, which makes it a candidate for solving large-scale dynamical systems with a large number of inputs in an efficient way.*

For the sake of completeness and to shed light on the technical differences between the approaches, we first analyse the quality of the solutions obtained through CN, KSMOR and KSMOR* according to the L_2-error (6.77) *recorded at a specific time t_F. Second, we compare the performance of KSMOR* with that of KSMOR-SVD and POD.*

For an evaluation, let us consider a two-dimensional linear heat equation on a rectangular domain $\Omega = [0, 3/2] \times [0, 2]$ equipped with time- and space-dependent boundary conditions. The diffusion problem reads as

$$\partial_t u\,(x, y, t) = a\partial_{xx} u\,(x, y, t) + b\partial_{yy} u\,(x, y, t) + f\,(x, y)\,, \quad (x, y, t) \in \Omega \times [0, t_F] \tag{6.66}$$

where the fluxes and the source term are given by

$$a = \frac{1}{18\pi^2}, \quad b = \frac{1}{2\pi^2}, \quad f\,(x, y) = 2\cos\left(3\sqrt{2}\pi x\right)\cos\left(\sqrt{2}\pi y\right) \tag{6.67}$$

The corresponding external boundary conditions are fixed to

$$\begin{aligned} g_1(y,t) &= u(0,y,t) = \cos\left(\sqrt{2}\pi y\right) \\ g_2(y,t) &= u(3/2,y,t) = \cos(\pi y)\, e^{-t} + \cos\left(\tfrac{9\sqrt{2}}{2}\pi\right)\cos\left(\sqrt{2}\pi y\right) \\ h_1(x,t) &= u(x,0,t) = \sin(3\pi x)\, e^{-t} + \cos\left(3\sqrt{2}\pi x\right) \\ h_2(x,t) &= u(x,2,t) = \sin(3\pi x)\, e^{-t} + \cos\left(3\sqrt{2}\pi x\right)\cos\left(2\sqrt{2}\pi\right) \end{aligned} \tag{6.68}$$

with the initial condition specified as

$$u(x,y,0) = \sin(3\pi x)\cos(\pi y) + \cos\left(3\sqrt{2}\pi x\right)\cos\left(\sqrt{2}\pi y\right) \tag{6.69}$$

The semi-discrete system can be expressed in the form

$$\dot{\boldsymbol{u}}(t) = L\boldsymbol{u}(t) + \boldsymbol{g}_1 + K_{g_2}\boldsymbol{g}_2(t) + K_{h_1}\boldsymbol{h}_1(t) + K_{h_2}\boldsymbol{h}_2(t) + \boldsymbol{f}, \quad \boldsymbol{u}(0) = \boldsymbol{u}^0 \tag{6.70}$$

or as a transformed system with zero initial conditions

$$\dot{\widetilde{\boldsymbol{u}}}(t) = L\widetilde{\boldsymbol{u}}(t) + \widetilde{K}\widetilde{\boldsymbol{w}}(t), \quad \widetilde{\boldsymbol{u}}(0) = \boldsymbol{0} \tag{6.71}$$

with input matrix and input

$$\widetilde{K} = \left[\boldsymbol{g}_1, K_{g_2}, K_{h_1}, K_{h_2}, \boldsymbol{f}, L\boldsymbol{u}^0\right], \quad \widetilde{\boldsymbol{w}}(t) = \left(1, \boldsymbol{g}_2(t), \boldsymbol{h}_1(t), \boldsymbol{h}_2(t), 1, 1\right)^\top \tag{6.72}$$

Furthermore, we set $h = 0.02$, $\tau = 0.001$ and $t_F = 5$. In this case, the number of grid points amounts to $N = 7676$ and the individual input dimensions result in $\boldsymbol{g}_2 \in \mathbb{R}^{101}$ and $\boldsymbol{h}_1, \boldsymbol{h}_2 \in \mathbb{R}^{76}$ with $K_{g_2} \in \mathbb{R}^{7676\times 101}$ and $K_{h_1}, K_{h_2} \in \mathbb{R}^{7676\times 76}$, respectively. In total, the input and the corresponding input matrix are of size $\widetilde{\boldsymbol{w}} \in \mathbb{R}^{256}$ and $\widetilde{K} \in \mathbb{R}^{7676\times 256}$, respectively.

To compare the performances of KSMOR and KSMOR*, the result of the CN method is used as a reference solution. We note that the computational costs of CN for the fixed parameters amount to 5.97 seconds. Obviously, the number of subspaces used within the KSMOR methods is still a free parameter and influences the approximation quality as well as the efficiency of the numerical scheme, see Figure 6.6. As expected, KSMOR leads to a reduction of the L_2-error as the number of subspaces is increased, but the computational costs are significantly higher compared to CN. Even if the subspace order $q = 1$ is used, the costs are almost twice as high. Thus, this experiment clarifies the inefficiency of KSMOR for problems with a large number of inputs. Conversely, the proposed KSMOR* method can be applied to tackle this problem. For the application of this approach, snapshot data are required for $\boldsymbol{g}_2$, $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$, while the other inputs are constant and can be ignored. To construct the snapshot-based input matrix $\overline{K} \in \mathbb{R}^{7676\times(3s+3)}$ and the corresponding subspace $\overline{V}$, the model is simulated using the CN method for $\tau = \frac{t_F}{s}$ and the input data is stored as

$$W_1 = \left[\boldsymbol{g}_2(t_1), \ldots, \boldsymbol{g}_2(t_s)\right], \quad W_2 = \left[\boldsymbol{h}_1(t_1), \ldots, \boldsymbol{h}_1(t_s)\right], \quad W_3 = \left[\boldsymbol{h}_2(t_1), \ldots, \boldsymbol{h}_2(t_s)\right] \tag{6.73}$$

with $W_1 \in \mathbb{R}^{101\times s}$ and W_2, $W_3 \in \mathbb{R}^{76\times s}$. The snapshot-based matrix is then determined via

$$\overline{K} = \left[\boldsymbol{g}_1, K_{g_2}W_1, K_{h_1}W_2, K_{h_2}W_3, \boldsymbol{f}, L\boldsymbol{u}^0\right] \tag{6.74}$$

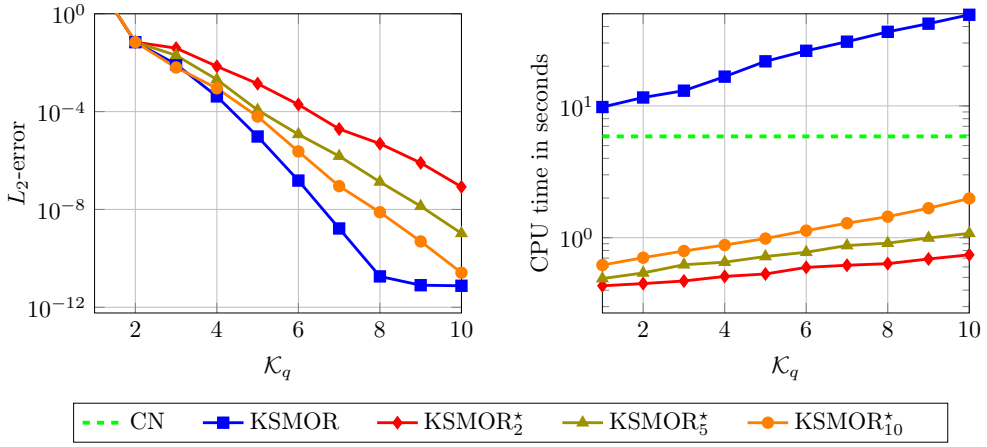


Figure 6.6: Results for the test example (6.66)-(6.69) at a fixed time $t_F = 5$. Comparison of the L_2-error (**left**) and the CPU time (**right**) between KSMOR and KSMOR* for a different number of Krylov subspaces $\mathcal{K}_q$. The reference solution is computed with the CN method. In experiments we study different numbers of snapshots $s = 2, 5, 10$ used for the KSMOR* method, which is abbreviated as KSMOR$^\star_s$. Obviously, the KSMOR* technique is much more efficient compared to the original KSMOR method.

After the construction of $\overline{V} = \mathcal{K}_q((L - \sigma I)^{-1}, (L - \sigma I)^{-1}\overline{K})$ *with* $\sigma = 0$, *the dimension reduction is applied to the original system* (6.71). *For the sake of convenience, the latter snapshot dependence on* s *is abbreviated to KSMOR*$^\star_s$. *The results for KSMOR*$^\star_2$, *KSMOR*$^\star_5$ *and KSMOR*$^\star_{10}$ *shown in Figure 6.6 clearly demonstrate a much higher efficiency compared to CN and KSMOR. For example, using KSMOR*$^\star_5$ *with* $q = 10$ *results in an excellent trade-off between quality and efficacy and can save approximately 95% and 80% of the CPU time in relation to KSMOR and CN, respectively.*

When using the KSMOR-SVD method, the snapshot data are first assembled into one matrix, i.e.

$$W = \left[K_{g_2} W_1, K_{h_1} W_2, K_{h_2} W_3\right] \tag{6.75}$$

Based on (6.63)-(6.65) *the* l *dominant singular vectors form then the matrix* $\overline{W} \in \mathbb{R}^{N \times l}$ *with* $l \leq s$. *Afterwards, the reduced input matrix can be assembled by means of*

$$\overline{K} = \left[\boldsymbol{g}_1, \overline{W}, \boldsymbol{f}, L\boldsymbol{u}^0\right] \in \mathbb{R}^{7676 \times (l+3)} \tag{6.76}$$

and the subspace $\overline{V}$ *is constructed. First, Figure 6.7 shows an investigation of KSMOR-SVD to* (6.66)-(6.69) *in terms of varying numbers of snapshots* $s = 2, 5, 10$ *combined with a complete spectrum* $l = 3s$. *The method produces results with the same accuracy compared to KSMOR**, *but at the same time causes to higher computational costs. In Figure 6.8 the experiment is repeated with* $s = 5$ *for a different number of dominant singular values* $l = 2, 5, 10, 15$, *whereby* $l = 10$ *gives the best trade-off. Based on these investigations a comparison between KSMOR** *and KSMOR-SVD with* $s = 5, l = 10$ *is visualised in Figure 6.9. For this test example, our proposed KSMOR** *approach is most efficient. In conclusion, both approaches are absolutely beneficial for problems with large-scale input and can remarkably reduce the computational costs compared to the original KSMOR method as well as the implicit CN scheme.*

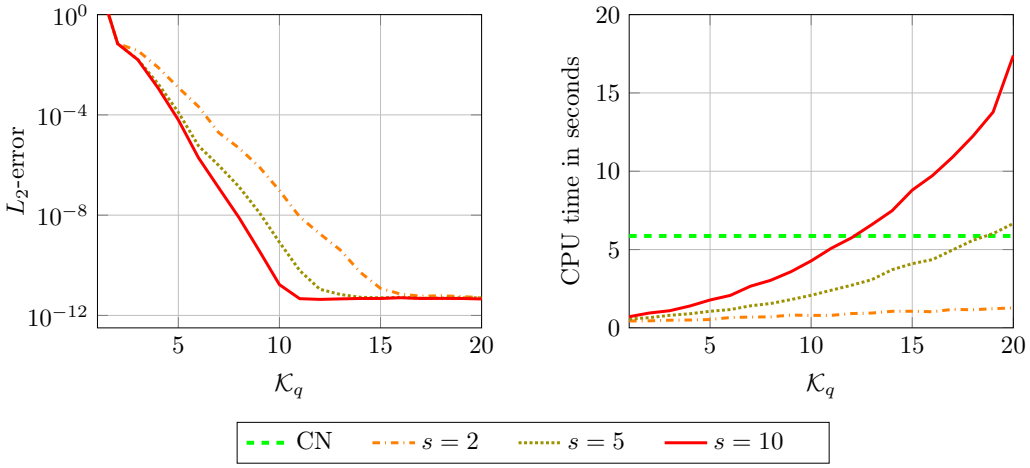


Figure 6.7: Results for the test example (6.66)-(6.69) at a fixed time $t_F = 5$. Comparison of the L_2-error (**left**) and the CPU time (**right**) using KSMOR-SVD for a different number of Krylov subspaces $\mathcal{K}_q$. The reference solution is computed with the CN method. In experiments we study different numbers of snapshots $s = 2, 5, 10$ used for the KSMOR-SVD method. Let us mention, that we apply a complete SVD ($l = 3s$), without selecting dominant singular values. The results are equally accurate compared to KSMOR*, but also linked to higher computational costs, cf. Figure 6.6.

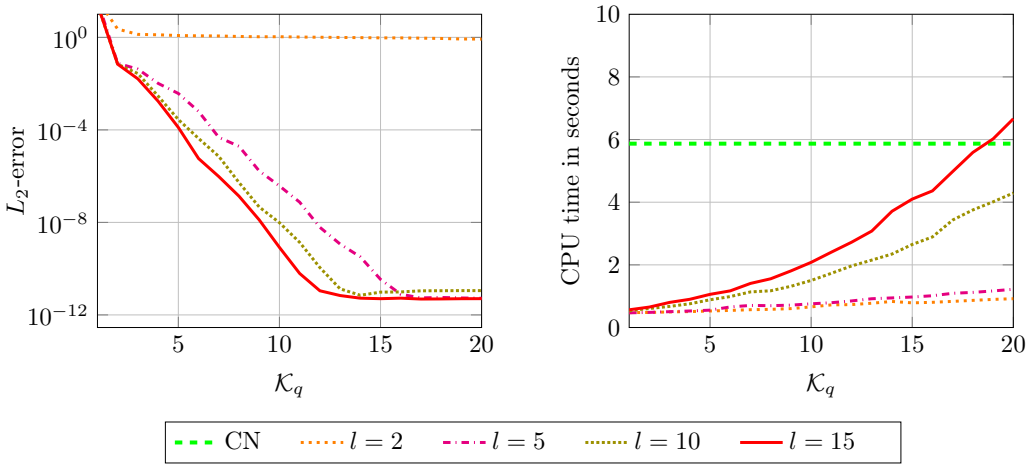


Figure 6.8: Results for the test example (6.66)-(6.69) at a fixed time $t_F = 5$. Comparison of the L_2-error (**left**) and the CPU time (**right**) using KSMOR-SVD for $s = 5$ and a different number of Krylov subspaces $\mathcal{K}_q$. The reference solution is computed with the CN method. In total, the spectrum amounts to $l = 15$. In this experiment we study different numbers of dominant singular values $l = 2, 5, 10, 15$ used for the KSMOR-SVD method.

Lastly, we also examine the POD method, which can be easily used in case of large-scale inputs. The results presented in Figure 6.10 demonstrate that the POD technique cannot maintain the performances of KSMOR* and KSMOR-SVD. In order to be competitive in this case, POD requires a very large number of snapshots.

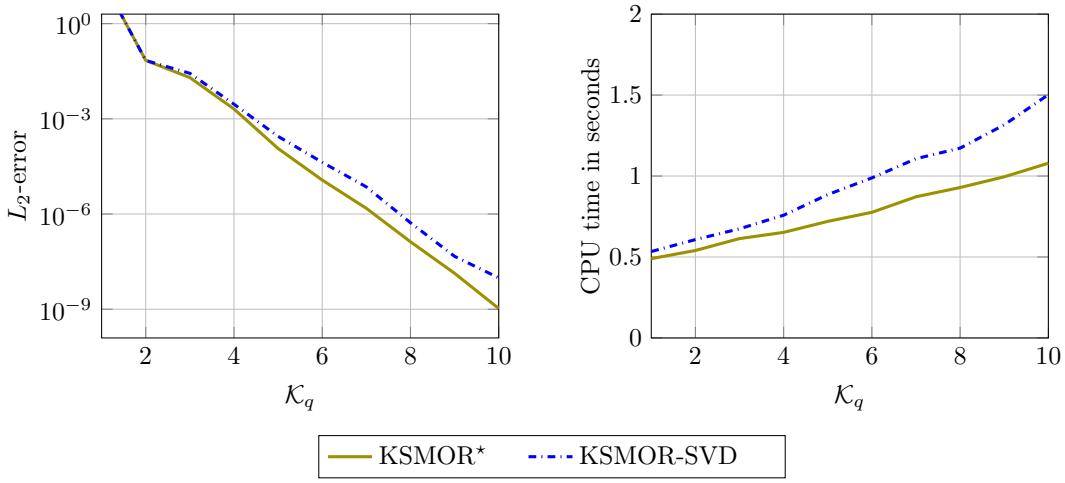


Figure 6.9: Results for the test example (6.66)-(6.69) at a fixed time $t_F = 5$. Comparison of the L_2-error (**left**) and the CPU time (**right**) between KSMOR* and KSMOR-SVD for $s = 5$ snapshots and a different number of Krylov subspaces $\mathcal{K}_q$. The reference solution is computed with the CN method. The number of dominant singular values for KSMOR-SVD is fixed to $l = 10$. Obviously, the KSMOR* technique is more efficient compared to the KSMOR-SVD method. Nevertheless, both approaches can significantly reduce the computational costs compared to the original KSMOR method and the implicit CN scheme.

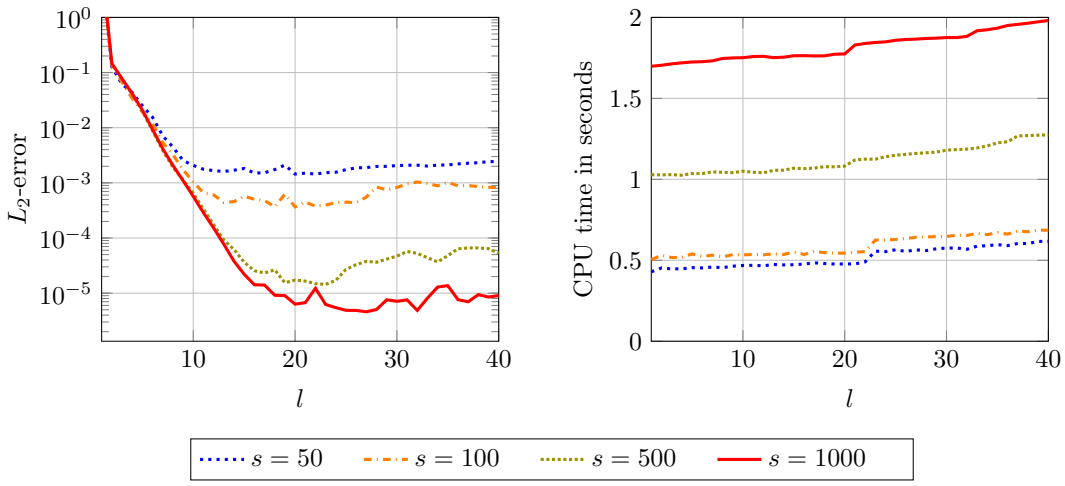


Figure 6.10: Results for the test example (6.66)-(6.69) at a fixed time $t_F = 5$. Comparison of the L_2-error (**left**) and the CPU time (**right**) using POD for a different number of dominant singular values l. The POD method is studied using various snapshots $s = 50, 100, 500, 1000$. The reference solution is computed with the CN method. The results are less accurate compared to KSMOR* and KSMOR-SVD, although a relatively large number of snapshot data are applied.

6.5 Comparison of Solvers for Long-Term GES Simulation

The numerical long-term simulation of the GES model considered in this chapter, is mainly based on the explicit FEDRK method, the implicit sparse direct/iterative solver and the KSMOR* technique. In the following we evaluate the numerical solvers by two different experiments and assess their performance in terms of approximation quality and CPU time. As methods of choice we consider:

1. EE method. The upper bound $\tau_{\max}$ is given by Proposition 6.2.
2. FEDRK scheme. $\tau_{\max}$ is known, the number M of cycles is the only tuning parameter.
3. Sparse direct solver applied to the implicit CN method (called *direct CN*). The internal MATLAB-function *decomposition*[3] including a Cholesky factorisation is used. The tuning parameter is the time step size τ.
4. Sparse iterative solver using CG or PCG applied to the implicit CN method (called *iterative CN*). User-defined parameters are the time step size τ and the tolerance $\varepsilon > 0$, the latter parameter is required to terminate the algorithm using the relative residual (2.136). The preconditioners IC or MIC are employed for PCG. Another tuning parameter $\gamma > 0$ is required, which corresponds to a numerical fill-in strategy IC(γ) or MIC(γ). Furthermore, the initial condition within CG and PCG is a user-defined issue, so we test the current heat distribution as initialisation $\boldsymbol{x}_0 = \boldsymbol{u}^k$.
5. KSMOR* method. This technique can be tuned by the number of projection subspaces $\mathcal{K}_q$ and snapshot data $W = [\boldsymbol{w}(t_1), \boldsymbol{w}(t_2), \ldots, \boldsymbol{w}(t_s)]$ used. The sparse direct solver is applied to compute the Krylov subspace generated by the Block Arnoldi method. The reduced model is then solved for a selected τ via the direct CN scheme.

To evaluate the accuracy of the methods the solution of the EE scheme is used as reference. Based on a simplified GES application without a source term, we first examine the optimal selection of the parameters within the methods used. Second, we validate the mathematical model on real data including sources. In addition, a uniform grid with $\Delta x = \Delta y = h$ is considered in all experiments and the evaluation is measured by the L_2-error defined as

$$L_2(\boldsymbol{u}, \widetilde{\boldsymbol{u}}) = \sqrt{\sum_{i=1}^{N} (u_i - \widetilde{u}_i)^2} \tag{6.77}$$

between the analytical/reference solution $\boldsymbol{u}$ and the numerical solution $\widetilde{\boldsymbol{u}}$.

Remark 6.9. *All experiments were done in MATLAB R2018b with an Intel Xeon(R) CPU E5-2609 v3 CPU. All CPU times presented incorporate the modelling (linear system, preconditioning, reduction) and the numerical resolution. Therefore, the performances are easily comparable.*

6.5.1 Geothermal Energy Storage Simulation without Source

First, we consider the simulation of a GES under real-world conditions without sources/sinks. The heat tank is assumed to be placed underground, closed on its sides and upwards, compare

[3] The same performance is also achieved by the object-oriented factorisation of SuiteSparse [74] in undocumented tests.

Figure 6.1. The lower part is open to accommodate the heat delivery. The thermophysical parameters of the materials used are assumed to be constant and are given in Table 6.1. In this setup, we specify that the upper and the lower ground have the same material properties, and further parameters are $\alpha_c = 0.5$ and $\alpha_A = 10$. As mentioned in Section 6.2.3, the external conditions are given via time-dependent Robin (6.6), time- and space-dependent Dirichlet (6.8) and time-dependent Dirichlet boundary conditions (6.10) with $b_1 = 0.5$ and $b_2 = 10$.

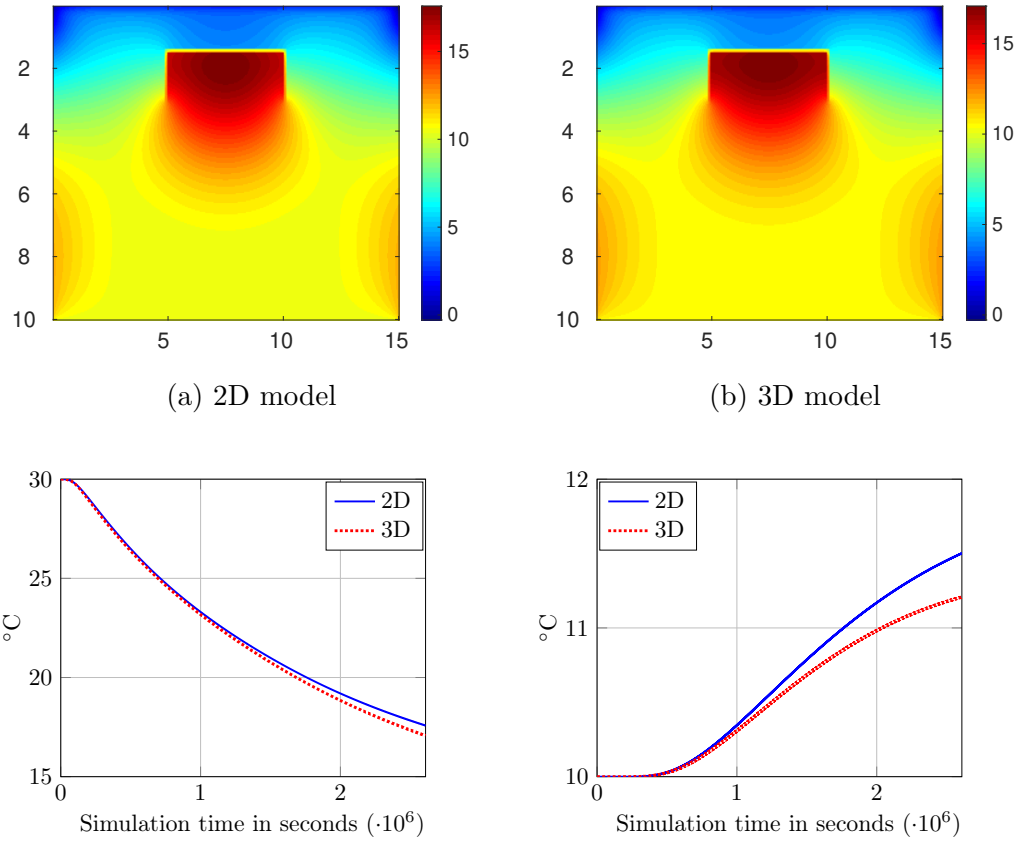
For the evaluation, a two-dimensional rectangular domain is considered given by $[0, 15] \times [0, 10]$ in meters, whereby the heat tank with size 5×1.5 meters has an installation depth of 1.5 meters. The thickness of the insulation is determined to 12 centimetres. Moreover, a mid-size model problem is considered by setting $h = 0.04$ centimetres ($N = 94376$ grid points) and the stopping time is assumed to be $t_F = 2.609 \cdot 10^6$ seconds (around 30 days). As initialisation, the temperature is fixed to 30 degrees for the heat tank, 20 degrees for the insulation and 10 degrees otherwise.

Let us first discuss the spatial dimensioning of the underlying model. The original geothermal field is given in 3D, but it is clear that its fundamental simulation is computationally more expensive than a two-dimensional model. In fact, however, a full three-dimensional simulation is not necessary, which can be explained as follows: the considered model problem is isotropic, homogeneous and linear, and almost symmetric with respect to the spatial environment. As a result, the physics of the three-dimensional continuous model can be reproduced by considering a cross section of a 3D geothermal field in 2D. Besides the purely physical justification, we confirm the reduction in the spatial model dimensions by comparing the results in 2D and 3D using the EE method. Later it will also be shown that the approach is justified by taking real data into account.

Results on EE According to Proposition 6.2, the time step size restriction for the two-dimensional model problem is given by $\tau_{\max,2D} \approx 417.44$ seconds. On this basis, 6250 iterations have to be performed to reach the final stopping time t_F, whereby the corresponding computational costs of EE amount to 11.1 seconds. In contrast, the upper stability bound of the underlying three-dimensional problem is determined with $\tau_{\max,3D} \approx 278.145$ such that 9380 iterations are required for the simulation. In this case, the discrete domain is defined by $N = 35485376$ grid points, which enlarges the size of the system matrix considerably and thus increases the computational effort. In consequence, the entire simulation requires around 20000 seconds which is considered to be unsatisfactory. As expected, Figure 6.11 confirms the use of a two-dimensional model as a substitute. First, it shows almost equal results when comparing the 2D and 3D model, in which the latter is evaluated via a cross section along the

Table 6.1: Thermophysical properties of the materials involved for the synthetic experiment.

Material/Layer	Conductivity λ (W/(m·K))	Density ρ (kg/m^3)	Specific heat c (J/(kg·K))	Diffusivity a (m^2/s)
Water-saturated soil	2.3	2100	1143	$9.5821 \cdot 10^{-7}$
Insulation walls	0.03	100	1000	$3 \cdot 10^{-7}$



(a) 2D model

(b) 3D model

(c) Temporal temperature of S1

(d) Temporal temperature of S2

Figure 6.11: Results of the GES experiment without source: visual representation of the 2D and 3D solution at $t_F = 2.609 \cdot 10^6$ by applying the EE method for $h = 0.04$. **(a)** Result of the 2D model after 6250 iterations with $\tau_{\max,2D} = 417.44$. **(b)** Result of the 3D model after 9380 iterations with $\tau_{\max,3D} = 278.145$, visualised in form of a cross section along the middle of the 3D-GES. **(c)** Temporal temperature profile S1 fixed in the centre of the heat tank. **(d)** Temporal temperature profile S2 fixed in the centre between heat tank and bottom border.

middle of the 3D-GES. Second, a comparison of two artificial temporal temperature profiles S1 and S2 demonstrate the expected similarities of the 2D and 3D simulation results. In the further course, we therefore compare the proposed methods using the 2D-GES model on both synthetic and real data.

Let us emphasise that for the evaluation of the solvers no analytical solution is available and a reference solution is required. In general, the EE method provides the most precise solution that can be used as reference solution for a fair comparison in this section. To this end, we assess the L_2-error at a fixed moment in time and the corresponding CPU time between the solutions of EE and the tested solver. In doing so, we do not consider a full error computation along the temporal axis because the methods yield completely different sampling rates, which potentially leading to unfair comparisons.

Results on Direct CN In a first test, we examine the performance using the sparse direct solver applied to IE and CN. The results illustrated in Figure 6.12 demonstrate the expected superiority of the CN scheme due to its second order accuracy in time. For this reason, the IE method is no longer considered here for the GES application.

Results on Iterative CN We now study the iterative solver and its parameters involved. For this purpose, the influence of the stopping criterion on the accuracy and the CPU time is analysed in Figure 6.13. The value $\varepsilon = 10^{-4}$ provides the best trade-off and is assumed to be fixed for all subsequent tests. Due to the size of the system matrix a preconditioning is suitable. By consideration of the specified preconditioners, Figure 6.14 shows that MIC(10^{-2}) outperforms both CG and IC(10^{-3}). Based on this investigation, we use MIC(10^{-2}) for all further experiments.

Since the heat conduction flow modelled by a linear heat equation is rather slow, it is to be expected that the approximation $\boldsymbol{u}^k$ and $\boldsymbol{u}^{k+1}$ do not differ strongly. Therefore, it may be useful to employ the last heat distribution $\boldsymbol{u}^k$, as initialisation within the CG and PCG method, for the computation of the new iterate $\boldsymbol{u}^{k+1}$. This strategy is investigated in Figure 6.15 for the iterative CN solver. Using the strategy $\boldsymbol{x}_0 = \boldsymbol{u}^k$ is highly beneficial for the CG method and can significantly improve the efficiency. In contrast, no substantial improvements are achieved for the MIC preconditioner.

Results on FEDRK We now experimentally confirm the considerations mentioned in Section 6.4.1. For realisation the differences between the basic scheme and the adapted versions using inner updates of $\boldsymbol{w}^{m,k}$ are examined in Figure 6.16. In case of the basic scheme we only consider the FEDRK technique, since both methods yield equally numerical solutions. Let us first note that FED and FEDRK using inner updates cannot deliver the same output anymore because the schemes built on different integration points. For convenience only we denote the adapted versions in this paragraph as *FED-updated* and *FEDRK-updated*. As

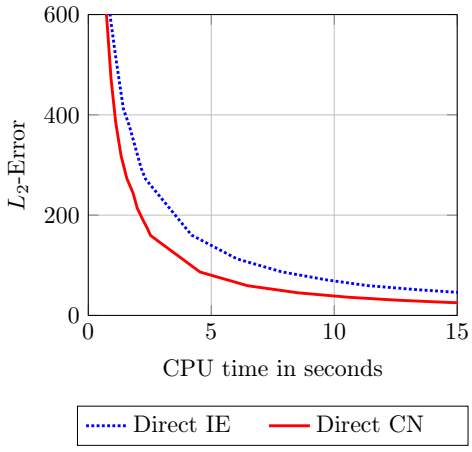


Figure 6.12: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time between direct IE and direct CN at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. The CN scheme clearly outperforms the IE method in terms of efficiency.

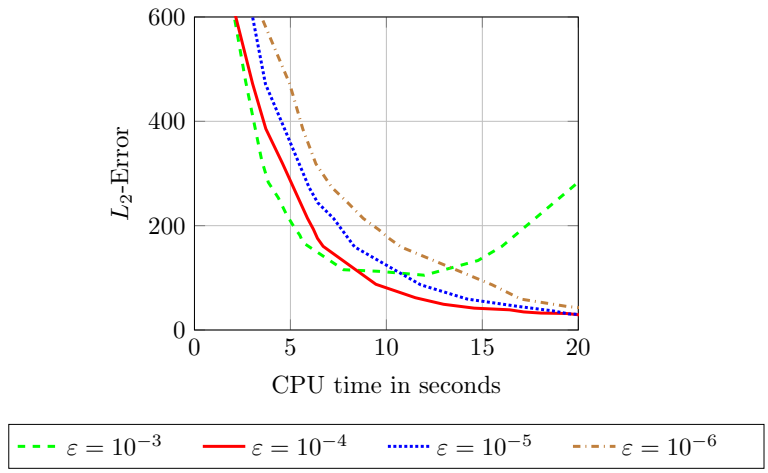


Figure 6.13: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time for iterative CN using the CG method and different values ε at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. The choice $\varepsilon = 10^{-4}$ provides the best trade-off.

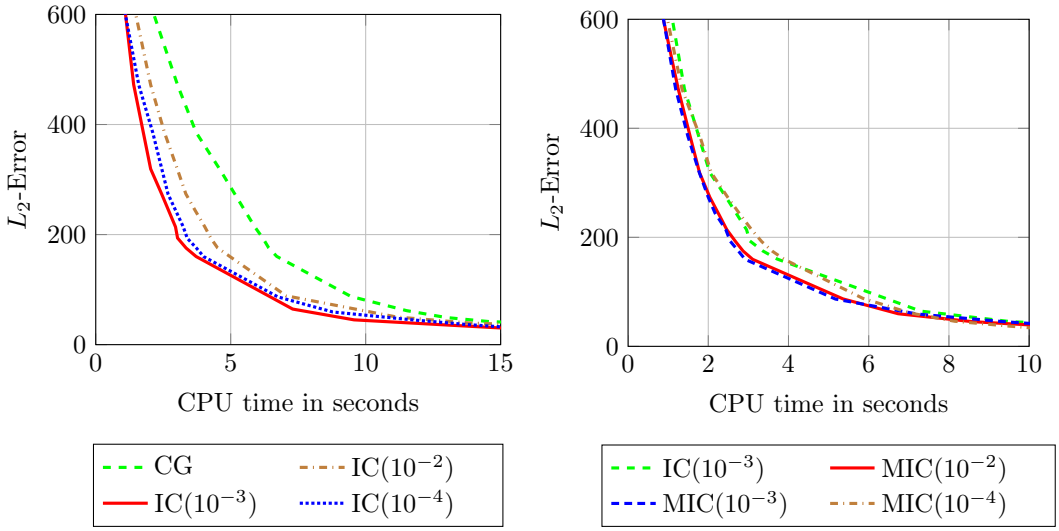


Figure 6.14: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time using iterative CN at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$. The stopping criterion is fixed to $\varepsilon = 10^{-4}$. The reference solution is computed with the EE method. **Left:** Iterative CN is computed using IC-preconditioner for different drop tolerances γ. The choice $\gamma = 10^{-3}$ provides the best trade-off. **Right:** Iterative CN is computed by IC(10^{-3}) and MIC-preconditioner for different drop tolerances γ. A value for $\gamma \in [10^{-3}, 10^{-2}]$ is optimal.

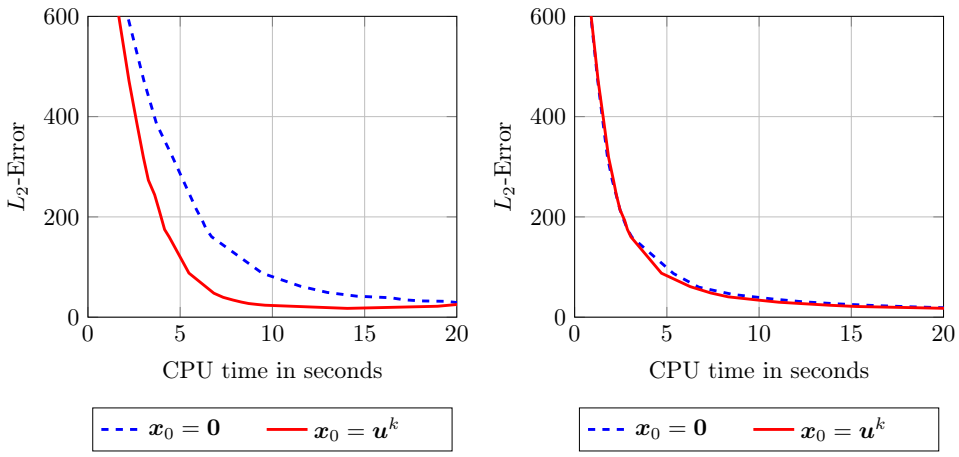


Figure 6.15: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time for iterative CN at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$ using the initialisation $\boldsymbol{x}_0 = \boldsymbol{u}^k$. The stopping criterion is fixed to $\varepsilon = 10^{-4}$. The reference solution is computed with the EE method. **Left:** Iterative CN is computed by CG. **Right:** Iterative CN is computed by MIC(10^{-2}). The strategy $\boldsymbol{x}_0 = \boldsymbol{u}^k$ is a highly useful tool for the effectiveness of the CG method. Contrary, no significant improvements are achieved for MIC.

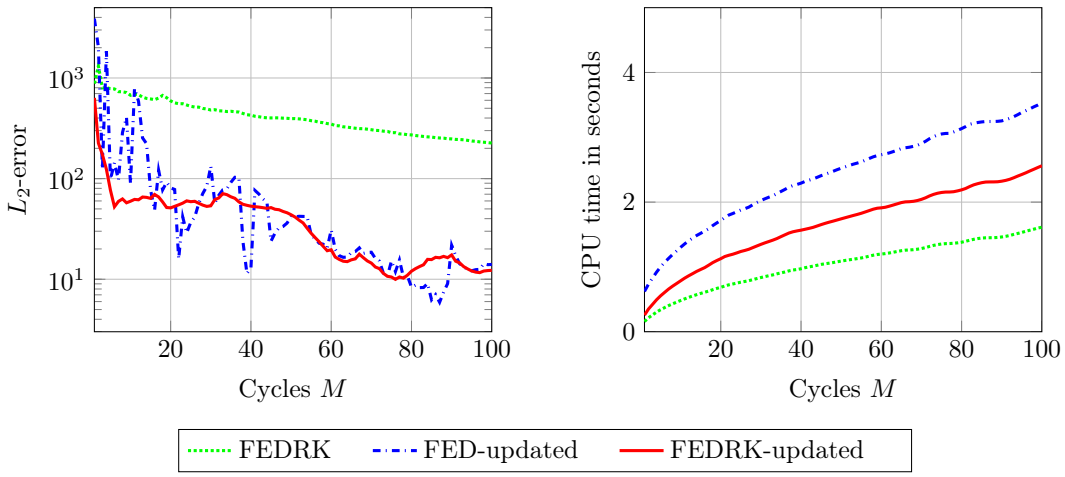


Figure 6.16: Results of the two-dimensional GES experiment without source: comparison of the computed L_2-error (**left**) and the corresponding CPU time (**right**) between FEDRK, FED-updated and FEDRK-updated for a different number of cycles M at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. To ensure numerical stability within the FED method we apply a Leja-ordering with $\kappa = 7$, see [107]. Inner updates significantly improve the performance of both schemes. However, FED-updated may lead to highly unstable intermediate solutions and therefore also affects the final solution, especially if a small number of M cycles is used. Contrary, FEDRK-updated is absolutely stable and causes lower computational costs than FED-updated.

expected, FED-updated is highly unstable, especially for a small number of cycles M. The latter fact is due to unstable intermediate solutions as a consequence of a rearrangement, where we used the Leja ordering with $\kappa = 7$ as suggested in [107]. This implies the use of a relatively large cycle Number M, nevertheless, a stable solution is not guaranteed. In contrast, FEDRK-updated provides stable solutions and is additionally much more efficient. Using the basic FEDRK scheme is less computationally intensive than using FEDRK-updated, at the cost of a significant loss of performance. The lack of efficiency between the two FEDRK versions is also presented in Figure 6.17.

Results on KSMOR* Due to external, lateral time- and space-dependent Dirichlet boundary conditions and internal matching conditions, the model has to deal with many inputs. In particular, the input variable results in $\widetilde{\boldsymbol{w}}(t) \in \mathbb{R}^{1106}$, with $\boldsymbol{u}_2(t) \in \mathbb{R}^{852}$ and $\boldsymbol{T}_g(t) \in \mathbb{R}^{251}$, while $\boldsymbol{u}_1$ does not have to be taken into account due to the identical material properties of both soils (upper and lower ground).

The performance of the original KSMOR technique for $q = 2, 3, 5$ and $\sigma = 0$ is illustrated in Figure 6.18. The reduction technique provides very reasonable results, but is also linked to extremely high computational costs. This is hardly surprising as a common weakness of KSMOR is the handling of a large-scale input.

To overcome this limitation, we apply the KSMOR* approach proposed in Section 6.4.3. Initially, we fix $q = 2$ and vary the size of the snapshot data as visualised on the left in Figure 6.19. Obviously, increasing s leads to better approximations. However, a saturation occurs for high values s, for which $s = 20$ provides the best performance trade-off. In a second test, we increase the number of subspaces q if $s = 20$ is used, cf. on the right in Figure 6.19. As expected, increasing moment matching improves the accuracy. Unfortunately, large values q are associated with high computational costs. Overall, KSMOR* is much more efficient compared to the original KSMOR method.

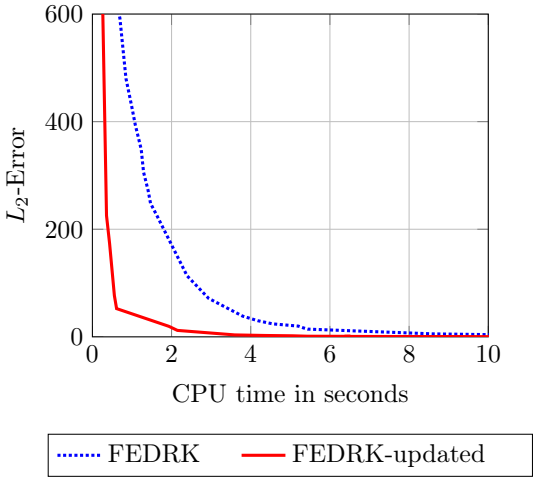


Figure 6.17: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time between FEDRK and FEDRK-updated at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. The scheme FEDRK-updated clearly outperforms their counterpart without inner updates.

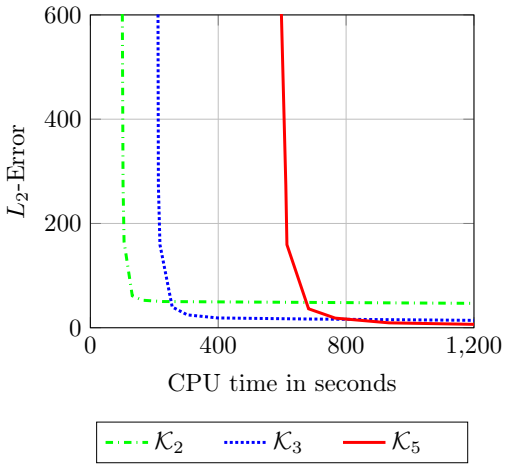


Figure 6.18: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time using KSMOR at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$. KSMOR is used by constructing subspaces with $\widetilde{\boldsymbol{w}}(t) \in \mathbb{R}^{1106}$, $\widetilde{K} \in \mathbb{R}^{N \times 1106}$ and $\sigma = 0$ for $q = 2, 3, 5$. The reference solution is computed with the EE method. The results are accurate, but also linked to extremely high computational costs.

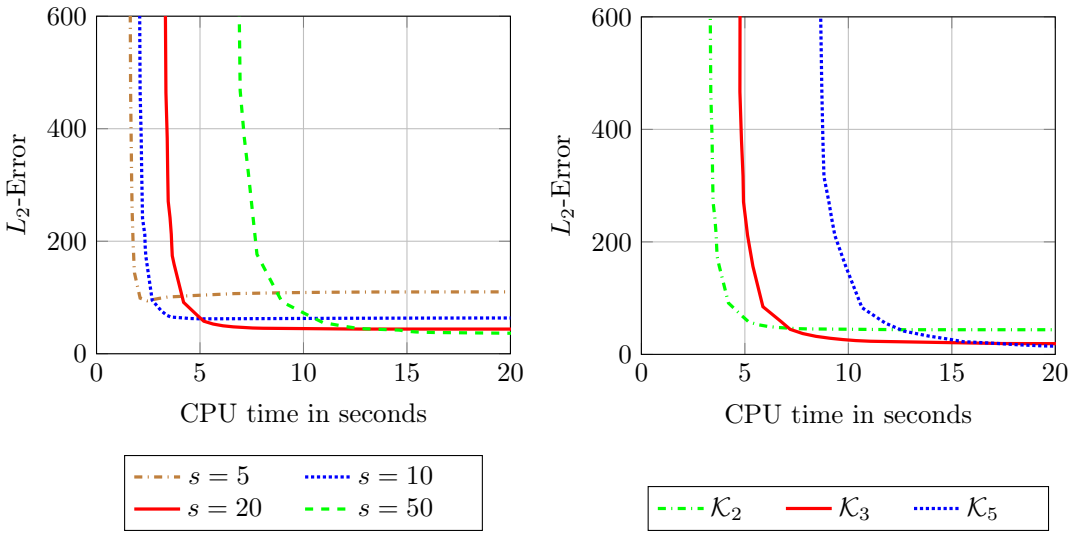


Figure 6.19: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time using KSMOR* at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$. The reference solution is computed with the EE method. **Left:** KSMOR* is applied with $q = 2$ and a varying size of snapshot data for $s = 5, 10, 20, 50$. **Right:** KSMOR* is used by constructing subspaces with $s = 20$ for $q = 2, 3, 5$. The best compromise in performance is achieved with $s = 20$ and $q = 3$. Compared to the original KSMOR method the approximations are computed dramatically fast.

Comparison of the Solvers Finally, a full comparison of the methods tested – direct CN, iterative CN using CG with $\boldsymbol{x}_0 = \boldsymbol{u}^k$ as well as MIC, FEDRK with inner updates and KSMOR* – is shown on the left in Figure 6.20.

Solving the CN method with the sparse direct solver is more efficient than with the iterative solver. Only for more accurate solutions, which correspond to small time step sizes τ, both implicit solvers perform comparably well due to a faster convergence of the iterative solver.

The KSMOR* technique, in which the parameters $s = 20$ and $q = 3$ are fixed, achieves the worst performance in comparison to the other methods. In contrast to the investigated test example in Section 6.4.3, the selected number of snapshots has to be relatively large. In combination with the number of Krylov subspaces linked to expensive offline precomputations and the additional enlargement of the reduced solution (6.54), the KSMOR* approach cannot maintain the efficiency of the other solvers.

The results on the left in Figure 6.20 clearly demonstrate the superior efficiency of FEDRK, which outperforms all other methods. This is mainly explained by two facts: first, FEDRK is an explicit method and is built on cheap matrix-vector multiplications. Second, the input $\boldsymbol{w}(t)$ can be updated within one cycle. And exactly this update process, including the external and internal boundary conditions, that is of great importance for an accurate approximation.

Finally, the performances of the proposed methods should also be considered using a finer grid, especially in the case of explicit methods in which the associated stability requirement depends quadratically on the spatial grid size. Apart from that, the computational cost of solving linear equations belonging to very large systems is also of interest. Therefore, we rerun the experiment for $h = 0.01$ which corresponds to $N = 1502501$ grid points. In this

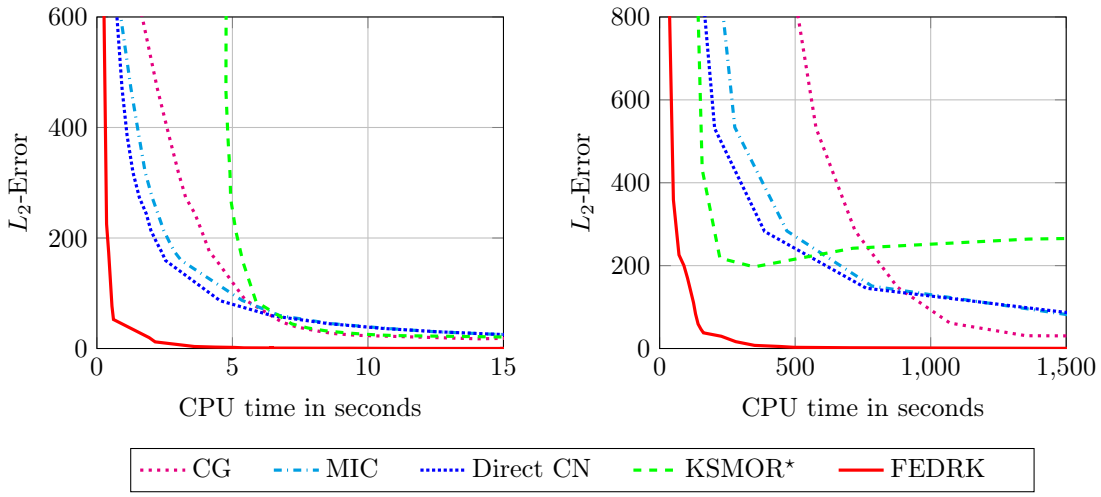


Figure 6.20: Results of the two-dimensional GES experiment without source: comparison of the L_2-error and the corresponding CPU time between iterative and direct CN, KSMOR* as well as FEDRK at $t_F = 2.609 \cdot 10^6$. The iterative methods are based on CG (with initialisation $\boldsymbol{x}_0 = \boldsymbol{u}^k$) and MIC($10^{-2}$), and the KSMOR* is performed with $q = 3$ and $s = 20$. The reference solution is computed with the EE method. **Left:** Proposed methods applied for $h = 0.04$ ($N = 94376$ grid points). **Right:** Proposed methods applied for $h = 0.01$ ($N = 1502501$ grid points). The FEDRK method clearly outperforms all other methods, even for finer grids.

case, the upper bound is given by $\tau_{\max,2D} \approx 26.09$ and the application of EE requires 100000 iterations, resulting in a CPU time of around 2800 seconds. The final outcome, illustrated on the right in Figure 6.20, gives exactly the same rating as before. Here as well, the FEDRK method achieves significantly higher efficiency. Apart from that, the result of KSMOR*, shows another interesting fact. For the considered grid, the selected number of snapshots and subspaces is not sufficient to converge towards the EE solution. This demonstrates that the values for q and s are related to the fineness of the grid and the definition of a fixed set of parameters for our application cannot be done directly without further investigations.

From an engineering point of view, however, a more detailed investigation of the dynamic behaviour of the underlying model shows that small values s and q are deemed appropriate. A visualisation of the results between the absolute differences of EE and KSMOR* for $h = 0.04$ and a different number of s and q is visualised in Figure 6.21. As expected, increasing the number of snapshots s or subspaces q leads to better results in terms of the maximum error and the L_2-error. An individual change of s and q leads to a kind of saturation behaviour, so that both parameters require relatively large values for an accurate approximation.

Furthermore, a reasonable number of snapshots s is significantly important because less data lead to strong artefacts at the interfaces where the matching conditions between different materials have to be fulfilled. This observation is a consequence of the proposed input matrix reduction, which clarifies the interdependence between snapshot data and the errors at the interfaces. Due to the fact that small values of s and q only influence the area around the interfaces (cf. Figure 6.21), which is generally negligible for a reasonable reproduction of the GES model, the use of KSMOR* appears to be significantly appropriate.

Based on the presented findings, we investigate the performance of FEDRK, direct CN and KSMOR* with $s = 10$ and $q = 2$ on real data.

Remark 6.10. *The same interdependence between snapshot data and errors at the interfaces as specified above can also be determined when using the KSMOR-SVD approach.*

6.5.2 Geothermal Energy Storage Simulation on Real Data

Lastly, we present a comparison concerning a full error computation along the temporal axis using the two-dimensional GES simulation including sources. The evaluation is based on real data, in particular matching of temperature probes of a test field and its thermal behaviour

Table 6.2: Thermophysical properties of the materials involved for the real-world data scenario from the present test field.

Layer	Conductivity λ (W/(m·K))	Density ρ (kg/m^3)	Specific heat c (J/(kg·K))	Diffusivity a (m^2/s)
Lower ground	0.5	1900	750	$3.51 \cdot 10^{-7}$
Upper ground	1.7	2000	1250	$6.8 \cdot 10^{-7}$
Insulation walls	0.035	40	1500	$5.83 \cdot 10^{-7}$

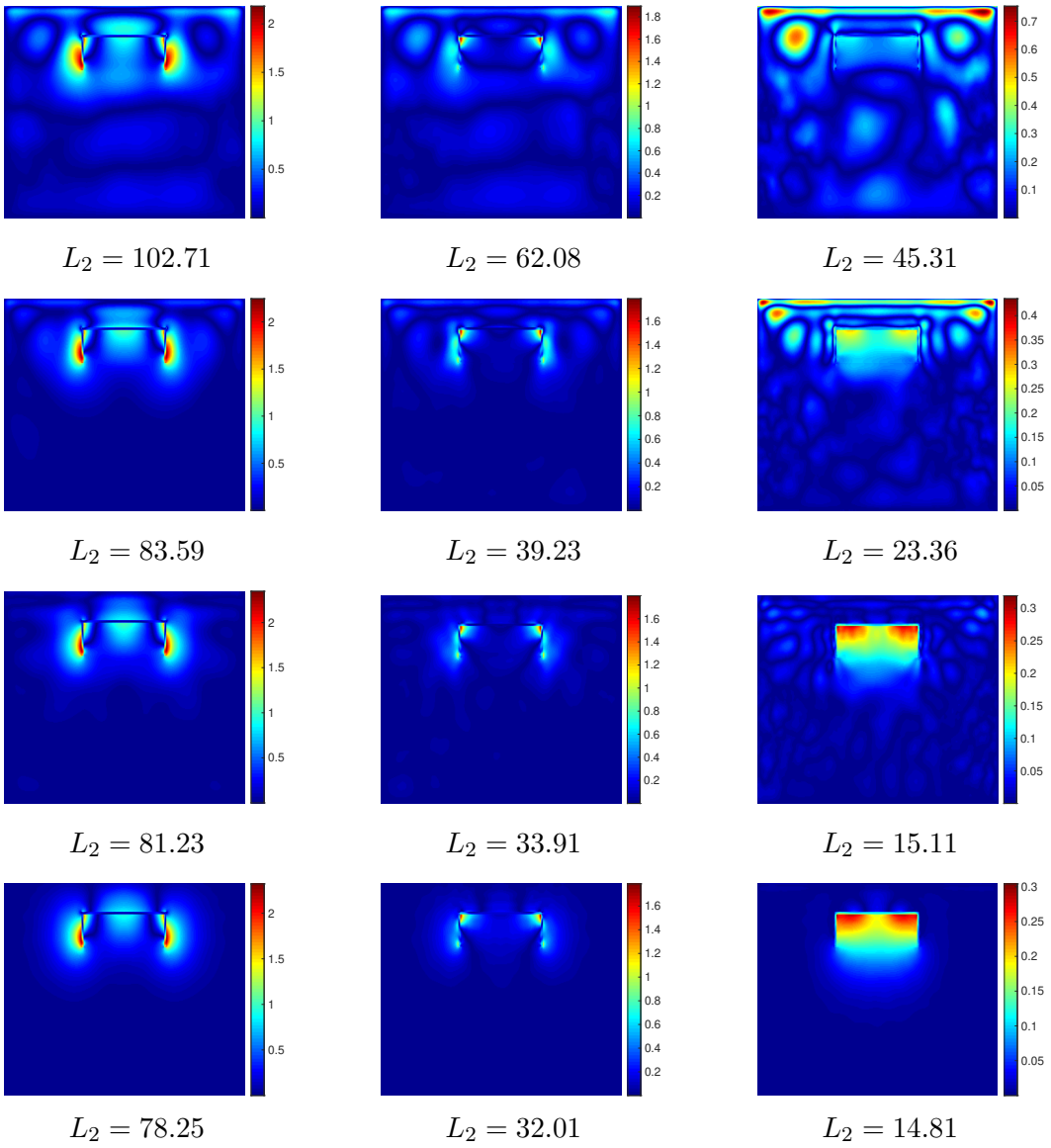


Figure 6.21: Results of the two-dimensional GES experiment without source: visual comparison (absolute differences) of the results between EE and KSMOR* for varying number of s and q at $t_F = 2.609 \cdot 10^6$ for $h = 0.04$ with $\tau = 2609$. The visualisations are ordered **from left to right** by $s = 5, 10, 20$ and **from top to bottom** via $q = 2, 3, 5, 10$. Special care must be taken in rating the visualisations, because no uniform temperature scaling between the absolute differences of the solutions is given: we still opted to represent the solutions in this way, as the differences between the errors that occur are too large to employ a uniform scaling. For a better comparison, the L_2-error is also displayed. Increasing the number of snapshots s or subspaces q, better results in terms of the maximum error and the L_2-error are obtained. Obviously, a reasonable selection of s is of great importance, since less snapshot data lead to strong artefacts at the interfaces.

given in 3D, cf. Figure 6.22. In this way, the experiment will also demonstrate that the proposed linear two-dimensional model is sufficient to reproduce the heat exchange correctly.

For this experiment, the thermophysical parameters of the materials used are assumed to be constant and are given in Table 6.2. The other parameters are fixed to $\alpha_c = 0.1$, $\alpha_A = 10$, $b_1 = 0.5$ and $b_2 = 12$. The two-dimensional rectangular domain is defined via $[0, 20] \times [0, 8]$ in meters, whereby the size of the heat tank amounts to 10×1.2 meters. The installation depth and the insulation thickness are determined to 70 and 12 centimetres, respectively. In addition, the thermal energy

$$Q = mc\Delta T \tag{6.78}$$

with the specific heat capacity c and the mass m of the fluid and temperature change of the inlet and return temperatures ΔT is given over a period of $t_F = 21 \cdot 10^6$ seconds (around 243 days). The parameters are fixed to $c = 1.16$ and $m = 850$ and the source is simply distributed over three pipe levels inside the heat tank. The initialisation shown in Figure 6.23 is computed by the Laplace interpolation, as described in Section 6.2.4, based on given temperature probes.

In order to validate the correct heat exchange behaviour, we initially apply the EE method to the two-dimensional GES model. The visualisation of the real temperature probe B1 (3rd

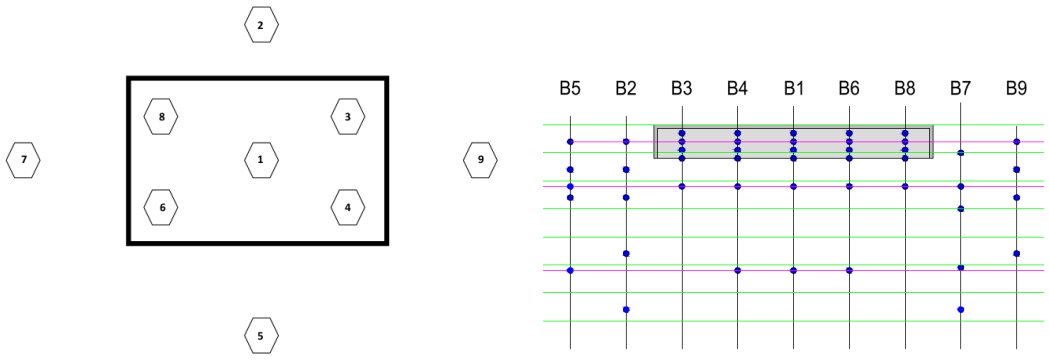


Figure 6.22: Position of the temperature probes. **Left:** Plan view of the GES and the installation location of the temperature probes. **Right:** Cross section of a 3D-GES and the installation height of temperature probes. Each probe has several measuring points.

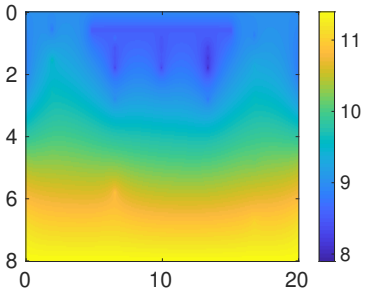


Figure 6.23: The initial heat distribution computed via Laplace interpolation using the initial temperature given from temperature probes at time $t = 0$, see Section 6.2.4. The interpolation task is performed with suitable boundary conditions.

from the top, see Figure 6.22) compared to EE is shown exemplarily on the top left in Figure 6.24. The result clearly demonstrates the appropriate use of a two-dimensional simulation of a 3D-GES model. Note that one may use any working probe to come to the same conclusion, and we show other corresponding results in this section as well.

Building on the validation, the performances of FEDRK, direct CN and KSMOR* with $q = 2$ and $s = 10$ are identified using the temperature probe B1 by two variants. First, a visual assessment of the approximations compared to real data is illustrated in Figure 6.24. In the experiment documented in Figure 6.24 we give quickly computed approximations (CPU time of 10 seconds) in comparison with EE (CPU time of 69 seconds). Second, the relation between the L_2-error (along the temporal axis) and the corresponding CPU time is evaluated in Figure 6.25. In both investigations, the FEDRK scheme is still considered as a superior method, which provides a fast computation combined with high accuracy. Nevertheless, the KSMOR* method is also very efficient. As in the previous tests the direct CN method provides the worst performance.

Finally, the FEDRK scheme and the KSMOR* method are tested on the temperature probe B6 (5th from the top) and B9 (2nd from the top), see Figure 6.26. Both approaches achieve the desired reproduction of heat distribution behaviour compared to the real data. However, the KSMOR* results may also exhibit oscillations, especially in a local area around the interfaces between different materials. This illustrates again that the numerical solutions obtained by KSMOR* are locally representative away from the interfaces if q and s are chosen in a competitive range in our application, whereas the FEDRK method correctly reproduces the global behaviour of the underlying GES model.

6.6 Summary

In this chapter we have demonstrated that a model based on a linear heat equation equipped with external and internal boundary conditions is suitable to represent the long-term behaviour of the GES in a realistic way. Furthermore, we have shown experimentally using real-world data from a three-dimensional test field that a two-dimensional GES simulation is sufficient to tackle the long-term simulation task. As a result, the computational costs can be extremely reduced and a long-term simulation is practicable.

In view of the possible candidates for an efficient numerical simulation, a state-of-the-art method that appears to be attractive is the KSMOR scheme. For our practical problem we have seen that this scheme is not easy to apply, since the underlying discretised model is linked to a large number of inputs due to the modelled boundary conditions. Independent of the numerical approach, it has also been noticed that the presence of sources and boundary conditions makes the long-term simulation issue in practice delicate to handle. With these fundamental difficulties, we illuminated in detail that it is not straightforward to device an efficient and accurate enough numerical scheme.

In total, we have demonstrated the practical usability of the FEDRK scheme and the KSMOR variant introduced here as KSMOR* which turn out to be the two most powerful methods among the schemes for our GES application. The explicit FEDRK scheme borrowed from image processing, is highly efficient due to cost-effective matrix-vector multiplications and the natural frequent update process of the input within the approach. In addition to this, it is by construction of the FEDRK method also possible to use modern parallel architectures

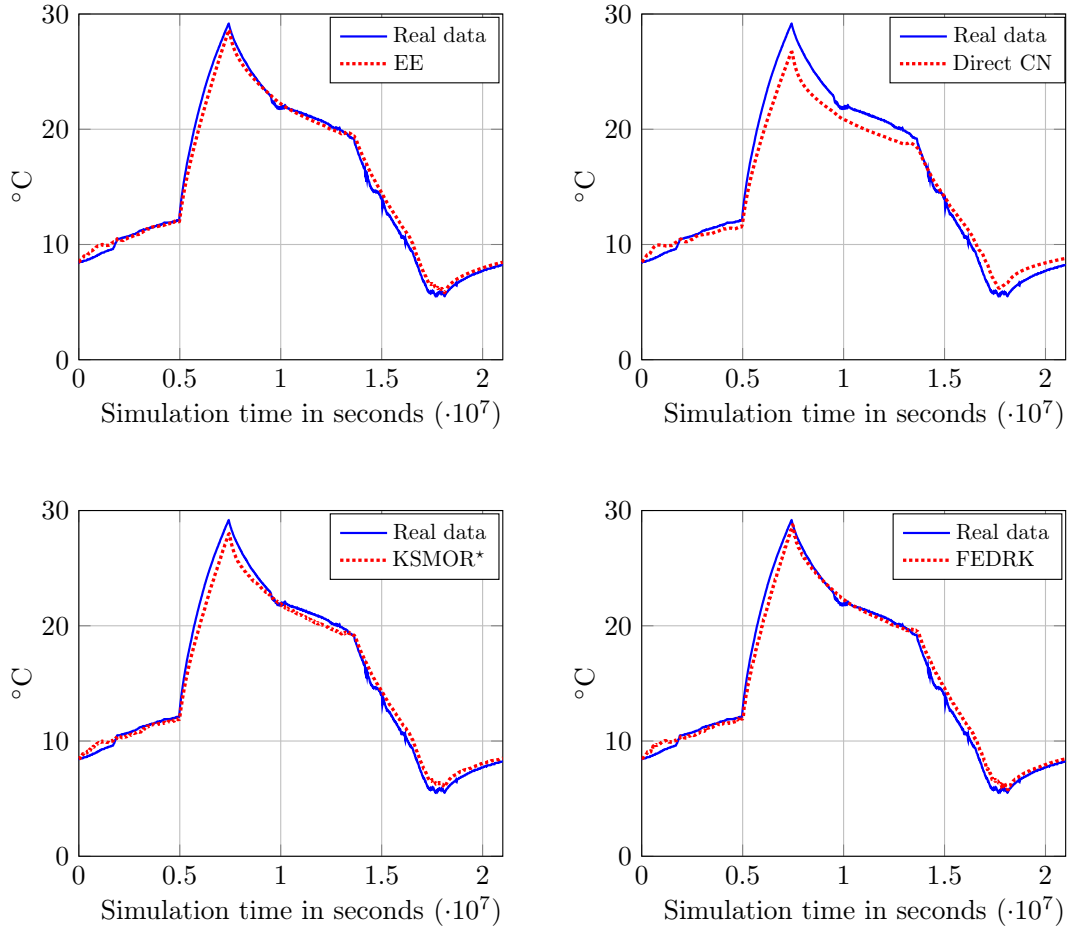


Figure 6.24: Results of the GES experiment for temperature probe B1 (3rd from the top): visual comparison (along the temporal axis) between the real temperature from 3D test field and the approximations of the 2D model computed by the proposed solvers. A special emphasis on this is that the results generated by FEDRK, direct CN and KSMOR* cause *computational costs of 10 seconds*, whereas the EE scheme produces substantial *costs of 69 seconds*. **Left top:** Result of EE after 35839 iterations for $\tau_{\text{max,2D}} = 588.23$. **Right top:** Result of direct CN for $\tau = 44854$, which leads to 470 iterations. **Left bottom:** Result of KSMOR* with $s = 10$ and $q = 2$ for $\tau = 13176$, which corresponds to 1600 iterations. **Right bottom:** Result of FEDRK for $M = 220$ cycles, which yields 4840 iterations. The EE method applied to the two-dimensional model problem reproduces a highly accurate approximation with regard to real data given from the three-dimensional test field. The numerical solvers FEDRK and KSMOR* are extremely efficient and achieve almost the same output as EE. On closer inspection, the FEDRK scheme provides the best trade-off as it is less cost-intensive due to matrix-vector multiplications and additionally produces a better accuracy based on a more frequent update process. In contrast, the direct CN method cannot maintain the performance of the other two methods.

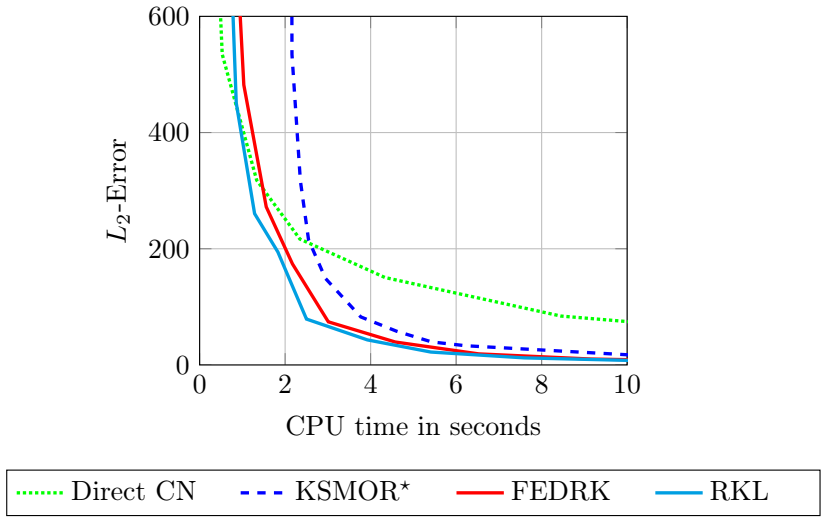


Figure 6.25: Results of the GES experiment for temperature probe B1 (3rd from the top): comparison of the L_2-error (along the temporal axis) and the corresponding CPU time between FEDRK, direct CN and KSMOR* with $q = 2$ as well as $s = 10$. The simulation time amounts to $t_F = 21 \cdot 10^6$ seconds and the grid size is fixed to $h = 0.04$ ($N = 100701$ grid points). The reference solution is computed with the EE method. The FEDRK scheme is the most efficient solver. For the sake of completeness, the RKL scheme is also tested that even provides a slightly better performance than FEDRK.

like GPUs, which can even further increase the efficiency. We would also like to point out here that to our best knowledge, this work provides the first application of FEDRK outside the field of image processing. Our proposed efficient KSMOR* technique uses an input matrix reduction via snapshots and generates a small-sized reduced order model, which can then be resolved easily using the direct solver. We illustrated and analysed the viability of KSMOR*, which is in this form a new variant of existing schemes. At this point it should be stressed that compared to previous works in the area of KSMOR schemes, we have made our method here explicit in all the details and important parameters, which is through the computational experience gained in the course of this chapter highly relevant for practical application.

Apart from the fact that FEDRK and KSMOR* are predestined solvers for tackling the long-term simulation of a GES, we also specifically discussed the local and global behaviour of their solutions. In summary, we (absolutely) suggest the use of the FEDRK scheme when the global behaviour of solutions is of interest. Let us also mention that this may be preferred in the framework of our application as this takes into account general configurations, also at interfaces, to measure temperatures with probes in realistic environments. Otherwise, both techniques perform equally effective for local areas, but generally not at interfaces.

Overall, we have precisely analysed the features of all the applied solvers and at the same time illustrated their properties using different experiments. For these reasons the comprehensive work given here provides, from our point of view, a reasonable overview of the state-of-the-art numerical solvers of various scientific areas and can be very helpful in solving similar problems in many engineering fields.

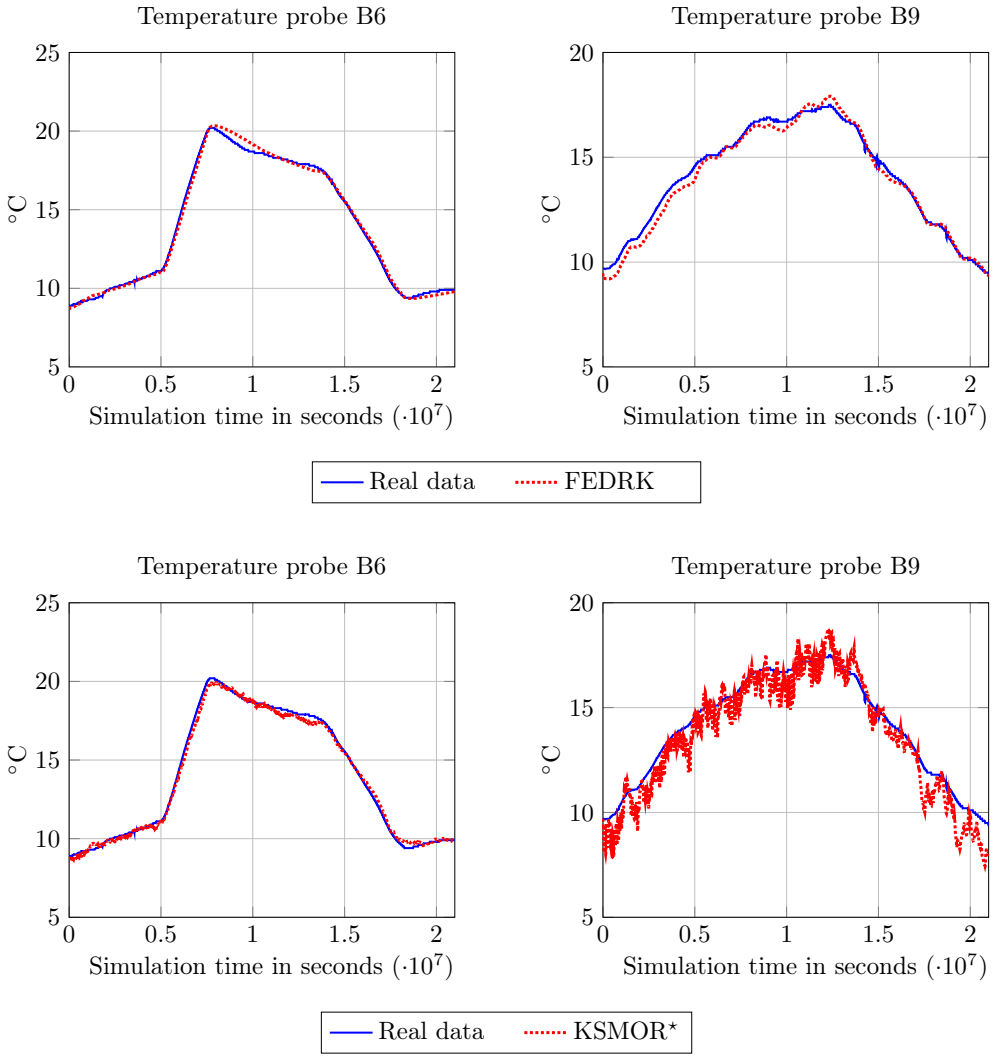


Figure 6.26: Results of the GES experiment for temperature probe B6 (5th from the top) and temperature probe B9 (2nd from the top) on the **left** and **right** side in this figure, respectively: visual comparison (along the temporal axis) between the real temperature from 3D field and the approximations of the 2D model computed by FEDRK as well as KSMOR*. The computational costs of the computed approximations of both methods amount again to 10 seconds. **Top:** Result of FEDRK for $M = 220$ cycles, which gives 4840 iterations. **Bottom:** Result of KSMOR* with $s = 10$ and $q = 2$ for $\tau = 13176$, which corresponds to 1600 iterations. Both methods reproduce the same temperature behaviour compared to the real data. However, the corresponding temperature of the KSMOR* scheme can oscillate highly sensitive, especially in a local area around the interfaces between different materials.

6.6.1 Acknowledgements

The work in this chapter was supported by *hartig & ingenieure GmbH*. This includes in particular the provision of technical information and corresponding real data of a three-dimensional GES. We also acknowledge the funding from the *AiF Projekt GmbH*.

Chapter 7

Efficient Linear Osmosis Filtering

In this chapter we are interested in an efficient computation of the PDE osmosis model based on fast explicit methods. The osmosis process, which is based on the nonsymmetric linear drift-diffusion equation, is widely used for image processing tasks and typically has to deal with large images. Hence, an efficient numerical scheme is of essential importance. However, even nowadays it is still a challenge to find a method that combines accuracy and computational efficiency, while preserving physical properties from the continuous setting also in the discrete case.

In this context, standard methods show their weaknesses in terms of inefficiency or inaccuracies, which we will demonstrate by our experiments. In particular, the IE method is being unconditionally stable and produces accurate approximations, but requires a complete LU factorisation which is extremely costly for large images. Unfortunately, iterative Krylov methods such as the BiCGSTAB method, which have to solve large sparse nonsymmetric linear systems, suffer from its poor convergence rate and thus cannot provide a desirable fast computation. In contrast, operator splitting schemes enable the handling of tridiagonal systems, which can be solved highly efficiently, whereas causing unfavourable splitting errors even for small time step sizes. The splitting errors that occur have a strong influence on the accuracy of the approximation, so that postprocessing steps are often required in order to obtain a desired image quality.

To address these issues we propose to use FEDRK and RKL for osmosis-based image processing tasks, as fast explicit methods are based solely on simple matrix-vector multiplications and are well-suited for parallel GPU processing. As a result, the osmosis approximation obtained is computed both accurately and extremely efficiently, and is a significant improvement over the state-of-the-art-methods. However, when using fast explicit methods the preservation of the natural osmotic properties and also the numerical stability cannot be guaranteed from a theoretical point of view. So far, these aspects can only be verified experimentally by our numerical tests. In addition, we will show that the fast explicit methods are well applicable for anisotropic osmosis filtering which is a useful extension of linear (isotropic) osmosis.

Within the framework of linear osmosis we analyse the use of the KSMOR method as an alternative to the IE method. The projection onto a reduced order model achieves an essential efficiency gain. Nonetheless, for large images this technique inherits worse performance due to convergence problems of the iterative BiCGSTAB method itself. Apart from that important properties such as stability and positivity preservation can only be ensured experimentally.

For the performance evaluation of the solvers, we conduct a thorough numerical study and analyse the schemes using image processing applications. The simple compatible and the more appealing quasi-compatible case are considered, in which visual computing problems such as image cloning and shadow removal with different resolutions are discussed.

The FEDRK method is well-known in image processing. Remarkably, it appears that the fast explicit methods are still underestimated, especially when nonsymmetric matrices such as for the osmosis process are considered, as in this situation the internal stability cannot be guaranteed from a theoretical point of view. Nevertheless, FEDRK and RKL are ideally suited to problems with nonsymmetric matrices provided that the hyperbolic-parabolic equation possesses dominant diffusion and the corresponding system matrix is close to a normal matrix, which is generally fulfilled for the osmosis model.

Chapter Organisation After a brief introduction of linear osmosis filtering in Section 7.1, we recall the general framework in a continuous and discrete setting in the Sections 7.2 and 7.3. As our work relies on related numerical methods, we introduce and discuss in Section 7.4 the use of numerical solvers for solving the underlying osmosis model. Afterwards, we present in Section 7.5 a detailed comparison of the solvers in terms of accuracy and efficiency using different scenarios related to image processing tasks in which the osmosis process is normally applied. Finally, the chapter ends with a summary.

7.1 Introduction

Diffusion processes are a frequent and successful tool in image processing and computer vision such as image denoising, image inpainting, centroidal Voronoi tessellation or shape analysis (cf. Chapter 5). In order to deal with more sophisticated imaging tasks where the steady state is nonflat, e.g. image editing or shadow removal, a modified version of the diffusion equation can be used. To achieve those nontrivial steady states the original PDE is extended by a drift term. The resulting drift-diffusion equation is known as linear osmosis filter and is closely related to its transport phenomenon in nature. In fact, osmosis describes the transport of liquid concentrations through a semipermeable membrane where different steady state concentrations can arise on both sides. This natural process can be transferred to image processing, where image intensities correspond to concentrations and the membrane represents the boundary between two neighbouring pixels. In other words, osmosis can be seen as the nonsymmetric counterpart to diffusion, in which the nonsymmetry arises due to the nonsymmetric permeability of osmosis in both directions. As a result, the osmosis processes enable nontrivial steady states by steering the user-defined drift term (gradient data) and thus can tackle different interesting imaging problems. Remarkably, the osmosis model remains completely linear, although the diffusion equation incorporates an additional drift term that is of particular importance from a theoretical and numerical point of view. We mention that the linearity results due to the use of time-independent drift term structure which differs from nonlinear diffusion.

Based on the fact that osmosis is basically derived from diffusion, both show similarities in their physical processes, so that essential properties also hold for the osmosis model. In particular, the resulting drift-diffusion equation is in divergence form so that the average grey value as well as the nonnegativity of the initial image is preserved. A first comprehensive description of the linear osmosis model with special attention to the continuous setting was given in [295]. In an accompanying paper [289] also the fully discrete counterpart was provided and studied.

The most important feature of the osmosis model is its evolution towards a desired steady

state. Consequently, the steady state solution of the linear osmosis process is of interest. This suggests to solve the corresponding elliptic PDE directly, as proposed in [72], which represents the osmotic steady state solution of the parabolic PDE. However, the elliptic problem has infinitely many solutions, so the solution of the parabolic-based problem is preferred that is positive and has the same average grey value as the initial data. Another significant aspect of osmosis is that the process converges towards a multiplicatively rescaled version of the initial drift term data. This implies that the osmosis model provides invariance under multiplicative illumination changes and can also adapt the contrasts of input images that differ significantly. This makes the osmosis filter to be a powerful tool for visual computing applications and overcomes those weaknesses that occur when using the well-known Poisson image processing [215]. In addition, the osmosis model is flexible and also enables an anisotropic extension [205] as has already been done in connection with diffusion filtering. The anisotropic modelling incorporates local directions depending on the image orientation and can improve the linear osmosis, in particular blurring artefacts, but can itself lead an over-smoothing effect. Recently, another application based on osmosis was presented which uses the osmosis energy term to derive a new variational model for nonlinear image fusion [204]. Within the process, the osmosis energy used is related to the osmotic steady state which is a minimiser of a quadratic energy functional.

Apart from the continuous framework, the numerical solution of the nonsymmetric semi-discrete system, especially for large images is of practical interest. As shown in [289], the EE and IE methods preserve the fundamental osmotic properties, whereby EE being subject to a time step size restriction and IE being unconditionally stable. In order to efficiently compute the osmotic steady state the IE scheme combined with the BiCGSTAB method is proposed [289]. Although BiCGSTAB is often used to solve nonsymmetric large and sparse linear systems, the solver loses efficiency due to its poor convergence rate within the osmosis framework. Therefore, efficient operator splitting methods in the form of ADI schemes have been considered [53]. Unfortunately, the Peaceman-Rachford splitting fulfils the osmotic properties only for sufficiently small time step sizes. In contrast, the Douglas splitting provides unconditional stability, whereas nonnegative off-diagonal entries cannot be guaranteed. To overcome these problems the additive and multiplicative operator splitting schemes for osmosis have been proposed [206] which are unconditionally stable and causes extremely low computational costs. Unfortunately, these splitting methods cause a natural splitting error, which means that the time step size must be relatively small in practice.

Before we discuss in more detail the numerical realisation, let us give a short description about the theoretical background of linear osmosis filtering. The classic theory of linear osmosis was introduced by Weickert *et al.* [289, 295]. In the works both the continuous and the discrete setting is studied.

7.2 Continuous Linear Osmosis Filter

Let $\Omega \subset \mathbb{R}^2$ be a bounded rectangular domain with boundary $\partial\Omega$ and $u, v, f : \Omega \to \mathbb{R}_+$ be positive greyscale images. Moreover, a *drift vector field* $\boldsymbol{d} : \Omega \to \mathbb{R}^2$ is given. Then a *linear osmosis filter* computes iteratively a version $u(\boldsymbol{x}, t)$ of $f(\boldsymbol{x})$ by solving the drift-diffusion

(advection-diffusion) equation

$$\partial_t u(\boldsymbol{x},t) = \Delta u(\boldsymbol{x},t) - \operatorname{div}\Big(\boldsymbol{d}(\boldsymbol{x})u(\boldsymbol{x},t)\Big), \quad \text{on } \Omega \times (0,t_F] \tag{7.1}$$

with initial condition

$$u(\boldsymbol{x},0) = f(\boldsymbol{x}), \quad \text{on } \Omega \tag{7.2}$$

and typical homogeneous Neumann boundary conditions, specifying a vanishing flux to the image boundary

$$\langle \nabla u(\boldsymbol{x},t) - \boldsymbol{d}(\boldsymbol{x})u(\boldsymbol{x},t), \boldsymbol{n}\rangle = 0, \quad \text{on } \partial\Omega \times (0,t_F] \tag{7.3}$$

where $\langle\cdot,\cdot\rangle$ denotes the Euclidean scalar product and $\boldsymbol{n}$ the outer normal vector.

In the continuous setting it can be shown that any solution of (7.1)-(7.3) preserves the average grey value and the positivity. Furthermore, if the vector field $\boldsymbol{d}$ in the form $\boldsymbol{d}(\boldsymbol{x}) = \nabla(\ln(v(\boldsymbol{x})))$ then the steady-state solution is a minimiser of a so-called *osmosis energy* functional. The essential theoretical properties are summarised in theorem below:

Theorem 7.1. *([295]) The solution of the linear osmosis process* (7.1)-(7.3) *with positive initial image f and drift vector field $\boldsymbol{d}$ satisfies the following properties:*

(a) The preservation of the average grey value:

$$\frac{1}{|\Omega|}\int_\Omega u(\boldsymbol{x},t)\,\mathrm{d}\boldsymbol{x} = \frac{1}{|\Omega|}\int_\Omega f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}, \quad \forall t > 0 \tag{7.4}$$

(b) The preservation of the positivity:

$$u(\boldsymbol{x},t) > 0, \quad \forall (\boldsymbol{x},t) \in \Omega \times (0,t_F] \tag{7.5}$$

(c) The convergence to a nontrivial steady state:
If $\boldsymbol{d}$ satisfies $\boldsymbol{d}(\boldsymbol{x}) = \nabla(\ln(v(\boldsymbol{x})))$ with some positive image v. The steady state equation

$$\Delta u(\boldsymbol{x},t) - \operatorname{div}\Big(\boldsymbol{d}(\boldsymbol{x})u(\boldsymbol{x},t)\Big) = 0 \tag{7.6}$$

is equivalent to the Euler-Lagrange equation of the energy functional

$$E_v(u) := \int_\Omega v(\boldsymbol{x}) \left\| \nabla\left(\frac{u(\boldsymbol{x})}{v(\boldsymbol{x})}\right)\right\|^2 \mathrm{d}\boldsymbol{x} \tag{7.7}$$

Moreover, the steady state solution is given by $w(\boldsymbol{x}) = \frac{\mu_f}{\mu_v} v(\boldsymbol{x})$, where μ_f and μ_v denote the average grey values of f and v, respectively.

The positivity preservation of osmosis filtering is a weaker property compared to pure diffusion filtering which fulfils the maximum-minimum principle. Remarkably, osmosis implies a nontrivial steady state solution, whereas diffusion converges to flat steady states. Consequently, the steady state of the linear osmosis model depends on the drift term and enables the drift vector field to be used as a model parameter to steer the process towards a desired steady state solution. This is an interesting aspect for image processing applications.

Since $\boldsymbol{d}$ contains the gradient information of $\nabla(\ln v)$, osmosis filtering can be used as process of data integration such as image editing and image fusion. In addition, $\boldsymbol{d} = \nabla(\ln v)$ is invariant under multiplicative rescaling of v. It should be mentioned that linear osmosis is not limited to the greyscale case. To deal with colour images, the osmosis process is performed using individual drift vector fields in each RGB channel separately.

The described linear drift-diffusion process enables to integrate gradient data information based on its drift term in a simple way without requiring any nonlinearities. This is advantageous from a computational and practical point of view. Nevertheless, the linear setting also has general limitations in its model solution. For instance, shadow removal applications generally require a postprocessing inpainting step in order to improve the output of linear osmosis. To overcome this issue Parisotto *et al.* [205] introduce anisotropic osmosis filtering that includes local directional structures via a modified tensor, and thus encodes local directional information. This enhanced technique has also been employed in connection with nonlinear diffusion filtering. However, anisotropic models are both theoretically and numerically more demanding. From a numerical point of view, designing a suitable scheme is not a trivial task as spatial discretisation should ensure that the underlying matrix has nonnegative off-diagonals. Fortunately, the anisotropic tensor is only computed once with respect to the initial image, so the model remains linear. We emphasise that the use of anisotropic osmosis can lead to an over-smoothing effect.

Elliptic Steady State Obviously, when using osmosis filtering one is mainly interested in the steady state solution. The osmotic steady state is given by the elliptic PDE

$$\Delta u(\boldsymbol{x}, t) - \operatorname{div}\Big(\boldsymbol{d}(\boldsymbol{x}) u(\boldsymbol{x}, t)\Big) = 0 \tag{7.8}$$

Thus, it appears to solve (7.8) directly in connection with homogeneous Neumann boundary conditions by using numerical solvers for sparse linear systems. Compared to the time-dependent parabolic equation (7.1), solving only one sparse linear system leads to lower computational costs and, moreover, no setting of a stopping time is required. In this framework, the authors [72] suggests to replace the global elliptic model by a local model in which the data locality represents a region of interest. This local model can be solved highly efficiently and leaves the region outside of the local part unaltered. This is beneficial, but may also lead to some saturation effects.

However, it should be noted that the elliptic problem has infinitely many solutions, meaning for any solution $w(\boldsymbol{x})$, also $cw(\boldsymbol{x})$ with some arbitrary constant c is a solution. Consequently, the main information of a given v such as the average grey value gets lost. This suggests to compute the steady state by solving the parabolic equation (7.1).

7.3 Discrete Linear Osmosis Filter

Building on the continuous model the fully discrete theory for osmosis filtering was studied in [289]. For the discrete case we adopt the spatial finite-difference discretisation proposed by Weickert *et al.* [295], and consider a discrete rectangular image domain Ω of size $n \times m$. Moreover, for a given uniform grid size $h > 0$, we denote by $u_{i,j}$ the approximated function value of u at grid point $\boldsymbol{x}_{i,j} = (x_i, y_j) = ((i - \frac{1}{2})h, (j - \frac{1}{2})h))$, where $i = 1, \ldots, n$, $j = 1, \ldots, m$.

The numerical approximation of the given drift-diffusion equation

$$\partial_t u = \Delta u - \operatorname{div}(\boldsymbol{d}u) = \partial_{xx}u + \partial_{yy}u - \Big(\partial_x(\boldsymbol{d}u) + \partial_y(\boldsymbol{d}u)\Big) \tag{7.9}$$

using standard finite difference spatial discretisation and setting $\boldsymbol{d} = (d_1, d_2)^\top$ yields

$$\begin{aligned} u'_{i,j} &= \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} - \frac{1}{h}\Big[(\boldsymbol{d}u)_{i+\frac{1}{2},j} - (\boldsymbol{d}u)_{i-\frac{1}{2},j}\Big] \\ &\quad + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} - \frac{1}{h}\Big[(\boldsymbol{d}u)_{i,j+\frac{1}{2}} - (\boldsymbol{d}u)_{i,j-\frac{1}{2}}\Big] \\ &= \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} - \frac{1}{h}\left(d_{1,i+\frac{1}{2},j}\frac{u_{i+1,j} + u_{i,j}}{2} - d_{1,i-\frac{1}{2},j}\frac{u_{i,j} + u_{i-1,j}}{2}\right) \\ &\quad + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2} - \frac{1}{h}\left(d_{2,i,j+\frac{1}{2}}\frac{u_{i,j+1} + u_{i,j}}{2} - d_{2,i,j-\frac{1}{2}}\frac{u_{i,j} + u_{i,j-1}}{2}\right) \end{aligned} \tag{7.10}$$

The approximation of the drift vector field $(d_1[v], d_2[v])^\top = \frac{\nabla v}{v}$ for some positive image at intermediate grid points is defined using upwind discretisation via

$$d_{1,i+\frac{1}{2},j} = \frac{2(v_{i+1,j} - v_{i,j})}{h(v_{i+1,j} + v_{i,j})}, \quad d_{2,i,j+\frac{1}{2}} = \frac{2(v_{i,j+1} - v_{i,j})}{h(v_{i,j+1} + v_{i,j})} \tag{7.11}$$

and fulfil

$$|d_1(\boldsymbol{x})| < \frac{2}{h}, \quad |d_2(\boldsymbol{x})| < \frac{2}{h}, \quad \forall \boldsymbol{x} \in \Omega \tag{7.12}$$

The latter equations identify the osmosis process (semi-permeable membrane between neighbouring pixels) and describe the transport of particles, here image intensities, depending on the orientation. Notably, the model remains linear after introducing the drift term. Incorporating the still missing homogeneous Neumann boundary conditions correspond to mirror the image at its boundaries and setting zero drift vector across the boundaries. Rearranging of (7.10) subsequently gives

$$\begin{aligned} u'_{i,j} &= u_{i+1,j}\left(\frac{1}{h^2} - \frac{d_{1,i+\frac{1}{2},j}}{2h}\right) + u_{i-1,j}\left(\frac{1}{h^2} + \frac{d_{1,i-\frac{1}{2},j}}{2h}\right) \\ &\quad + u_{i,j+1}\left(\frac{1}{h^2} - \frac{d_{2,i,j+\frac{1}{2}}}{2h}\right) + u_{i,j-1}\left(\frac{1}{h^2} + \frac{d_{2,i,j-\frac{1}{2}}}{2h}\right) \\ &\quad + u_{i,j}\left(-\frac{4}{h^2} - \frac{d_{1,i+\frac{1}{2},j}}{2h} + \frac{d_{1,i-\frac{1}{2},j}}{2h} - \frac{d_{2,i,j+\frac{1}{2}}}{2h} + \frac{d_{2,i,j-\frac{1}{2}}}{2h}\right) \end{aligned} \tag{7.13}$$

If using now an N-dimensional vector $\boldsymbol{u}(t) \in \mathbb{R}^N$ that represents the unknown values (replace double by single indexing, $N = nm$ denotes the number of pixels), we end up with the following semi-discrete system:

$$\dot{\boldsymbol{u}}(t) = A\boldsymbol{u}(t), \quad t \in (0, t_F],\ \boldsymbol{u}(0) = \boldsymbol{f} \tag{7.14}$$

where the matrix $A \in \mathbb{R}^{N\times N}$ is large, sparse and nonsymmetric based on the additional drift term. This differs from its counterpart, diffusion filtering, which leads to symmetric system matrices. Using the defined drift vector discretisation (7.11) ensures that the weights of the neighbours of $u_{i,j}$ within the scheme (7.13) are positive and implies that the off-diagonals of A are nonnegative.

As shown above, the continuous linear osmosis model preserves both mass and nonnegativity and, in addition, creates a nontrivial steady state. These properties should also be valid in the discrete case, which is a particularly important issue for the corresponding fully discrete theory. As shown in [289], the osmotic properties can be achieved in complete analogy to the continuous case built on Weickert's discrete framework for diffusion filters [293] and can be seen as the nonsymmetric counterpart.

Theorem 7.2 ([289]). *For a given $\boldsymbol{f} \in \mathbb{R}^N_+$, consider the fully discretised problem*

$$\boldsymbol{u}^{k+1} = P\boldsymbol{u}^k, \quad \boldsymbol{u}^0 = \boldsymbol{f}, \quad k \geq 0 \tag{7.15}$$

where the nonsymmetric matrix $P \in \mathbb{R}^{N\times N}$ is assumed to be irreducible, nonnegative with strictly positive diagonal entries and with unit column sum, then the following properties hold:

(a) The preservation of the average grey value:

$$\frac{1}{N}\sum_{i=1}^{N} u_i^k = \frac{1}{N}\sum_{i=1}^{N} f_i, \quad \forall k > 0 \tag{7.16}$$

(b) The evolution preserves positivity:

$$u_i^k > 0, \quad \forall i \in \{1,\dots,N\},\ \forall k > 0 \tag{7.17}$$

(c) There exists a unique steady state for $k \to \infty$ which is given by the eigenvector of P associated with the eigenvalue 1.

The crucial point of the latter theorem lies in its generalisation. In fact, the important properties for the discrete linear osmosis are ensured if the numerical scheme used fulfils the four requirements of the corresponding matrix P, namely that all column sums of P are one, P is nonnegative, P is irreducible and P has only positive diagonal entries. Furthermore, the theorem is not based on a specific spatial discretisation.

7.4 Numerical Methods

In what follows we discuss the numerical realisation of the semi-discrete system (7.14) with nonsymmetric system matrix A. As shown in [289] the EE and IE method fit this discrete framework and, in particular, the implicit method remains unconditionally stable. In order to improve the efficiency for large images, Parisotto *et al.* proposed to use operator splitting methods as presented in [53, 206]. However, these numerical solvers suffer strongly from efficiency or accuracy problems, and thus we will introduce the use of fast explicit methods as well as MOR methods based on KSMOR.

Discrete Time Integration Solving the arising ODE system (7.14) involves a temporal discretisation and can be done with the time integration methods. According to Section 2.1, the EE method is given by

$$\boldsymbol{u}^{k+1} = (I + \tau A)\boldsymbol{u}^k \tag{7.18}$$

and analogously the IE method

$$(I - \tau A)\boldsymbol{u}^{k+1} = \boldsymbol{u}^k \tag{7.19}$$

where $\tau > 0$ is the time step size and index k denotes the approximation of $\boldsymbol{u}^k = \boldsymbol{u}(k\tau)$ at time $k\tau$. Assuming that the matrix $(I - \tau A)$ is invertible, both schemes (7.18) and (7.19) can be expressed as

$$\boldsymbol{u}^{k+1} = P\boldsymbol{u}^k \tag{7.20}$$

with $P := I + \tau A$ and $P := (I - \tau A)^{-1}$ for the explicit and the implicit case, respectively. The authors in [289] have shown that the EE and IE method fulfil the requirements of the corresponding matrix P, which is summarised by the theorem as follows:

Theorem 7.3 ([289]). *For a given $\boldsymbol{f} \in \mathbb{R}^N_+$, consider the semi-discretised problem* (7.14) *where the nonsymmetric matrix $A \in \mathbb{R}^{N\times N}$ is assumed to be irreducible, with nonnegative off-diagonal entries and zero-column sum. Then the following results hold:*

(a) The EE scheme

$$\boldsymbol{u}^{k+1} = (I + \tau A)\boldsymbol{u}^k \tag{7.21}$$

satisfies the requirements for the discrete linear osmosis processes provided that

$$\tau < \frac{h^2}{|a_{i,i}|}, \quad \forall i \in \{1, \dots, N\} \tag{7.22}$$

(b) The IE scheme

$$(I - \tau A)\boldsymbol{u}^{k+1} = \boldsymbol{u}^k \tag{7.23}$$

satisfies the requirements for the discrete linear osmosis processes for all time step sizes $\tau > 0$.

For both schemes, P is an irreducible, nonnegative matrix with strictly positive diagonal entries and unit column sum so that the Theorem 7.2 holds true.

In image processing applications the grid size is fixed to $h = 1$. Thus, based on the condition (7.12), the EE scheme preserves the positivity, the average grey value and converges to its unique steady state for $\tau < \frac{1}{8}$. More precisely, the latter condition on τ can be seen as a *natural stability criterion* for which the discrete model preserves the fundamental physical properties of the continuous osmosis model. We mention that this condition should not be confused with numerical stability. Otherwise, the IE method is unconditionally stable (or unconditionally positive), since any arbitrarily large time step sizes τ are allowed. Apart from that, the second order CN method

$$\left(I - \tfrac{\tau}{2}A\right)\boldsymbol{u}^{k+1} = \left(I + \tfrac{\tau}{2}A\right)\boldsymbol{u}^k \tag{7.24}$$

can also be used, but it can be shown that the osmotic properties are only fulfilled for sufficiently small time step sizes.

Lemma 7.1. *For a given $\boldsymbol{f} \in \mathbb{R}^N_+$, consider the semi-discretised problem* (7.14) *where the nonsymmetric matrix $A \in \mathbb{R}^{N\times N}$ is assumed to be irreducible, with nonnegative off-diagonal entries and zero-column sum. Then the CN scheme* (7.24) *preserves the average grey value, the positivity and converges to a unique steady state for $\tau < 2(\max|a_{i,i}|)^{-1},\ i = 1,\dots,N$.*

Proof. The CN scheme take the form $\boldsymbol{u}^{k+1} = P\boldsymbol{u}^k$ via

$$\boldsymbol{u}^{k+1} = \left(I - \tfrac{\tau}{2}A\right)^{-1}\left(I + \tfrac{\tau}{2}A\right)\boldsymbol{u}^k \tag{7.25}$$

where $P = P^-P^+$ with $P^- = (I - \frac{\tau}{2}A)^{-1}$ and $P^+ = (I + \frac{\tau}{2}A)$. Analogous to Theorem 7.3, the matrix P^- is strictly column diagonally dominant with nonpositive off-diagonal entries. Consequently, P^- is a nonsingular M-matrix and it holds that its inverse has only strictly positive entries. Moreover, P^- has unit column sums and the same holds true for its inverse. Otherwise, the matrix P^+ has unit column sum based on the zero column sums of A and is only nonnegative for $\tau < 2(\max|a_{i,i}|)^{-1}$. As a result, the matrix multiplication P^-P^+ is ensured to be nonnegative for $\tau < 2(\max|a_{i,i}|)^{-1}$. Finally, one can show that the unit column sum is retained after the matrix multiplication P^-P^+. □

Based on these observations, the IE method is preferred to solve the semi-discrete linear osmosis problem (7.14) due to its unconditional stability. However, when dealing with large images the direct solution of sparse linear systems becomes computationally intensive. Also note that for colour images sparse linear systems have to be solved separately for each colour channel. Therefore, the use of efficient sparse iterative solvers such as Krylov subspace methods is required. Since the system matrix is nonsymmetric, CG and its preconditioned variants IC and MIC are not useful. One of the most efficient and popular Krylov solver for handling nonsymmetric matrices is the BiCGSTAB method cf. [178, 237]. The BiCGSTAB method was also used for the practical performance in [289].

7.4.1 Operator Splitting Schemes

The Krylov subspace solvers such as the BiCGSTAB method can cause high computational costs for large images. To tackle this issue Parisotto *et al.* proposed the use of *operator splitting schemes* to efficiently compute the numerical solution of the osmosis model [53, 206]. The dimensional splitting strategy implies the solution of only simple tridiagonal systems for which efficient matrix factorisation techniques can be used. Unfortunately, this method class is also confronted with problems, for instance natural stability restrictions or splitting errors. Let us give a brief overview, we follow the work of Parisotto [203].

Basic Idea The splitting methods were developed simultaneously in the Soviet Union and the USA around 1960s. Operator splitting has been well discussed and is a widespread technique to solve complex problems. The class of operator splitting includes a variety of approaches and often the same method is denoted by different names due to its wide applicability. For a general overview of splitting methods we refer to [138].

The idea behind the technique is that a complicated semi-discretised operator is split into simpler operators, which are formally written as a sum. This means, the problem is split into a set of simpler tasks called *sub-problems*, where each of them is of a type for which there are simpler and more efficient solvers. Finally, an appropriate numerical scheme for each sub-model is required, and the solutions of the schemes are combined to form a solution of the original problem.

The splitting approach has several advantages, for example, the numerical method is generally more efficient, reduce memory requirements, can increase the stability range, and even provide methods that are unconditionally stable. In contrast, the splitting implementation of the operators can become cumbersome for complex model problems. Even more problematic is that the operator splitting introduce a splitting error, which can be large and is related to the selected time step size. For reducing the errors originating from these splitting mechanisms, the time step size must normally be reduced. Overall, operator splitting schemes can be a powerful tool for efficiently solving initial value problems, but their approximation quality depends on the underlying model problem.

Alternating Direction Implicit In order to reduce the computational costs for solving the semi-discretised problem (7.14), a splitting of the operator A into the sum $A_1 + A_2$ along the space direction x and y is performed. The discretisation from (7.10) can be split by $A\boldsymbol{u} = A_1\boldsymbol{u} + A_2\boldsymbol{u}$ into two one-dimensional problems

$$\begin{aligned}\left(A_1(\boldsymbol{u})\right)_{i,j} := u_{i,j}&\left(-\frac{2}{h^2} - \frac{d_{1,i+\frac{1}{2},j}}{2h} + \frac{d_{1,i-\frac{1}{2},j}}{2h}\right)\\ &+ u_{i+1,j}\left(\frac{1}{h^2} - \frac{d_{1,i+\frac{1}{2},j}}{2h}\right) + u_{i-1,j}\left(\frac{1}{h^2} + \frac{d_{1,i-\frac{1}{2},j}}{2h}\right)\end{aligned} \tag{7.26}$$

and

$$\begin{aligned}\left(A_2(\boldsymbol{u})\right)_{i,j} := u_{i,j}&\left(-\frac{2}{h^2} - \frac{d_{2,i,j+\frac{1}{2}}}{2h} + \frac{d_{2,i,j-\frac{1}{2}}}{2h}\right)\\ &+ u_{i,j+1}\left(\frac{1}{h^2} - \frac{d_{2,i,j+\frac{1}{2}}}{2h}\right) + u_{i,j-1}\left(\frac{1}{h^2} + \frac{d_{2,i,j-\frac{1}{2}}}{2h}\right)\end{aligned} \tag{7.27}$$

where A_1, A_2 are now tridiagonal operators. Again the grid size is fixed to $h = 1$. On this basis, two ADI methods can then be formed which are in general unconditionally stable.

The Peaceman-Rachford (PR) scheme [211]: For every $k \geq 0$ and time step size $\tau > 0$, the second order accurate PR scheme computes an approximation $\boldsymbol{u}^{k+1}$ via

$$\begin{aligned}\boldsymbol{u}^{k+\frac{1}{2}} &= \boldsymbol{u}^k + \tfrac{\tau}{2}A_1\boldsymbol{u}^k + \tfrac{\tau}{2}A_2\boldsymbol{u}^{k+\frac{1}{2}}\\ \boldsymbol{u}^{k+1} &= \boldsymbol{u}^{k+\frac{1}{2}} + \tfrac{\tau}{2}A_1\boldsymbol{u}^{k+1} + \tfrac{\tau}{2}A_2\boldsymbol{u}^{k+\frac{1}{2}}\end{aligned} \tag{7.28}$$

In an analogous manner to the CN method a forward and backward Euler step is applied. The PR scheme ensures the osmotic properties only for sufficiently small time step sizes.

Lemma 7.2 ([53]). *Let* $\boldsymbol{f} \in \mathbb{R}^N_+$ *and* $\tau < 2(\max\{\max|a^1_{i,i}|, \max|a^2_{i,i}|\})^{-1},\ i = 1, \ldots, N$, *where* $a^1_{i,i}$ *and* $a^2_{i,i}$ *are the corresponding diagonal elements of* A_1 *and* A_2, *respectively. Then the PR scheme* (7.28) *based on the splitting* (7.26)-(7.27) *preserves the average grey value, the positivity and converges to a unique steady state.*

The weighted-Douglas (WD) scheme [80]: For every $k \geq 0$, time step size $\tau > 0$ and $\theta \in [0, 1]$, the approximation $\boldsymbol{u}^{k+1}$ is computed using the rule

$$\begin{aligned} \boldsymbol{y}^0 &= \boldsymbol{u}^k + \tau A_1 \boldsymbol{u}^k + \tau A_2 \boldsymbol{u}^k \\ \boldsymbol{y}^j &= \boldsymbol{y}^{j-1} + \theta\tau \left(A_j \boldsymbol{y}^j - A_j \boldsymbol{u}^k \right), \quad j = 1, 2 \\ \boldsymbol{u}^{k+1} &= \boldsymbol{y}^2 \end{aligned} \tag{7.29}$$

The latter scheme computes, at each time level, firstly a forward predictor and secondly a stabilisation based on the specified splitting, where the weight θ influences the implicit or explicit behaviour of these intermediate steps. The consistency order of the scheme is two for the weight $\theta = \frac{1}{2}$, otherwise of order one. For the WD scheme one can show that the following lemma holds:

Lemma 7.3 ([53]). *Let* $\boldsymbol{f} \in \mathbb{R}^N_+, \tau > 0$ *and* $\theta \in [0, 1]$. *Then, the WD scheme* (7.29) *based on the splitting* (7.26)-(7.27) *preserves the average grey value.*

We stress that for the WD scheme exists no guarantee that the off-diagonal entries are nonnegative, so the requirements of the Theorem 7.2 are not satisfied. Overall, due to the presence of explicit steps within the abovementioned two ADI schemes, both methods suffer from natural stability requirements on τ as already seen when considering the CN method.

Additive and Multiplicative Operator Splitting Another class of operator splitting schemes is the so-called *additive operator splitting (AOS)* and *multiplicative operator splitting (MOS)* which are widely used within image processing applications such as nonlinear diffusion models, proposed in [21, 296].

Let us exemplarily show how the AOS schema is derived. The IE method reads as

$$\boldsymbol{u}^{k+1} = (I - \tau A)^{-1} \boldsymbol{u}^k \tag{7.30}$$

Using the operator splitting $A = A_1 + A_2$ one can rewrite the latter equation as

$$\begin{aligned} \boldsymbol{u}^{k+1} &= (I - \tau (A_1 + A_2))^{-1} \boldsymbol{u}^k \\ &= \left(\sum_{l=1}^{2} \tfrac{1}{2} I - \tau \sum_{l=1}^{2} A_l \right)^{-1} \boldsymbol{u}^k = \left(\sum_{l=1}^{2} \tfrac{1}{2} (I - 2\tau A_l) \right)^{-1} \boldsymbol{u}^k \end{aligned} \tag{7.31}$$

To extract the sum from the power operator, one can show

$$\left(\sum_{l=1}^{2} \tfrac{1}{2} (I - 2\tau A_l) \right)^{-1} \approx \frac{1}{4} \sum_{l=1}^{2} \left(\tfrac{1}{2} (I - 2\tau A_l) \right)^{-1} \tag{7.32}$$

so that the AOS scheme is finally obtained by

$$\boldsymbol{u}^{k+1} = \frac{1}{2}\sum_{l=1}^{2} (I - 2\tau A_l)^{-1}\, \boldsymbol{u}^k \tag{7.33}$$

The tridiagonal matrix structure of $(I - 2\tau A_l)^{-1}$ for $l = 1, 2$ can be solved very efficiently by using *tridiagonal* LU factorisation. A second advantage of AOS is that both operators are applied independently, which makes it well-suited for parallel processing [297].

In contrast the MOS scheme is based on the multiplication of the operators in the form of

$$\boldsymbol{u}^{k+1} = \prod_{l=1}^{2} (I - \tau A_l)^{-1}\, \boldsymbol{u}^k \tag{7.34}$$

Furthermore, Parisotto proposed to use a more accurate *additive-multiplicative operator splitting (AMOS)* scheme as a combination of AOS and MOS, where $\boldsymbol{u}^{k+1}$ is computed via

$$\boldsymbol{u}^{k+1} = \frac{1}{2}\sum_{l=1}^{2} \left((I - \tau A_{j_l})^{-1} (I - \tau A_{i_l})^{-1} \right) \boldsymbol{u}^k, \quad i = \{1, 2\}, j = \{2, 1\} \tag{7.35}$$

The schemes AOS, MOS and AMOS are first order accurate in time. Apart from the splitting errors that occur, the advantageous use of all three methods is their unconditional stability coupled with their highly efficient factorisation.

Lemma 7.4 ([206])**.** *Let $\boldsymbol{f} \in \mathbb{R}_+^N$, the schemes AOS, MOS and AMOS in the form of* (7.33)*,* (7.34) *and* (7.35)*, respectively, based on the splitting* (7.26)-(7.27) *preserves the average grey value, the positivity and converges to a unique steady state for any $\tau > 0$.*

7.4.2 Fast Explicit Methods

As already seen, the EE method is restricted by a rather small upper bound that is given by the natural stability condition (7.22). This directly implies the numerical stability for $\tau < (\max|a_{i,i}|)^{-1}$, since

$$\|I + \tau A\|_1 \leq 1 \tag{7.36}$$

ensures the sufficient stability condition (2.61). We emphasise that the numerical stability is also satisfied for the time step size $\tau \leq \frac{1}{4}$ due to the fact

$$\left\| (I + \tau A)^{k+1} \right\| \leq C \tag{7.37}$$

with a moderate constant C, then the matrix $(I + \tau A)^{k+1}$ approaches a constant matrix as $k \to \infty$. Obviously, osmosis does not allow to give stability results in terms of decreasing L_p-norms for $p > 1$, as osmosis that starts from a flat initial and converges to a nonflat one represents a counterexample. However, the larger upper stability bound, i.e. $\tau \leq \frac{1}{4}$, is accompanied by destroying the positivity properties which we want to maintain when time discretisation is performed.

In order to overcome the inefficiency of the EE method, but also to avoid splitting errors when using operator splitting schemes, we propose the use of fast explicit methods as presented

in Chapter 3. Although the underlying matrix is not symmetric and thus the internal stability of this class of methods is not guaranteed, FEDRK and RKL are very well applicable to such problems in practice, provided that the spectrum firstly is located in a narrow strip around the negative axis of the complex plane and secondly the corresponding system matrix is close to a normal matrix. These two requirements are usually fulfilled in connection with the linear osmosis model. Up to now, this cannot be proven from a theoretical point of view and is left for future research. Fortunately, all of our numerical tests suggest that both properties remain valid. With regard to defining when a matrix is close to a normal, meaning that $\rho(A) \approx \|A\|_2$ is satisfied.

The application of FEDRK and RKL to the osmosis model is easily performed as in Chapter 3 extensively discussed. It should be remembered that the fast explicit methods may violate the L_∞-stability as shown in Example 3.2. In consequence, the positivity property

$$u_i^k > 0, \quad \forall i \in \{1, \dots, N\}, \ \forall k > 0 \tag{7.38}$$

cannot be ensured. In fact, the desired steady state solution with a large stopping time t_F, which is steered by the drift term, is mainly important for the osmosis evolution process. In our experiments, it holds that $u_i^{k_0} > 0$ for $k_0 > k > 0$ for large t_F so that the positivity is preserved from a numerical point of view. Nevertheless, it is not possible to exclude the existence of pathological cases where the positivity may violate even for large stopping times. This issue remains left for future research.

7.4.3 Krylov Subspace Model Order Reduction

In the case of large images, the numerical realisation via the IE method becomes very expensive as a consequence of computing a complete LU factorisation. To tackle this issue in general, efficient iterative solvers such as the BiCGSTAB method are used to solve the resulting nonsymmetric large and sparse linear systems. An alternative approach is the use of MOR methods which generate a reduced order model in a preprocess, as described in Chapter 4. The application of MCR, BT and POD can be excluded on grounds of efficiency in advance, since performing an eigendecomposition or solving Lyapunov equations is required. Consequently, we discuss the KSMOR method, as introduced in Section 4.5, for solving the semi-discretised problem (7.14).

Let us first note that the linear osmosis model is obviously stable because all eigenvalues have negative real part. This follows directly from the matrix measure property (4.32), so that $\alpha(A) \leq \mu_1(A) = 0$ holds due to the zero column sum of A. However, the stability of the reduced system using a one-sided projection method cannot be guaranteed a priori, since Theorem 4.1 is only valid for the Euclidean norm. Despite the nonguaranteed stability, the reduced system remains stable, which we could only identify experimentally by our tests.

For approximate the transfer function via moment matching, we construct the orthogonal basis $V \in \mathbb{R}^{N \times r}$ using the one-sided Arnoldi approach with $W = V$ by

$$V = \operatorname{span}\left((A - \sigma I)^{-1} \boldsymbol{u}^0, \dots, (A - \sigma I)^{-r} \boldsymbol{u}^0\right) \tag{7.39}$$

The latter means that the projection matrix is constructed using the zero-input response of the system. In other words, there is no input variable within the osmosis model and therefore

no coordinate transformation is applied to obtain zero initial condition. The drift-diffusion equation is characterised by a rather slow dynamic so that the still open parameter σ is determined at $\sigma \approx 0$. In particular, for $\sigma = 0$ the inverse $(A-\sigma I)^{-1}$ does not exist, since $\lambda = 0$ is an eigenvalue of A, which results from the homogeneous Neumann boundary conditions within the modelling.

Using the projection matrix V the reduced system of order r yields

$$\dot{\boldsymbol{u}}_r(t) = A_r \boldsymbol{u}_r(t), \quad t \in (0, t_F],\ \boldsymbol{u}_r(0) = V^\top \boldsymbol{u}(0) \tag{7.40}$$

with $A_r = V^\top A V \in \mathbb{R}^{r\times r}$, which is finally solved by the IE method. It should be noted that although the reduced model (7.40) experimentally preserves stability, this does not necessarily imply that the positivity is also ensured. In any case, our experiments have shown that for small values σ the positivity property holds.

7.5 Numerical Experiments

In this section we discuss the accuracy and efficiency of the numerical solvers for linear osmosis filtering for small and medium colour image sizes at two different cases, namely the *compatible* and *quasi-compatible* case. In doing so, visual computing problems are considered such as seamless image cloning and shadow removal as also were used for practical purposes in [295]. In addition, the efficiency of the solvers concerning a large colour image is evaluated. In this setting, we also analyse the use of the KSMOR method.

Computational Aspects All experiments were done in MATLAB R2018b with an Intel Xeon(R) CPU E5-2609 v3 CPU. The splitting schemes AOS, MOS, AMOS, PR and WD incorporate tridiagonal matrices based on the splitting matrices A_1 and A_2. For this reason, the highly efficient tridiagonal LU factorisation by the internal MATLAB function *lu* within the computation is employed. In contrast, for IE and CN a complete LU factorisation is performed by the internal MATLAB function *decomposition* which leads to higher computational costs. Using the SuiteSparse package gives the same performance. The source code implementations of the schemes AOS, MOS, AMOS, PR and WD can be found in [203].

The computations were taken with the *Parallel Computing Toolbox* integrated in MATLAB. As mentioned, we consider colour images such that the osmosis process can be computed separately for each colour channel. Consequently, this step can be parallelised using the MATLAB *parfor* loop to distribute the code to 3 workers. We only do this for the implicit methods AOS, MOS, AMOS, PR, WD, IE and CN. Note that AOS and AMOS may be accelerated by using parallel code, this is not done here.

For a fair comparison we compute the explicit schemes EE, FEDRK, RKL and second order RKL (denoted as RKL2) by parallelisation on GPUs using NVIDIA GeForce GTX 690, whereby MATLAB *gpuArray* is used which additionally improves the computational efficiency. Let us stress that for the explicit methods, the GPU computation is more efficient than using CPU coupled with *parfor* distribution.

Moreover, $\tau = 0.12$ is chosen as the upper bound for both the EE method and the fast explicit methods, which obviously satisfies the condition (7.22). All CPU times presented incorporate the modelling (linear system, factorisation, parallelisation) and the numerical computation. The performances are therefore easily comparable.

7.5.1 Compatible Case

The *compatible* case is related to the situation where the drift vector field $\boldsymbol{d}$ is defined as a given positive image as $\boldsymbol{d} = \nabla(\ln v)$ over all pixels of the image domain. In this setting the osmosis process convergence to a rescaled version of v. Obviously, the convergence to a rescaled version of an image v, that is already known, is unappealing from a practical point of view, but the compatible case is reasonable for a first assessment of the solvers.

For the evaluation the well-known *mandrill* test image of size 256×256 is applied. Here $v = (v_1, v_2, v_3)$ represents the RGB image and the drift vectors in colour channel i are fixed chosen as $\nabla(\ln v_i)$. Finally, starting from a flat initial image in which each colour channel has the same mean value as the *mandrill* image, the osmosis model (7.1)-(7.3) evolves towards a known steady-state. The convergence behaviour is illustrated in Figure 7.1 using the EE method with $\tau = 0.12$ and stopping time $t_F = 100000$.

Results on First Order Accurate Schemes The performances with regard to the MSE and the corresponding CPU time of the first order accurate schemes AOS, MOS, AMOS, WD with $\theta = 1$, IE, FEDRK and RKL using varying time step sizes τ or cycles M with stopping time $t_F = 100000$ are illustrated on the top in Figure 7.2. It can be seen that all computed approximations converge towards the EE solution, with the exception of the solvers MOS and AMOS, which result in different behaviour due to their multiplicative splitting nature. Since, in contrast to additive splittings, not all coordinate axes are treated in the same manner. The best performance can be achieved with AOS, IE and the fast explicit methods.

Results on Second Order Accurate Schemes In an analogous manner the corresponding performances of the second order accurate schemes PR, WD with $\theta = 0.5$, CN and RKL2 are visualised on the bottom in Figure 7.2. Obviously, the approximations of the second order solvers provide a faster convergence towards the EE solution if τ and M are sufficiently small and large, respectively. The performances of PR, WD with $\theta = 0.5$ and RKL2 are competitive, the worst performance is achieved by CN due to its stability restriction and higher computational costs for a complete factorisation.

Figure 7.1: Convergence of osmosis to the *mandrill* image (256×256) using the EE method with $\tau = 0.12$. **From left to right: (a)** Initial image, each colour channel has the same mean value as the *mandrill* image. **(b)** Osmosis at evolution time $t = 100$. **(c)** Osmosis at evolution time $t = 1000$. **(d)** Osmosis at stopping time $t_F = 100000$.

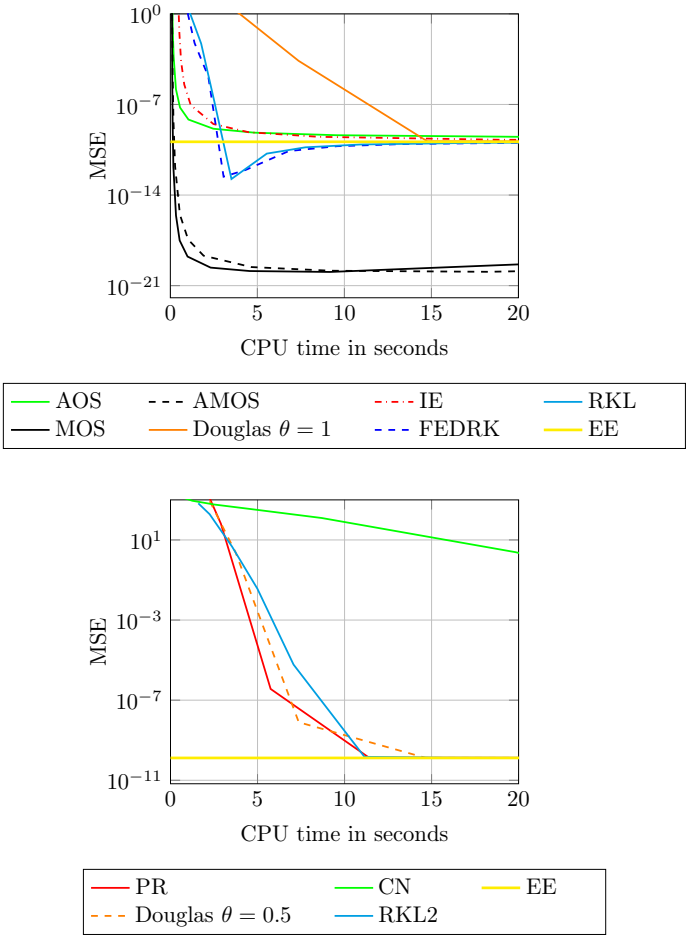


Figure 7.2: Results of osmosis using *mandrill* image (256×256): comparison of the MSE and the corresponding CPU time between the numerical solvers using varying time step sizes τ or cycles M with stopping time $t_F = 100000$. With regard to the MSE, the original image as reference solution is used. For a better comparability, also the MSE of the EE method (yellow line) is visualised. Applying EE leads to a CPU time of 213.64 seconds. **Top:** First order accurate schemes. **Bottom:** Second order accurate schemes.

Overall, the most efficient solvers for the compatible case are AOS, IE, FEDRK and RKL. As the compatible case is trivial, we consider the quasi-compatible case where $\boldsymbol{d}$ is locally modified which is much more interesting in practice.

7.5.2 Quasi-Compatible Case

As already indicated, more interesting is the osmosis process in which the drift vector field can be steered in a local region by its user, denoted as *quasi-compatible* case. To this end, we compare the performances of the solvers using osmosis as a process for seamless image cloning and shadow removal, cf. [295]. Another application of linear osmosis, for instance, is the solution of the light balance problem in thermal-quasi reflectography imaging [206] for cultural heritage, in order to support restoration purposes of mural paintings.

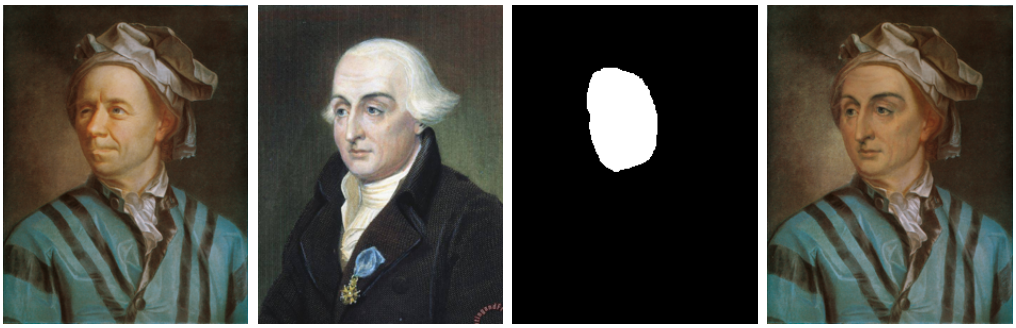


Figure 7.3: Seamless image cloning using osmosis-based image editing for images (300×230) of *Euler* and *Lagrange*. **From left to right: (a)** Painting of *Euler*. **(b)** Painting of *Lagrange*. **(c)** Mask for seamless image cloning. **(d)** Osmosis image cloning at stopping time $t_F = 100000$ using the EE method with $\tau = 0.12$.

Seamless Image Cloning *Image cloning* is an old image processing problem where the general task is to merge two images. Nowadays often *Poisson image editing* [215] is used which is a powerful tool. However, when both input images differ significantly in their contrast, the fused image will also show limitations in contrast. The osmosis-based process is an alternative to Poisson image editing and achieves impressive results, cf. [295].

Let us briefly explain the idea of image cloning at hand of the paintings of *Euler* and *Lagrange* of image size 300×230, illustrated in Figure 7.3. Two images f_1 and f_2 of both mathematicians are given and should be merged in such a way that f_2 replaces image information of f_1. In our case the direct cloning of *Lagrange* on top of *Euler's* face is executed. In doing so, the canonical drift vectors of f_1 on the local region of the face are replaced by the ones of f_2. In addition, at the boundary of the local region the arithmetic mean of both drift vectors is used.

The osmosis process is initialised with f_1 on the whole image domain, where each colour channel has the same mean value as the *Euler* image. The final osmosis image editing solution using the EE method with $\tau = 0.12$ and stopping time $t_F = 100000$ is shown in Figure 7.3. Obviously, using osmosis as seamless image cloning gives a very good contrast reconstruction.

Results on First Order Accurate Schemes: In order to give an evaluation, we analyse the MSE and the corresponding CPU time of the first order accurate schemes AOS, MOS, AMOS, WD with $\theta = 1$, IE, FEDRK and RKL using varying time step sizes τ or cycles M with the reference solution. The reference solution[1] is computed using the EE method with stopping time $t_F = 100000$, cf. Figure 7.3. The performances are visualised on the top in Figure 7.4. The splitting schemes AOS, MOS and AMOS do not converge to the elliptic steady state solution unless the time step size as $\tau \to 0$. This demonstrates a similar behaviour as also seen for the compatible case. In contrast, the solvers IE, FEDRK and RKL are highly efficient and achieve competitive results. The WD scheme is only accurate for sufficiently small τ.

[1] The approximation by the EE method gives in general the most accurate solution, as already seen in the compatible case.

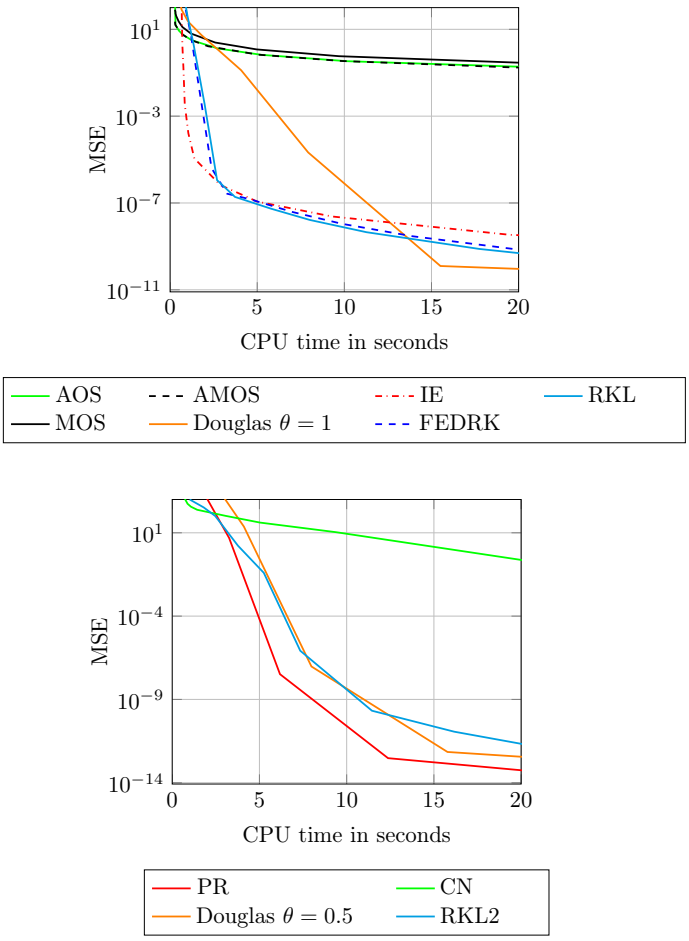


Figure 7.4: Results of seamless image cloning using osmosis-based image editing for images (300×230) of *Euler* and *Lagrange*: comparison of the MSE and the corresponding CPU time between the numerical solvers using varying time step sizes τ or cycles M at stopping time $t_F = 100000$. With regard to the MSE, the reference solution computed with the EE method is used (cf. Figure 7.3). Applying EE leads to a CPU time of 211.46 seconds. **Top:** First order accurate schemes. **Bottom:** Second order accurate schemes.

Results on Second Order Accurate Schemes: The performances of the second order accurate schemes PR, WD with $\theta = 0.5$, CN and RKL2 are shown on the bottom in Figure 7.4. Once again, these solvers provide a faster convergence than the first order schemes, when τ and M is chosen sufficiently small and large. As expected, these solver are linked with higher computational costs. The best and the worst performance achieves PR and CN, respectively.

In total, IE, FEDRK, RKL and PR are the most efficient solvers based on the examined example of seamless image cloning.

Shadow Removal Another important image processing task is *shadow removal*. This is often applied as a preprocessing step in which shadows within an image can lead to complications in the subsequent image processing due to their artefacts and should be avoided.

In general, the task of shadow removal is to remove the shadow appearing in an image while preserving the image geometry. This can be mathematically formulated, cf. [203], by the continuous linear osmosis model adapted to shadow removal in the form

$$\begin{cases} \partial_t u(\boldsymbol{x},t) = \Delta u(\boldsymbol{x},t) - \operatorname{div}\Big(\boldsymbol{d}(\boldsymbol{x})u(\boldsymbol{x},t)\Big), & \text{on } \Omega_1 \cup \Omega_2 \times (0,t_F] \\ \partial_t u(\boldsymbol{x},t) = \Delta u(\boldsymbol{x},t), & \text{on } \Omega_3 \times (0,t_F] \\ u(\boldsymbol{x},0) = f(\boldsymbol{x}), & \text{on } \Omega \\ \langle \nabla u(\boldsymbol{x},t) - \boldsymbol{d}(\boldsymbol{x})u(\boldsymbol{x},t), \boldsymbol{n}\rangle = 0, & \text{on } \partial\Omega \times (0,t_F] \end{cases} \tag{7.41}$$

with $\boldsymbol{d}(\boldsymbol{x}) = \nabla(\ln(f(\boldsymbol{x})))$ and where f denotes the shaded positive image. In particular, the rectangular domain Ω is decomposed into three parts $\Omega = \Omega_1 \cup \Omega_2 \cup \Omega_3$. The regions Ω_1, Ω_2 and Ω_3 represent the shaded region, unshaded region and the shadow boundaries of the image f. Furthermore, the drift vector field, which contains the image geometry (gradient data) of the initial image, is modified as $\boldsymbol{d} = \boldsymbol{0}$ on Ω_3. The *shadow boundary*, where $\boldsymbol{d} = \boldsymbol{0}$, is a thin stripe and defines the jump of light caused by the shadow. More precisely, the shadow itself is only encoded by the drift vector field on the edge of the shadow. Setting the drift vectors at the shadow boundaries to zero particularly describes image inpainting by linear diffusion.

Due to the linear inpainting process on Ω_3, the final unshaded image in this area will be blurred based on the properties of the linear diffusion operator. In order to avoid the undesirable blurred artefacts the thickness of the shadow boundary can be optimised in a certain sense. However, the automatic segmentation of the shadow boundary is very challenging. Otherwise, the manual shadow boundary selection may be very tedious. An alternative to this would be a postprocessing inpainting correction step to remove the blurring artefacts. As already noted, also the use of an anisotropic osmosis model is possible. Although this resulting challenge is an interesting aspect, we will not address this problem further in this work.

The osmosis-based shadow removal process is shown in Figure 7.5, starting from a shaded positive image f of size 425×640 and setting $\boldsymbol{d} = \boldsymbol{0}$ at the shadow boundary. The initial condition contains in each colour channel the same mean value as the original shaded image. The final osmosis steady state solution, in which the shadow has been removed, is computed by the EE method with $\tau = 0.12$ and stopping time $t_F = 100000$. The corresponding CPU time of EE amounts to 527.81 seconds. Obviously, blurred artefacts (zooming on the solution) due to linear diffusion inpainting are visible within the steady state solution, otherwise the contrast is also slightly changed. This issue could be improved by using a local model similar to the approach in [72], which solve the osmosis process within a local region of interest and leaves the region outside be unaltered.

Results on AOS, PR, IE and FEDRK: As seen in the previous evaluations the most efficient solvers were AOS, PR, IE and FEDRK[2], so only these are considered here for the shadow removal application. The performances of the solvers concerning the shadow removal quality of the approximations are given visually in the Figure 7.6, also the corresponding CPU times are reported. Obviously, the approximations of AOS and PR suffer from splitting errors, for the latter also the small upper bound of the natural stability restriction is evident. In contrast, IE and FEDRK provide accurate results already for large τ and small M, respectively. Due

[2] The same performance is also achieved by RKL in undocumented tests.

Figure 7.5: Shadow removal by osmosis. **From left to right: (a)** Original image (425×640). **(b)** User-selected shadow boundaries, with $\boldsymbol{d} = \mathbf{0}$ on the boundaries (shadow's edge), otherwise the canonical drift vectors are used. **(c)** Osmosis reconstruction at stopping time $t_F = 100000$ using the EE method with $\tau = 0.12$ and the required CPU time of 527.81 seconds.

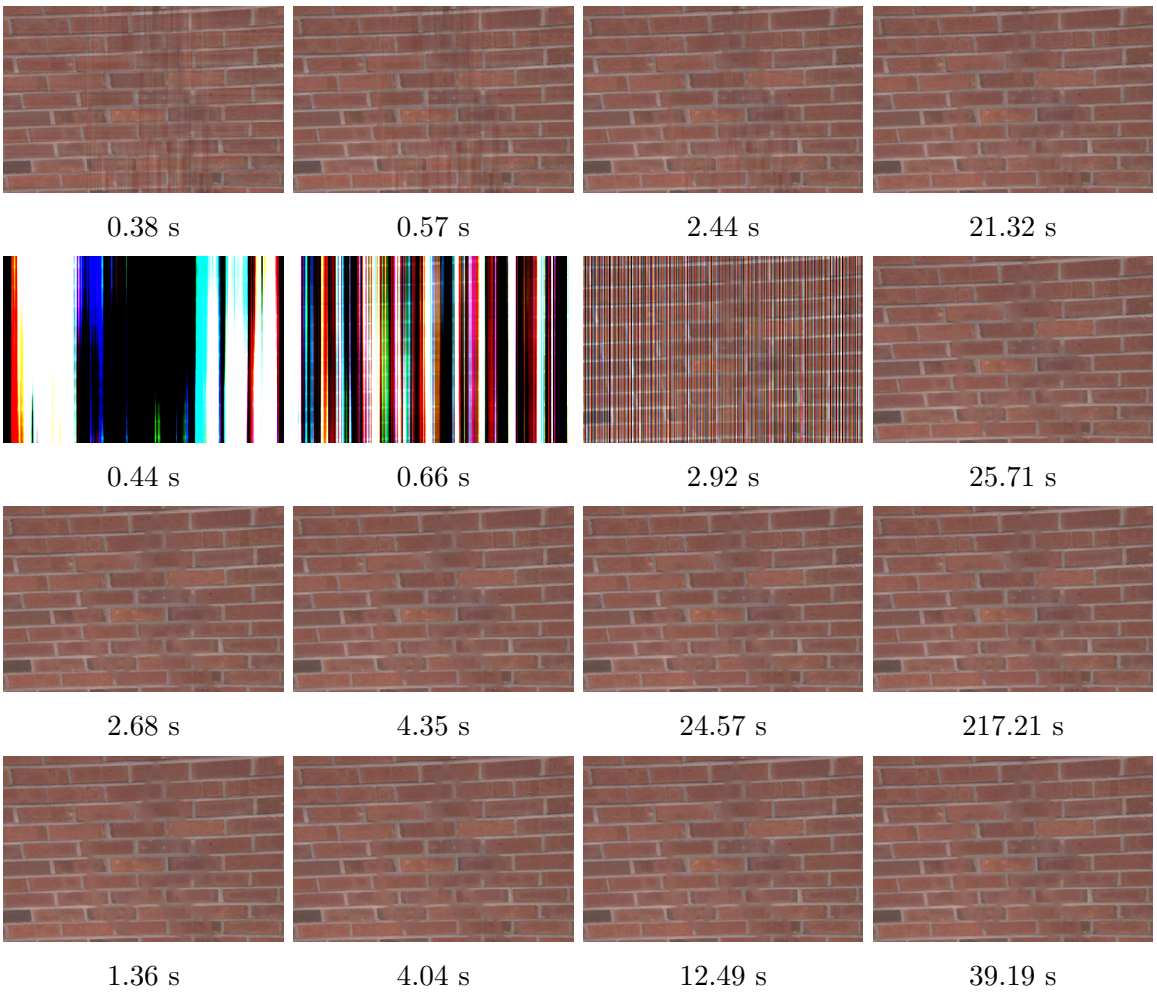


Figure 7.6: Shadow removal by osmosis applied to data from Figure 7.5 using AOS, PR, IE and FEDRK for varying time step sizes τ or cycles M and stopping time $t_F = 100000$. The corresponding CPU time is reported below the respective image. **From left to right: (a)** $\tau = 100000$ or $M = 1$. **(b)** $\tau = 10000$ or $M = 10$. **(c)** $\tau = 1000$ or $M = 100$. **(d)** $\tau = 100$ or $M = 1000$. **From top to bottom: (i)** AOS. **(iii)** PR. **(iii)** IE. **(iv)** FEDRK.

to lower computational costs in the GPU computation, instead of a complete factorisation in the case of IE, the fast explicit method FEDRK is highly efficient.

7.5.3 Higher Resolution

Finally, let us discuss the important issue of dealing with large images as resolutions increase continually due to the camera technology nowadays. To this end, the last experiment of osmosis-based shadow removal is repeated by double upsizing the original data to an image size of 900×1280.

For an evaluation, we compare the MSE and the corresponding CPU time of the schemes AOS, IE, FEDRK and RKL using varying time step sizes τ or cycles M with the reference solution computed with the EE method. The performance of the solvers is shown in the Figure 7.7. As is known, the fast explicit methods FEDRK and RKL are highly efficient and achieve high accuracy coupled with fast computation due to their explicit nature. In contrast, the AOS scheme provides significant low computational complexity, but suffers from its splitting error. Conversely, the IE method yields accurate approximations at the expense of extremely high computational costs.

Based on the fact that the *direct IE* method requires a complete LU factorisation which is highly computationally intensive for larger resolutions, the BiCGSTAB method with $\varepsilon = 10^{-6}$ (gives the best compromise in undocumented experiments) is applied, denoted

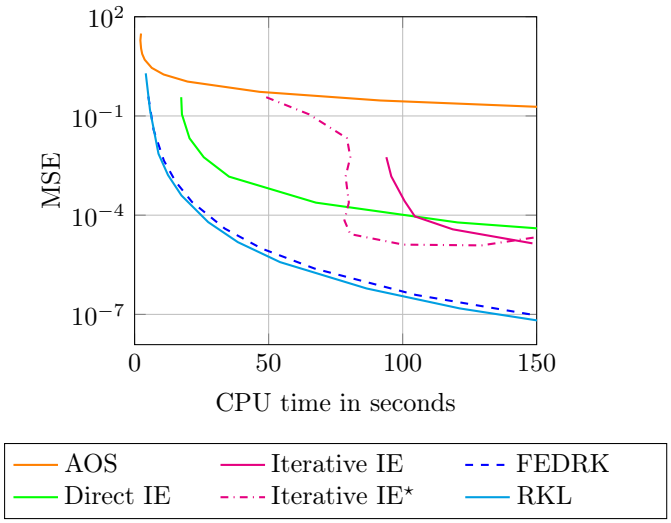


Figure 7.7: Results of osmosis-based shadow removal applied to data from Figure 7.5 for a larger image (900×1280), where the smaller image resolution is double upsized by internal MATLAB function *imresize*: comparison of the MSE and the corresponding CPU time between the schemes AOS, IE, FEDRK and RKL using varying time step sizes τ or cycles M at stopping time $t_F = 100000$. With regard to the MSE, the reference solution computed with the EE method is used. Applying EE leads to a CPU time of 2013 seconds. Within the results we distinguish between direct IE, iterative IE and iterative IE*. Direct, iterative and iterative* is related to complete LU factorisation, iterative BiCGSTAB and BiCGSTAB with modified initial vector $\boldsymbol{x}_0 = \boldsymbol{u}^k$.

here as *iterative IE*. However, the use of a relatively small tolerance ε coupled with a slow convergence rate makes the BiCGSTAB method inefficient, cf. Figure 7.7. We emphasise that the preconditioned BiCGSTAB with incomplete LU factorisation achieves even worse performance and is not reported here. In addition, the approach of using the previous osmosis evolution $\boldsymbol{u}^k$, as initialisation within the BiCGSTAB method for the computation of the new iterate $\boldsymbol{u}^{k+1}$, is examined as already done in Chapter 6. Using the strategy $\boldsymbol{x}_0 = \boldsymbol{u}^k$, labelled as *iterative IE**, significantly improve the efficiency of BiCGSTAB, nevertheless, the iterative method is generally unsuitable for large images in practice as shown in Figure 7.7.

On the basis of this example the KSMOR method is also analysed. For constructing the Krylov subspace $V = \mathcal{K}_r((A-\sigma I)^{-1}, (A-\sigma I)^{-1}\boldsymbol{u}^0)$ large sparse systems of linear equations have to be solved. Tests have shown that fixing the expansion point as $\sigma \leq 10^{-5}$ (but $\sigma \neq 0$) is necessary for reasonable approximations. The corresponding results of KSMOR by applying the complete LU factorisation for a different number of subspaces r are presented in Figure 7.8. Already, $r = 10$ subspaces are sufficient to achieve the same accuracy as IE, while the CPU time required for small time step sizes is dramatically lower. However, the use of the iterative BiCGSTAB method to build the projection matrix V causes extremely high computational costs (not documented here), which result from the extremely slow convergence rate based on the small value σ. Thus, both iterative-based schemes IE and KSMOR suffer from the poor performance of the iterative method itself.

Unfortunately, as a consequence of the unusability of the iterative methods, the analysis of the moment matching property for inexact solvers cannot be performed here. As stated in

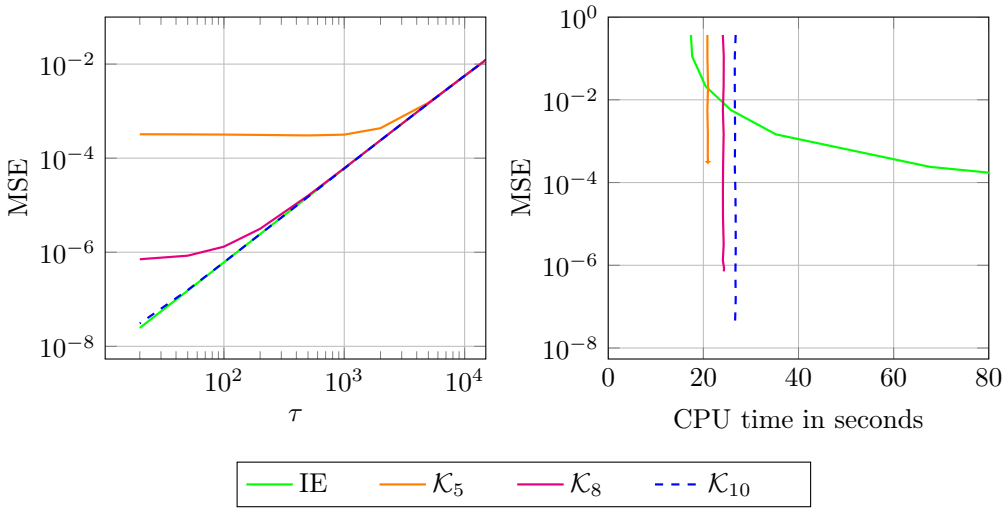


Figure 7.8: Results of osmosis-based shadow removal applied to data from Figure 7.5 for a larger image (900×1280): comparison between IE and KSMOR using varying time step sizes τ at stopping time $t_F = 100000$. The KSMOR method is used by constructing Krylov subspaces $\mathcal{K}_r$ for $r = 5, 8, 10$ and setting $\sigma = 10^{-6}$. Both schemes IE and KSMOR are supplemented with complete LU factorisation. With regard to the MSE, the reference solution computed with the EE method is used. **Left:** Comparison between MSE and varying τ. **Right:** Comparison between MSE and corresponding CPU time. A small number of $r = 10$ subspaces achieves the same MSE as IE with simultaneous faster computation time.

Section 4.5, inexact solvers cannot generally satisfy the moment matching property. Therefore, a suitable choice of the expansion point σ is necessary [26] in order to obtain robustness with respect to perturbations which are generated by the inexact solutions. This is an important aspect for the general applicability of KSMOR, the standard method for large-scale problems. This interesting point is left for future.

Stability Limit for Explicit Schemes So far, we have chosen the upper bound $\tau = 0.12$ for the explicit methods. As already mentioned, the condition can be satisfied by means of an a-priori estimate by a simple precomputation of $c := \max|a_{i,i}|$ for $i = 1, \ldots, N$. For the underlying data from Figure 7.5, the *true* time step size yields

$$\tau^\star = 0.2386 < \frac{1}{c}, \quad \text{with } c \approx 4.1893 \tag{7.42}$$

This upper bound is approximately twice as large as the standard one. As a result, this leads to a significant speed-up when using the fast explicit methods as shown in Figure 7.9. For large images, the RKL scheme even performs slightly better than FEDRK.

Overall, for larger images we prefer the use of FEDRK and RKL coupled with GPU computation. The fast explicit methods are simple schemes and provide a highly efficient approximation of osmosis-based imaging processes. For small and medium size images, the IE method and the KSMOR technique can also be used efficiently.

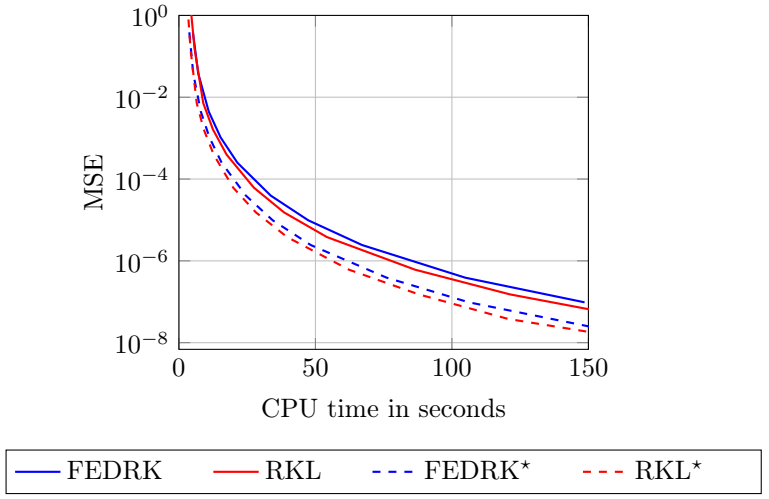


Figure 7.9: Results of osmosis-based shadow removal applied to data from Figure 7.5 for a larger image (900×1280): comparison of the MSE and the corresponding CPU time for FEDRK and RKL using varying cycles M at stopping time $t_F = 100000$. We compare the standard upper bound $\tau = 0.12 < \frac{1}{8}$ and the true natural stability bound (7.22). The latter can be specified as $\tau^\star = 0.2386 < \frac{1}{c}$ with $c = \max|a_{i,i}| \approx 4.1893$, and is denoted by FEDRK* as well as RKL*. With regard to the MSE, the reference solution computed with the EE method is used. Obviously, the larger upper stability bound improves the efficiency. For larger images, the RKL achieves a slightly better performance than FEDRK due to its smaller number of stages s required for preserving the internal stability.

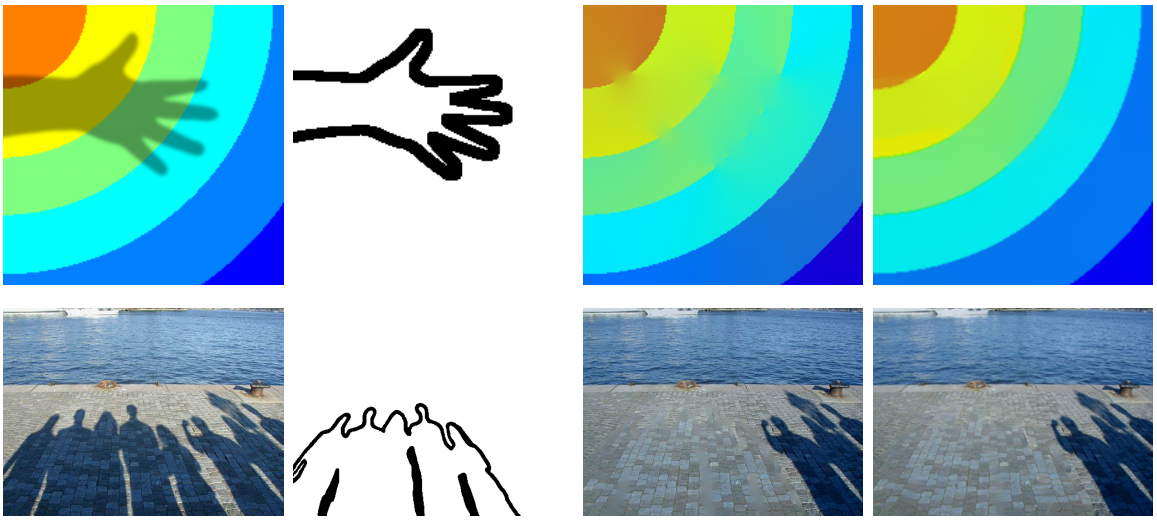


Figure 7.10: Shadow removal via osmosis on (**top**) synthetic image (201×201) and (**bottom**) real-world image (375×505). Comparison between isotropic and anisotropic osmosis using FEDRK with $t_F = 100000$ and $M = 1$. **From left to right: (a)** Shadowed image. **(b)** Mask, fixing $\boldsymbol{d} = \mathbf{0}$ on the boundaries, otherwise the canonical drift vectors are used. **(c)** Isotropic osmosis. **(d)** Anisotropic osmosis.

7.5.4 Anisotropic Osmosis Filtering

Lastly, it should be emphasised that the fast explicit methods can also be employed for anisotropic osmosis filtering. This is exemplarily shown in Figure 7.10 for solving the shadow removal problem by the FEDRK method using isotropic and anisotropic osmosis for synthetic and real-world images. For this experiment, we fixed $t_F = 100000$ and $M = 1$, and used the parameter settings as described[3] in [205]. Anisotropic osmosis effectively removes shadows both for synthetic and real-world images, but an over-smoothing effect is noticed.

7.6 Summary

We have extended the numerical framework for solving the linear osmosis model that has been presented in earlier literature [53, 206, 289]. Even if there exists no theoretical foundation for preserving positivity and numerical stability when using the fast explicit methods, we have demonstrated that FEDRK and RKL are well applicable for osmosis-based image processing. At this point it should be noted that FEDRK and RKL possess a natural damping so that over- and undershoots that can occur in the initial phase are usually smoothed out as the osmosis evolves towards the steady state. In contrast to the inefficient IE solver and the

[3] The source code for anisotropic osmosis filter with respect to shadow removal is available on the website `https://gitlab.developers.cam.ac.uk/sp751/anisotropic-osmosis-filter`. We used the default setting and only replaced the computational intensively explicit *expleja* solver by the highly efficient FEDRK method. Let us mention that the *expleja* solver belongs to the group of Krylov-based matrix exponential approximations, as presented in Section 2.3.2.

inaccurate splitting schemes, fast explicit methods overcome both issues and produce highly efficient and accurate approximations. Due to the fact that FEDRK and RKL are simple schemes based on its explicit nature, they are very well-suited for parallelisation on GPUs, which is especially of importance when using osmosis processes for large images. Remarkably, the fast explicit methods are also ideally suitable for anisotropic osmosis filtering.

In the osmosis framework we also analysed the use of MOR methods, in particular the KSMOR technique. Our experiments have shown that for a small number of $r \leq 10$ subspaces the scheme produces nearly the same approximations than its full order model counterpart. However, for large images that require an iterative method to construct the Krylov basis V, this technique suffers from convergence problems of the iterative method itself just like already the iterative IE method. Furthermore, the slow convergence rate results from using small values σ which are required within the KSMOR process to provide accurate approximations. To overcome this problem, the use of MG methods for computing the projection matrix V appear promising. This issue is of interest in the future.

Let us reiterate once again that we have not provided a theoretical basis for preservation the positivity and (numerical) stability property when using fast explicit methods and KSMOR. This issue will remain left for future research. As already indicated, we have analysed experimentally in undocumented tests that the solvers ensure these aspects when using a large stopping time t_F and a small value σ within the schemes.

Chapter 8

Summary and Outlook

The present thesis has dealt with the efficient solution of relevant applications in connection with linear parabolic-type PDEs. This problem class is still of high importance for the fast and accurate approximation of real-world applications that arise in the areas of image processing, computer vision and engineering. In the work, we focused on the numerical time integration approach that specifically results from the MOL method, a popular technique for solving PDEs in which all but one dimension is discretised. More precisely, MOL allows the use of any numerical method designed for time integration of large sparse semi-discretised ODE systems. Unfortunately, the ODE systems resulting from parabolic PDEs are known to be stiff and present severe numerical difficulties for the use of simple and computationally cheap explicit methods due to the stability requirements on the time step size. Although implicit schemes appear to be a good choice for numerically solving linear parabolic problems, we have demonstrated that there are better schemes for specific real-world problems that are equipped with different model settings.

In order to provide a fast solution for PSC tasks, the long-term simulation of a GES and image osmosis filtering, there was still a need for a numerical method that combines accuracy and reasonable computational efficiency. In the thesis, two popular approaches were discussed, which are based on extended stabilised RK methods as well as MOR techniques. Of course, both method classes are known in their respective scientific fields, but are mostly only used to solve problems that arise exactly in this area, and are often overlooked in order to efficiently solve similar problems from other fields. Therefore, we had set ourselves the aim to fully understand the numerical methods from their theoretical principles and derivations over their numerical analysis to their practical weaknesses and limitations. While the individual techniques are relatively well understood, a complete and joint analysis is missing in the literature to the best of our knowledge. However, this is very important in finding the best numerical method for the underlying linear model problem, since the basis for doing so is the correct use of the particular method.

The first main task of the thesis was to provide a complete and very detailed overview and discussion of several numerical solvers marking the state-of-the-art in diverse scientific fields that are often beneficially used to solve linear parabolic model problems. In particular, we presented a detailed analysis of sparse direct and iterative solvers, exponential integrators, fast explicit methods and model order reduction techniques. In addition to the theoretical basics, we have dealt intensively with the computational aspects from a numerical point of view. At this point it should be noted that we have elaborated in detail the theoretical and numerical similarities and differences of the fast explicit methods developed. From our point of view, this is another important component to supplement the documentation provided. Typically, the different variants RKC/RKL and FED/FEDRK are applied independently

from each other. Both main classes of fast explicit solvers, directly or indirectly derived, have interesting properties which we have also demonstrated in this thesis. While RKC/RKL can provide efficient higher order schemes, FED/FEDRK achieve better damping properties.

Based on the theoretical documentation as well as the linear model problems discussed, we have clearly shown that the fast explicit methods and the model order reduction techniques can significantly outperform the sparse direct and iterative methods. Of course, there is no best numerical method, since all solvers have specific weaknesses and are strongly dependent on the model settings of the underlying problem. However, we believe that our work not only provides extensive theoretical knowledge, but can also be very useful for tackling similar parabolic problems that arise in many applications more easily and efficiently based on the practical applications considered here.

Finally, for each application discussed in Chapters 5, 6 and 7, let us summarise the conclusions by highlighting the outcomes and the possible future interests.

Shape Correspondence Even nowadays, the shape correspondence task is a highly interesting area of research and there are various efficient solution strategies. We introduced simple and efficient time integrators based on MOR methods to compute numerical shape descriptors. By our experimental results, we have demonstrated that MOR methods are highly predestined for solving shape matching tasks using time integration methods of the underlying geometric PDEs. We have elaborated the spectrum-free KSMOR technique and the spectrum-based optimised MCR approach, which provide beneficial use and consequently achieve a better trade-off between quality and computational effort compared to the state-of-the-art solvers in the class of time-evolution methods. In our opinion, we have tweaked the MOR approaches very close to their limits with regard to their use in the PSC framework. In this context, we have discussed various questions when using the MCR approach, ranging from stable eigenvalue computations over scaling the integration domain to changing the initial condition and analysing the calculation method for the reduced solution.

We have shown through several real-world experiments that our optimised MCR technique is highly efficient for applications with high resolutions and clearly outperforms their direct counterparts HKS and WKS. In contrast, the KSMOR technique is less efficient compared to the spectrum-based methods, but can achieve a high correspondence quality, which we highlighted by mapping indicator functions. Both developed MOR methods are very powerful, nearly free of parameters and ultimately easy to implement.

For the future let us highlight two possible tasks that arise from our work. First, the use of soft correspondence maps as introduced in this thesis appears to be promising and is an interesting subject of research. As we pointed out, the soft correspondence map could be used to extract the ideal binary assignment matrix from itself using a more sophisticated optimisation approach. Nevertheless, in practice it needs sparse information for this in the soft correspondence matrix. Of course, it is not trivial to develop a method that could make full use of the soft correspondence information, which may result in a quite powerful approach. Another task is to integrate our MOR approaches into the functional map framework, which uses initial correspondences based on feature descriptors to compute a dense correspondence between points. We expect that our methods are useful in this framework as well, so the dense shape correspondence performances can be notably improved compared to the kernel-based methods HKS and WKS.

Geothermal Energy Storage We have numerically studied the recent GES technology as an effective alternative to storing the excess energy generated during the summer. Due to the fact that the heat tank is open downwards and interacts with the earth below the tank, long-term heat evolution simulations of the GES are required to assess the profitability, but also to ensure the heat supply of the consumer. In this context, we have demonstrated that a two-dimensional GES simulation based on a linear heat equation equipped with external and internal boundary conditions is suitable for representing the long-term behaviour of real GES facilities. This has been shown experimentally using real-world data from a three-dimensional test field. As a result, the simplified and dimensionally reduced GES model can easily be used in practice either for simulation purposes or for parameter optimisation.

In order to efficiently solve the two-dimensional GES model, we focused on the FEDRK method and the KSMOR technique, for which both solvers were investigated in detail. In particular, we have seen that the latter scheme is not easy to apply for the GES problem due to the large-scale input resulting from the discretisation of the internal boundary conditions. Under these circumstances, we have successfully adapted the original KSMOR technique by own adaptations for practical use when dealing with large input vectors. Our proposed efficient KSMOR* technique is based on an input matrix reduction via snapshots, so that the subspace construction is obtained in an explicit manner. We have demonstrated the practical usability of KSMOR*, which in this form represents a new variant of existing schemes. In total, FEDRK and KSMOR* turned out to be the most powerful methods for an efficient numerical simulation of our GES application. The only difference between the two techniques is the local and global behaviour of their solutions obtained. While FEDRK produces accurate approximations globally, KSMOR* shows oscillations, especially in a local area around the interfaces, which directly results from the input matrix construction.

For future research at least two aspects are of interest. Besides the successful application of the proposed KSMOR* method, we have not dealt with theoretical aspects. However, this is important to ensure the applicability of our approach to similar model problems. Apart from that, the GES setup is modelled on the assumption that the flowing groundwater has a large distance to the heat tank, but this requirement can of course be relatively strict. The integration of flowing groundwater requires a model adaptation of the underlying continuous and discrete model. In this case, the proposed numerical schemes must be carefully restudied, since the advection has a strong influence on the approximation to be computed, for example its numerical stability.

Osmosis Filtering We have discussed efficient numerical schemes for isotropic osmosis filtering based on direct and iterative methods, splitting methods, fast explicit methods and the KSMOR technique. We studied both the compatible and the quasi-compatible case and, for the latter in particular, presented a detailed performance analysis using relevant visual computing problems such as seamless image cloning and shadow removal. Through our experiments, we have clearly shown that the fast explicit solvers coupled with GPU computation provide highly efficient approximations to osmosis-based imaging processes, even for large images. In addition, we have highlighted the beneficial use of such methods for anisotropic osmosis filtering, which is an important extension of the isotropic osmosis filter.

As future research there are two possible main issues. While fast explicit methods are well applicable from a practical point of view, the preservation of positivity and stability

is theoretically not guaranteed because the system matrix is not symmetric. In all (also undocumented) tests, we were able to experimentally ensure these aspects for large stopping times. The positivity preservation is mainly due to the natural damping of FEDRK and RKL, so over- and undershoots are usually smoothed out as the osmosis evolves towards the steady state. The stability preservation in practice generally relies on the osmosis-based system matrix, which is close to a normal matrix. From a theoretical point of view, however, the full understanding of these aspects and the corresponding safe and stable approximate solution are of primary interest. The second future matter concerns the KSMOR technique. Besides theoretical aspects that need to be addressed, this method suffers from convergence problems of the iterative method itself. An interesting point is therefore the use of MG methods for computing the projection matrix, which appears promising to deal efficiently with large images.

Bibliography

[1] A. Abdulle. Explicit Stabilized Runge-Kutta Methods. In B. Engquist, editor, *Encyclopedia of Applied and Computational Mathematics*, pages 460–468. Springer, Berlin, Heidelberg, 1st edition, 2015.

[2] M. Abdykarim, J. Berger, D. Dutykh, and A. Agbossou. An efficient numerical method for a long-term simulation of heat and mass transfer: the case of an insulated rammed earth wall. In V. Corrado, E. Fabrizio, A. Gasparella, and F. Patuzzi, editors, *Proceedings of Building Simulation 2019: 16th Conference of IBPSA*, pages 1444–1451, 2019.

[3] M. Abdykarim, J. Berger, D. Dutykh, L. Soudani, and A. Agbossou. Critical assessment of efficient numerical methods for a long-term simulation of heat and moisture transfer in porous materials. *International Journal of Thermal Sciences*, 145:105982, 2019.

[4] R. C. Aiken. *Stiff Computation.* Oxford University Press, Inc., 1985.

[5] A. Al Takash, M. Beringhier, M. Hammoud, and J.-C. Grandidier. Numerical approach based on the collection of the most significant modes to solve cyclic transient thermal problems involving different time scales. *Journal of Computational Physics*, 375:950–959, 2018.

[6] P. F. Alcantarilla, J. Nuevo, and A. Bartoli. Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces. In T. Burghardt, D. Damen, W. Mayol-Cuevas, and M. Mirmehdi, editors, *Procedings of the British Machine Vision Conference*, pages 13.1–13.11. BMVA Press, 2013.

[7] V. Alexiades, G. Amiez, and P.-A. Gremaud. Super-time-stepping acceleration of explicit schemes for parabolic problems. *Communications in Numerical Methods in Engineering*, 12(1):31–42, 1996.

[8] A. C. Antoulas. *Approximation of Large-Scale Dynamical Systems.* SIAM, 2005.

[9] A. C. Antoulas, C. A. Beattie, and S. Gugercin. Interpolatory Model Reduction of Large-Scale Dynamical Systems. In J. Mohammadpour and K. M. Grigoriadis, editors, *Efficient Modeling and Control of Large-Scale Systems*, pages 3–58. Springer, Boston, MA, 2010.

[10] A. C. Antoulas, R. Ionutiu, N. Martins, E. J. W. ter Maten, K. Mohaghegh, R. Pulch, J. Rommes, M. Saadvandi, and M. Striebel. Model Order Reduction: Methods, Concepts and Properties. In M. Günther, editor, *Coupled Multiscale Simulation and Optimization in Nanoelectronics*, pages 159–265. Springer, Berlin, Heidelberg, 2015.

[11] A. C. Antoulas, D. C. Sorensen, and S. Gugercin. A survey of model reduction methods for large-scale systems. In *Structured Matrices in Mathematics, Computer Science, and Engineering I*. 2001.

[12] J. A. Atwell and B. B. King. Proper Orthogonal Decomposition for Reduced Basis Feedback Controllers for Parabolic Equations. *Mathematical and Computer Modelling*, 33(1-3):1–19, 2001.

[13] M. Aubry. *Representing 3D models for alignment and recognition*. PhD thesis, Sciences mathématiques de Paris Centre, Sorbonne University, Paris, France, 2015.

[14] M. Aubry, U. Schlickewei, and D. Cremers. The Wave Kernel Signature: A Quantum Mechanical Approach to Shape Analysis. In *Proceedings of the 2011 IEEE International Conference on Computer Vision Workshops*, pages 1626–1633, 2011.

[15] H. D. Baehr and K. Stephan. *Heat and Mass Transfer*. Springer, Berlin, Heidelberg, 3rd edition, 2011.

[16] M. Bähr, M. Breuß, Y. Quéau, A. S. Boroujerdi, and J. D. Durou. Fast and accurate surface normal integration on non-rectangular domains. *Computational Visual Media*, 3(2):107–129, 2017.

[17] M. Bähr, M. Breuß, and R. Wunderlich. Fast explicit diffiusion for long-time integration of parabolic problems. *AIP Conference Proceedings*, 1863(1):410002, 2017.

[18] M. Bähr, R. Dachsel, and M. Breuß. Fast Solvers for Solving Shape Matching by Time Integration. In M. Welk, M. Urschler, and P. M. Roth, editors, *Proceedings of OAGM Workshop 2018*, pages 65–72. Verlag der TU Graz, 2018.

[19] Z. Bai. Krylov subspace techniques for reduced-order modeling of large-scale dynamical systems. *Applied Numerical Mathematics*, 43(1-2):9–44, 2002.

[20] Z. Bai and Q. Ye. Error estimation of the Padé approximation of transfer functions via the Lanczos process. *Electronic Transactions on Numerical Analysis*, 7:1–17, 1998.

[21] D. Barash, M. Israeli, and R. Kimmel. An Accurate Operator Splitting Scheme for Nonlinear Difusion Filtering. In M. Kerckhove, editor, *Scale-Space and Morphology in Computer Vision*, volume 2106 of *LNCS*, pages 281–289. Springer, Berlin, Heidelberg, 2001.

[22] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, 1994.

[23] R. H. Bartels and G. W. Stewart. Solution of the Matrix Equation AX + XB = C [F4]. *Communications of the ACM*, 15(9):820–826, 1972.

[24] M. Bassenne, L. Fu, and A. Mani. Time-Accurate and highly-Stable Explicit operators for stiff differential equations. *Journal of Computational Physics*, 424(109847), 2021.

[25] U. Baur, P. Benner, and L. Feng. Model Order Reduction for Linear and Nonlinear Systems: A System-Theoretic Perspective. *Archives of Computational Methods in Engineering*, 21(4):331–358, 2014.

[26] C. A. Beattie and S. Gugercin. Inexact Solves in Krylov-based Model Reduction. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 3405–3411, 2006.

[27] C. A. Beattie, S. Gugercin, and V. Mehrmann. Model reduction for systems with inhomogeneous initial conditions. *Systems & Control Letters*, 99:99–106, 2017.

[28] T. Bechtold, E. B. Rudnyi, and J. G. Korvink. Error indicators for fully automatic extraction of heat-transfer macromodels for MEMS. *Journal of Micromechanics and Microengineering*, 15(3):430–440, 2004.

[29] T. Bechtold, E. B. Rudnyi, and J. G. Korvink. *Fast Simulation of Electro-thermal MEMS: Efficient Dynamic Compact Models*. Springer, Berlin, Heidelberg, 1st edition, 2007.

[30] S. Beisel. Vermessung, Modellierung und Bewertung des Erdreichwärmeübertragers beim Passiv-Solarhaus Cölbe. Master's thesis, Department of Physics, Hessen University, Marburg, Germany, 1999.

[31] M. Belkin, J. Sun, and Y. Wang. Discrete Laplace Operator on Meshed Surfaces. In *Proceedings of the 24th Annual Symposium on Computational Geometry*, pages 278–287. ACM, 2008.

[32] R. Bellman. *Introduction to Matrix Analysis*. SIAM, 2nd edition, 1997.

[33] A. Benaarbia and A. Chrysochoos. Proper orthogonal decomposition preprocessing of infrared images to rapidly assess stress-induced heat source fields. *Quantitative InfraRed Thermography Journal*, 14(1):132–152, 2017.

[34] P. Benedict and H. Khanal. Super time-stepping schemes for conduction-radiation heat transfer problem. *Neural, Parallel, and Scientific Computations*, 25(4):457–468, 2017.

[35] P. Benner. Numerical Linear Algebra for Model Reduction in Control and Simulation. *GAMM-Mitteilungen*, 29(2):275–296, 2006.

[36] P. Benner, Z. Bujanović, P. Kürschner, and J. Saak. A Numerical Comparison of Different Solvers for Large-Scale, Continuous-Time Algebraic Riccati Equations and LQR Problems. *SIAM Journal on Scientific Computing*, 42(2):A957–A996, 2020.

[37] P. Benner, J. M. Claver, and E. S. Quintana-Ortí. Efficient Solution of Coupled Lyapunov Equations via Matrix Sign Function Iteration. In *Proceedings of 3rd Portuguese Conference on Automatic Control*, pages 205–210, 1998.

[38] P. Benner, L. Feng, and E. B. Rudnyi. Using the Superposition Property for Model Reduction of Linear Systems with a Large Number of Inputs. In *Proceedings of the 18th International Symposium on Mathematical Theory of Networks & Systems*, 2008.

[39] P. Benner, S. Gugercin, and K. Willcox. A Survey of Projection-Based Model Reduction Methods for Parametric Dynamical Systems. *SIAM Review*, 57(4):483–531, 2015.

[40] P. Benner, R. Herzog, N. Lang, I. Riedel, and J. Saak. Comparison of model order reduction methods for optimal sensor placement for thermo-elastic models. *Engineering Optimization*, 51(3):465–483, 2019.

[41] P. Benner and A. Schneider. Model Reduction for Linear Descriptor Systems with Many Ports. In M. Günther, A. Bartel, M. Brunk, S. Schöps, and M. Striebel, editors, *Progress in Industrial Mathematics at ECMI 2010*, pages 137–143. Springer, Berlin, Heidelberg, 2012.

[42] M. Benzi. Preconditioning Techniques for Large Linear Systems: A Survey. *Journal of Computational Physics*, 182(2):418–477, 2002.

[43] G. Berkooz, P. Holmes, and J. L. Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25:539–575, 1993.

[44] B. Besselink, U. Tabak, A. Lutowska, N. van de Wouw, H. Nijmeijer, D. J. Rixen, M. E. Hochstenbach, and W. H. A. Schilders. A comparison of model reduction techniques from structural dynamics, numerical mathematics and systems and control. *Journal of Sound and Vibration*, 332(19):4403–4422, 2013.

[45] D. Bonvin and D. A. Mellichamp. A unified derivation and critical review of modal approaches to model reduction. *International Journal of Control*, 35(5):829–848, 1982.

[46] A. J. Bosch. The Factorization of a Square Matrix Into Two Symmetric Matrices. *The American Mathematical Monthly*, 93(6):462–464, 1986.

[47] M. Botsch, D. Bommes, and L. Kobbelt. Efficient Linear System Solvers for Mesh Processing. In R. Martin, H. Bez, and M. Sabin, editors, *Mathematics of Surfaces XI*, pages 62–83. Springer, Berlin, Heidelberg, 2005.

[48] B. Brands, J. Mergheim, and P. Steinmann. Reduced-order modelling for linear heat conduction with parametrised moving heat sources. *GAMM-Mitteilungen*, 39(2):170–188, 2016.

[49] A. M. Bronstein and M. M. Bronstein. Shape Recognition with Spectral Distances. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):1065–1071, 2011.

[50] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. *Numerical Geometry of Non-Rigid Shapes*. Springer, New York, 1st edition, 2009. TOSCA dataset freely available at `http://tosca.cs.technion.ac.il/book/resources_data.html`.

[51] M. M. Bronstein and I. Kokkinos. Scale-invariant heat kernel signatures for non-rigid shape recognition. In *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1704–1711, 2010.

[52] A. Bultheel and M. Van Barel. Padé techniques for model reduction in linear system theory: a survey. *Journal of Computational and Applied Mathematics*, 14(3):401–438, 1986.

[53] L. Calatroni, C. Estatico, N. Garibaldi, and S. Parisotto. Alternating Direction Implicit (ADI) schemes for a PDE-based image osmosis model. *Journal of Physics: Conference Series*, 904:012014, 2017.

[54] R. M. Caplan, Z. Mikić, J. A. Linker, and R. Lionello. Advancing parabolic operators in thermodynamic MHD models: Explicit super time-stepping versus implicit schemes with Krylov solvers. *Journal of Physics: Conference Series*, 837:012016, 2017.

[55] A. Castagnotto, H. K. F. Panzer, and B. Lohmann. Fast H2-Optimal Model Order Reduction Exploiting the Local Nature of Krylov-Subspace Methods. In *2016 European Control Conference*, pages 1958–1969. IEEE Computer Society Press, 2016.

[56] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll. Image Selective Smoothing and Edge Detection by Nonlinear Diffusion. *SIAM Journal on Numerical Analysis*, 29(1):182–193, 1992.

[57] J. N. Chadwick, S. S. An, and D. L. James. Harmonic Shells: A Practical Nonlinear Sound Model for Near-Rigid Thin Shells. *ACM Transactions on Graphics*, 28(5):1–10, 2009.

[58] D. Chaniotis and M. A. Pai. Model Reduction in Power Systems Using Krylov Subspace Methods. *IEEE Transactions on Power Systems*, 20(2):888–894, 2005.

[59] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current Science*, 78(7):808–817, 2000.

[60] S. Chaturantabut and D. C. Sorensen. Nonlinear Model Reduction via Discrete Empirical Interpolation. *SIAM Journal on Scientific Computing*, 32(5):2737–2764, 2010.

[61] F. Chinesta, A. Huerta, G. Rozza, and K. Willcox. Model Reduction Methods. In E. Stein, R. de Borst, and T. J. R. Hughes, editors, *Encyclopedia of Computational Mechanics Second Edition*, pages 1–36. John Wiley & Sons, Ltd., 2017.

[62] R. R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30, 2006.

[63] B. Commerçon, R. Teyssier, E. Audit, P. Hennebelle, and G. Chabrier. Radiation hydrodynamics with adaptive mesh refinement and application to prestellar core collapse - I Methods. *Astronomy and Astrophysics*, 529:A35, 2011.

[64] S. M. Cox and P. C. Matthews. Exponential Time Differencing for Stiff Systems. *Journal of Computational Physics*, 176(2):430–455, 2002.

[65] G. Cramer. *Introduction à l'analyse des lignes courbes algébriques*. Geneve: Freres Cramer & Cl. Philbert, 1750.

[66] M. Cruz Varona and B. Lohmann. Model Reduction of Linear Time-Varying Systems with Applications for Moving Loads. In P. Benner, M. Ohlberger, A. Patera, G. Rozza, and K. Urban, editors, *Model Reduction of Parametrized Systems*, pages 367–386. Springer, Cham, 2017.

[67] R. Dachsel, M. Breuß, and L. Hoeltgen. The Classic Wave Equation Can Do Shape Correspondence. In M. Felsberg, A. Heyden, and N. Krüger, editors, *Computer Analysis of Images and Patterns*, volume 10424 of *LNCS*, pages 264–275. Springer, Cham, 2017.

[68] R. Dachsel, M. Breuß, and L. Hoeltgen. Shape Matching by Time Integration of Partial Differential Equations. In F. Lauze, Y. Dong, and A. B. Dahl, editors, *Scale Space and Variational Methods in Computer Vision*, volume 10302 of *LNCS*, pages 669–680. Springer, Cham, 2017.

[69] R. Dachsel, M. Breuß, and L. Hoeltgen. A Study of Spectral Expansion for Shape Correspondence. In M. Welk, M. Urschler, and P. M. Roth, editors, *Proceedings of OAGM Workshop 2018*, pages 73–79. Verlag der TU Graz, 2018.

[70] G. Dahlquist. *Stability and Error Bounds in the Numerical Integration of Ordinary Differential Equations.* Almqvist & Wiksells, 1959.

[71] A. Daraghmeh, C. Hartmann, and N. Qatanani. Balanced model reduction of linear systems with nonzero initial conditions: Singular perturbation approximation. *Applied Mathematics and Computation*, 353:295–307, 2019.

[72] M. d'Autume, J.-M. Morel, and E. Meinhardt-Llopis. A Flexible Solution to the Osmosis Equation for Seamless Cloning and Shadow Removal. In *Proceedings of the 25th IEEE International Conference on Image Processing*, pages 2147–2151, 2018.

[73] T. A. Davis. *Direct Methods for Sparse Linear Systems.* SIAM, 2006.

[74] T. A. Davis. Algorithm 930: FACTORIZE: An Object-Oriented Linear System Solver for MATLAB. *ACM Transactions on Mathematical Software*, 39(4):1–18, 2013.

[75] T. A. Davis, S. Rajamanickam, and W. M. Sid-Lakhdar. A survey of direct methods for sparse linear systems. *Acta Numerica*, 25:383–566, 2016.

[76] E. J. Davison. A Method for Simplifying Linear Dynamic Systems. *IEEE Transactions on Automatic Control*, 11(1):93–101, 1966.

[77] C. de Villemagne and R. E. Skelton. Model reductions using a projection formulation. *International Journal of Control*, 46(6):2141–2169, 1987.

[78] M. Desbrun, M. Meyer, P. Schröder, and A. H. Barr. Implicit Fairing of Irregular Meshes Using Diffusion and Curvature Flow. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, pages 317–324. ACM Press/Addison-Wesley Publishing, 1999.

[79] M. P. do Carmo. *Differential geometry of curves and surfaces: revised and updated second edition.* Dover Publications, 2016.

[80] J. Douglas. On the Numerical Integration $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{\partial u}{\partial t}$ by Implicit Methods. *Journal of the Society for Industrial and Applied Mathematics*, 3(1):42–65, 1955.

[81] F. Dulat, J. P. Katoen, and V. Y. Nguyen. Model Checking Markov Chains Using Krylov Subspace Methods: An Experience Report. In A. Aldini, M. Bernardo, L. Bononi, and V. Cortellessa, editors, *Computer Performance Engineering*, volume 6342 of *LNCS*, pages 115–130. Springer, Berlin, Heidelberg, 2010.

[82] R. Eid. *Time Domain Model Reduction By Moment Matching.* PhD thesis, Faculty of Mechanical Engineering, Technical University of Munich, Germany, 2009.

[83] M. Eisenberger, Z. Lähner, and D. Cremers. Divergence-Free Shape Correspondence by Deformation. *Computer Graphics Forum*, 38(5):1–12, 2019.

[84] D. F. Enns. Model reduction with balanced realizations: An error bound and a frequency weighted generalization. In *Proceedings of the 23rd IEEE Conference on Decision and Control*, pages 127–132, 1984.

[85] N. J. Falkiewicz and C. E. S. Cesnik. Proper Orthogonal Decomposition for Reduced-Order Thermal Solution in Hypersonic Aerothermoelastic Simulations. *AIAA Journal*, 49(5):994–1009, 2011.

[86] G. Fan, K. Pan, and M. Canova. A Comparison of Model Order Reduction Techniques for Electrochemical Characterization of Lithium-ion Batteries. In *Proceedings of the 54th IEEE Conference on Decision and Control*, pages 3922–3931, 2015.

[87] Y. Fang, M. Sun, M. Kim, and K. Ramani. Heat-Mapping: A Robust Approach Toward Perceptually Consistent Mesh Segmentation. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2145–2152, 2011.

[88] P. Feldmann. Model order reduction techniques for linear systems with large numbers of terminals. In *Proceedings of the Conference on Design, Automation and Test in Europe*, page 20944. IEEE Computer Society Press, 2004.

[89] P. Feldmann and R. W. Freund. Efficient Linear Circuit Analysis by Padé Approximation via the Lanczos Process. In *Proceedings of the Conference on European Design Automation*, pages 170–175. IEEE Computer Society Press, 1994.

[90] P. Feldmann and F. Liu. Sparse and efficient reduced order modeling of linear subcircuits with large number of terminals. In *IEEE/ACM International Conference on Computer Aided Design*, pages 88–92, 2004.

[91] L. Feng. Review of model order reduction methods for numerical simulation of nonlinear circuits. *Applied Mathematics and Computation*, 167(1):576–591, 2005.

[92] L. Feng, P. Benner, and J. G. Korvink. System-Level Modeling of MEMS by Means of Model Order Reduction (Mathematical Approximations) - Mathematical Background. In T. Bechtold, G. Schrag, and L. Feng, editors, *System-Level Modeling of MEMS*, pages 53–93. John Wiley & Sons, Ltd, 2013.

[93] O. Föllinger. *Regelungstechnik: Einführung in die Methoden und ihre Anwendung*. VDE-Verlag, 11th edition, 2013.

[94] R. W. Freund. Reduced-Order Modeling Techniques Based on Krylov Subspaces and Their Use in Circuit Simulation. In B. N. Datta, editor, *Applied and Computational Control, Signals, and Circuits: Volume 1*, pages 435–498. Birkhäuser, Boston, MA, 1999.

[95] R. W. Freund. Krylov-subspace methods for reduced-order modeling in circuit simulation. *Journal of Computational and Applied Mathematics*, 123(1-2):395–421, 2000.

[96] R. W. Freund. Model reduction methods based on Krylov subspaces. *Acta Numerica*, 12:267–319, 2003.

[97] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev, and H.-P. Seidel. Towards PDE-Based Image Compression. In N. Paragios, O. Faugeras, T. Chan, and C. Schnörr, editors, *Variational, Geometric, and Level Set Methods in Computer Vision*, volume 3752 of *LNCS*, pages 37–48. Springer, Berlin, Heidelberg, 2005.

[98] E. Gallopoulos and Y. Saad. Efficient Solution of Parabolic Equations by Krylov Approximation Methods. *SIAM Journal on Scientific and Statistical Computing*, 13(5):1236–1264, 1992.

[99] M. Gentzsch and A. Schlüter. Über ein Einschrittverfahren mit zyklischer Schrittweitenänderung zur Lösung parabolischer Differentialgleichungen. *Zeitschrift für Angewandte Mathematik und Mechanik*, 58:T415–T416, 1978.

[100] S. A. Gerschgorin. Über die Abgrenzung der Eigenwerte einer Matrix. *Izvestija Akademii Nauk SSSR, Serija Matematika*, 7(3):749–754, 1931.

[101] Y. C. Gerstenmaier and G. Wachutka. Time dependent temperature fields calculated using eigenfunctions and eigenvalues of the heat conduction equation. *Microelectronics Journal*, 32(10-11):801–808, 2001.

[102] A. Ghai, C. Lu, and X. Jiao. A Comparison of Preconditioned Krylov Subspace Methods for Large-Scale Nonsymmetric Linear Systems. *Numerical Linear Algebra with Applications*, 26(1):e2215, 2019.

[103] L. Gilles, C. R. Vogel, and B. L. Ellerbroek. Multigrid preconditioned conjugate-gradient method for large-scale wave-front reconstruction. *Journal of the Optical Society of America A*, 19(9):1817–1822, 2002.

[104] G. H. Golub and C. F. Van Loan. *Matrix Computation*. The Johns Hopkins University Press, 3rd edition, 1996.

[105] A. R. Gourlay and J. L. Morris. The Extrapolation of First Order Methods for Parabolic Partial Differential Equations, II. *SIAM Journal on Numerical Analysis*, 17(5):641–655, 1980.

[106] M. Green and D. J. N. Limebeer. *Linear Robust Control*. Prentice-Hall, Inc., 1994.

[107] S. Grewenig. *Fast Explicit Methods for PDE-Based Image Analysis.* PhD thesis, Department of Mathematics, Saarland University, Saarbrücken, Germany, 2013.

[108] S. Grewenig, J. Weickert, and A. Bruhn. From Box Filtering to Fast Explicit Diffusion. In M. Goesele, S. Roth, A. Kuijper, B. Schiele, and K. Schindler, editors, *Pattern Recognition*, volume 6376 of *LNCS*, pages 533–542. Springer, Berlin, Heidelberg, 2010.

[109] E. J. Grimme. *Krylov Projection Methods for Model Reduction.* PhD thesis, Department of Electrical Engineering, University of Illinois at Urbana-Champaign, USA, 1997.

[110] C. Gu. *Model Order Reduction of Nonlinear Dynamical Systems.* PhD thesis, Electrical Engineering and Computer Sciences, University of California, Berkeley, USA, 2011.

[111] J. Guenther and W. Morgan. An Adaptive, Highly Accurate and Efficient, Parker-Sochacki Algorithm for Numerical Solutions to Initial Value Ordinary Differential Equation Systems. *SIAM Undergraduate Research Online*, 53(12):257–281, 2019.

[112] S. Gugercin and A. A. Antoulas. Model reduction of large-scale systems by least squares. *Linear Algebra and its Applications*, 415(2-3):290–321, 2006.

[113] S. Gugercin and C. A. Beattie. H2 model reduction for large-scale linear dynamical systems. *SIAM Journal on Matrix Analysis and Applications*, 30(2):609–638, 2008.

[114] I. Gustafsson. A class of first order factorization methods. *BIT Numerical Mathematics*, 18(2):142–156, 1978.

[115] R. J. Guyan. Reduction of Stiffness and Mass Matrices. *AIAA Journal*, 3(2):380–380, 1965.

[116] P. Gwosdek, H. L. Zimmer, S. Grewenig, A. Bruhn, and J. Weickert. A Highly Efficient GPU Implementation for Variational Optic Flow Based on the Euler-Lagrange Framework. In K. N. Kutulakos, editor, *Trends and Topics in Computer Vision*, volume 6554 of *LNCS*, pages 372–383. Springer, Berlin, Heidelberg, 2012.

[117] W. Hackbusch. *Multi-Grid Methods and Applications.* Springer, Berlin, Heidelberg, 1st edition, 1985.

[118] D. Hafner, P. Ochs, J. Weickert, M. Reißel, and S. Grewenig. FSI Schemes: Fast Semi-Iterative Solvers for PDEs and Optimisation Methods. In B. Rosenhahn and B. Andres, editors, *Pattern Recognition*, volume 9796 of *LNCS*, pages 91–102. Springer, Cham, 2016.

[119] E. Hairer, S. P. Nørsett, and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems.* Springer, Berlin, Heidelberg, 2nd edition, 1993.

[120] E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems.* Springer, Berlin, Heidelberg, 2nd edition, 1996.

[121] J. Han, A. Jentzen, and W. Ee. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34):8505–8510, 2018.

[122] D. Hartman and L. K. Mestha. A deep learning framework for model reduction of dynamical systems. In *Proceedings of the 2017 IEEE Conference on Control Technology and Applications*, pages 1917–1922, 2017.

[123] X. He, Q. Kong, and Z. Xiao. Fast Simulation Methods for Dynamic Heat Transfer through Building Envelope Based on Model-Order-Reduction. *Procedia Engineering*, 121:1764–1771, 2015.

[124] M. Heinkenschloss, T. Reis, and A. C. Antoulas. Balanced truncation model reduction for systems with inhomogeneous initial conditions. *Automatica*, 47(3):559 –564, 2011.

[125] P. J. Heres. *Robust and efficient Krylov subspace methods for Model Order Reduction*. PhD thesis, Department of Mathematics and Computer Science, Eindhoven University of Technology, 2005.

[126] P. Herholz, T. A. Davis, and M. Alexa. Localized solutions of sparse linear systems for geometry processing. *ACM Transactions on Graphics*, 36(6):183, 2017.

[127] P. Herholz, F. Haase, and M. Alexa. Diffusion Diagrams: Voronoi Cells and Centroids from Diffusion. *Computer Graphics Forum*, 36(2):163–175, 2017.

[128] M. R. Hestenes and E. Stiefel. Methods of Conjugate Gradients for Solving Linear Systems. *Journal of Research of the National Bureau of Standards*, 6(49):409–436, 1952.

[129] R. I. Hickson, S. I. Barry, G. N. Mercer, and H. S. Sidhu. Finite difference schemes for multilayer diffusion. *Mathematical and Computer Modelling*, 54(1-2):210–220, 2011.

[130] N. J. Higham. The Scaling and Squaring Method for the Matrix Exponential Revisited. *SIAM Journal on Matrix Analysis and Applications*, 26(4):1179–1193, 2005.

[131] K. Hildebrandt, C. Schulz, C. von Tycowicz, and K. Polthier. Interactive Spacetime Control of Deformable Objects. *ACM Transactions on Graphics*, 31(4):71, 2012.

[132] C. Himpe. emgr–The Empirical Gramian Framework. *Algorithms*, 11(7):91, 2018.

[133] M. Hinze and S. Volkwein. Proper Orthogonal Decomposition Surrogate Models for Nonlinear Dynamical Systems: Error Estimates and Suboptimal Control. In P. Benner, D. C. Sorensen, and V. Mehrmann, editors, *Dimension Reduction of Large-Scale Systems*, volume 45 of *LNCSE*, pages 261–306. Springer, Berlin, Heidelberg, 2005.

[134] M. Hochbruck and C. Lubich. On Krylov Subspace Approximations to the Matrix Exponential Operator. *SIAM Journal on Numerical Analysis*, 34(5):1911–1925, 1997.

[135] M. Hochbruck, C. Lubich, and H. Selhofer. Exponential Integrators for Large Systems of Differential Equations. *SIAM Journal on Scientific Computing*, 19(5):1552–1574, 1998.

[136] M. Hochbruck and A. Ostermann. Exponential Runge-Kutta methods for parabolic problems. *Applied Numerical Mathematics*, 53(2-4):323–339, 2005.

[137] M. Hochbruck and A. Ostermann. Exponential integrators. *Acta Numerica*, 19:209–286, 2010.

[138] W. Hundsdorfer and J. G. Verwer. *Numerical Solution of Time-Dependent Advection-Diffusion-Reaction Equations*. Springer, Berlin, Heidelberg, 1st edition, 2003.

[139] C. Iwamura, F. S. Costa, I. Sbarski, A. Easton, and N. Li. An efficient algebraic multigrid preconditioned conjugate gradient solver. *Computer Methods in Applied Mechanics and Engineering*, 192(20-21):2299–2318, 2003.

[140] O. Jadhav, E. B. Rudnyi, and T. Bechthold. Load Snapshot Based Nonlinear-Input Model Order Reduction of a Thermal Human Tissue Model. In G. Nicosia and R. Vittorio, editors, *The 12th International Conference on Scientific Computing in Electrical Engineering*. SCEE, 2018.

[141] D. L. James and D. K. Pai. DyRT: Dynamic Response Textures for Real Time Deformation Simulation with Graphics Hardware. *ACM Transactions on Graphics*, 21(3):582–585, 2002.

[142] R. Jeltsch and M. Torrilhon. Flexible Stability Domains for Explicit Runge-Kutta Methods. In R. Jeltsch, T.-T. Li, and I. H. Sloan, editors, *Some Topics in Industrial and Applied Mathematics*, volume 8 of *CAM*, pages 152–180. World Scientific Publishing, 2007.

[143] T. Kailath. *Linear systems*. Prentice-Hall Englewood Cliffs, N.J., 1980.

[144] M. Kamon, F. Wang, and J. White. Generating Nearly Optimally Compact Models from Krylov-Subspace Based Reduced-Order Models. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 47(4):239–248, 2000.

[145] C. T. Kelley. *Iterative Methods for Linear and Nonlinear Equations*. SIAM, 1995.

[146] C. T. Kelley. *Solving Nonlinear Equations with Newton's Method*. SIAM, 2003.

[147] G. Kerschen, J.-C. Golinval, A. F. Vakakis, and L. A. Bergman. The Method of Proper Orthogonal Decomposition for Dynamical Characterization and Order Reduction of Mechanical Systems: An Overview. *Nonlinear Dynamics*, 41(1):147–169, 2005.

[148] V. G. Kim, Y. Lipman, and T. Funkhouser. Blended Intrinsic Maps. *ACM Transactions on Graphics*, 30(4):79, 2011.

[149] G. P. Kirsten. Comparison of methods for solving Sylvester systems. Master's thesis, Department of Applied Mathematics, Stellenbosch University, South Africa, 2018.

[150] Q. Kong, X. He, and Y. Jiang. Fast simulation of dynamic heat transfer through building envelope via model order reduction. *Building Simulation*, 10(3):419–429, 2017.

[151] I. Koutis, G. L. Miller, and D. Tolliver. Combinatorial Preconditioners and Multilevel Solvers for Problems in Computer Vision and Image Processing. In G. Bebis et al., editors, *Advances in Visual Computing*, volume 5875 of *LNCS*, pages 1067–1078. Springer, Berlin, Heidelberg, 2009.

[152] P. Koutsovasilis and M. Beitelschmidt. Comparison of model reduction techniques for large mechanical systems - A study on an elastic rod. *Multibody System Dynamics*, 20(2):111–128, 2008.

[153] S. Koziel and A. Pietrenko-Dabrowska. Basics of Data-Driven Surrogate Modeling. In S. Koziel and A. Pietrenko-Dabrowska, editors, *Performance-Driven Surrogate Modeling of High-Frequency Structures*, pages 23–58. Springer, Cham, 2020.

[154] D. Krishnan, R. Fattal, and R. Szeliski. Efficient Preconditioning of Laplacian Matrices for Computer Graphics. *ACM Transactions on Graphics*, 32(4):142, 2013.

[155] K. Kunisch and S. Volkwein. Galerkin proper orthogonal decomposition methods for parabolic problems. *Numerische Mathematik*, 90(1):117–148, 2001.

[156] S. Lall, J. E. Marsden, and S. Glavaški. Empirical model reduction of controlled nonlinear systems. *IFAC Proceedings Volumes*, 32(2):2598–2603, 1999.

[157] S. Lall, J. E. Marsden, and S. Glavaški. A subspace approach to balanced truncation for model reduction of nonlinear control systems. *International Journal of Robust and Nonlinear Control*, 12(6):519–535, 2002.

[158] M. H. Langston, M. T. Harris, P.-D. Létourneau, R. Lethin, and J. R. Ezick. Combinatorial multigrid: Advanced preconditioners for ill-conditioned linear systems. In *Proceedings of the 2019 IEEE High Performance Extreme Computing Conference*, pages 1–7, 2019.

[159] A. J. Laub, M. T. Heath, C. C. Paige, and R. C. Ward. Computation of System Balancing Transformations and Other Applications of Simultaneous Diagonalization Algorithms. *IEEE Transactions on Automatic Control*, 32(2):115–122, 1987.

[160] V. I. Lebedev. Explicit difference schemes for solving stiff problems with a complex or separable spectrum. *Computational Mathematics and Mathematical Physics*, 40(12):1729–1740, 2000.

[161] R. J. LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*. SIAM, 2007.

[162] B. Lévy. Laplace-Beltrami Eigenfunctions Towards an algorithm that "understands" geometry. In *Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006*, 2006.

[163] J.-R. Li and J. White. Low Rank Solution of Lyapunov Equations. *SIAM Journal on Matrix Analysis and Applications*, 24(1):260–280, 2002.

[164] P. Li and W. Shi. Model Order Reduction of Linear Networks with Massive Ports via Frequency-Dependent Port Packing. In *Proceedings of the 43rd ACM/IEEE Annual Design Automation Conference*, pages 267–272. ACM, 2006.

[165] Y. C. Liang, H. P. Lee, S. P. Lim, W. Z. Lin, K. H. Lee, and C. G. Wu. Proper orthogonal decomposition and its applications - Part I: Theory. *Journal of Sound and Vibration*, 252(3):527–544, 2002.

[166] J. Liesen and Z. Strakoš. *Krylov Subspace Methods: Principles and Analysis*. Oxford University Press, 2013.

[167] B. Lohmann and B. Salimbahrami. Introduction to Krylov Subspace Methods in Model Order Reduction. Technical report, Institute of Automation, University of Bremen, Germany, 2003.

[168] Z. Long, Y. Lu, X. Ma, and B. Dong. PDE-Net: Learning PDEs from Data. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *PMLR*, pages 3208–3216. PMLR, 2018.

[169] O. Macchion, A. Suleng, M. Mork, and J. Šindler. Transient simulation of very large scale models using Krylov subspace-based model order reduction. In C. A. Brebbia and A. H.-D. Cheng, editors, *Boundary Elements and Other Mesh Reduction Methods XXXVII*, volume 57 of *WIT Transactions on Modelling and Simulation*, pages 177–188, 2014.

[170] C. C. Magruder and S. Gugercin. Model Reduction of Inhomogeneous Initial Conditions. Technical report, Department of Mathematics, Virginia Tech, Blacksburg, USA, 2012.

[171] A. Mang, T. A. Schütz, S. Becker, A. Toma, and T. M. Buzug. Cyclic Numerical Time Integration in Variational Non-Rigid Image Registration based on Quadratic Regularisation. In M. Goesele, T. Grosch, H. Theisel, K. Toennies, and B. Preim, editors, *Vision, Modeling and Visualization*, pages 143–150. The Eurographics Association, 2012.

[172] A. Mang, A. Toma, T. A. Schütz, S. Becker, T. M. Buzug, T. Eckey, C. Mohr, and D. Petersen. Biophysical modeling of brain tumor progression: From unconditionally stable explicit time integration to an inverse problem with parabolic PDE constraints for model calibration. *Medical physics*, 39(7):4444–4459, 2012.

[173] T. A. Manteuffel. An Incomplete Factorization Technique for Positive Definite Linear Systems. *Mathematics of Computation*, 34(150):473–497, 1980.

[174] J. Martín-Vaquero and B. Janssen. Second-order stabilized explicit Runge-Kutta methods for stiff problems. *Computer Physics Communications*, 180(10):1802–1810, 2009.

[175] J. Martín-Vaquero and A. Kleefeld. ESERK5: A fifth-order extrapolated stabilized explicit Runge-Kutta method. *Journal of Computational and Applied Mathematics*, 356:22–36, 2019.

[176] V. Mehrmann and T. Stykel. Balanced Truncation Model Reduction for Large-Scale Systems in Descriptor Form. In P. Benner, D. C. Sorensen, and V. Mehrmann, editors, *Dimension Reduction of Large-Scale Systems*, volume 45 of *LNCSE*, pages 83–115. Springer, Berlin, Heidelberg, 2005.

[177] J. A. Meijerink and H. A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric m-matrix. *Mathematics of Computation*, 31(137):148–162, 1977.

[178] A. Meister. *Numerik linearer Gleichungssysteme*. Springer Spektrum, 5th edition, 2015.

[179] S. Melzi, M. Ovsjanikov, G. Roffo, M. Cristani, and U. Castellani. Discrete Time Evolution Process Descriptor for Shape Analysis and Matching. *ACM Transactions on Graphics*, 37(1):4, 2018.

[180] S. Melzi, J. Ren, E. Rodolà, A. Sharma, P. Wonka, and M. Ovsjanikov. ZoomOut: Spectral Upsampling for Efficient Shape Correspondence. *ACM Transactions on Graphics*, 38(6):155, 2019.

[181] G. A. Meurant. *Computer solution of large linear systems*. Elsevier, North-Holland, 1st edition, 1999.

[182] C. D. Meyer, D. S. Balsara, and T. D. Aslam. A second-order accurate Super TimeStepping formulation for anisotropic thermal conduction. *Monthly Notices of the Royal Astronomical Society*, 422(3):2102–2115, 2012.

[183] C. D. Meyer, D. S. Balsara, and T. D. Aslam. A stabilized Runge-Kutta-Legendre method for explicit super-time-stepping of parabolic and mixed equations. *Journal of Computational Physics*, 257(Part A):594–626, 2014.

[184] M. Meyer, M. Desbrun, P. Schröder, and A. H. Barr. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In H.-C. Hege and K. Polthier, editors, *Visualization and Mathematics III*, pages 35–57. Springer, Berlin, Heidelberg, 2003.

[185] A. Mignone, G. Bodo, S. Massaglia, T. Matsakos, O. Tesileanu, C. Zanni, and A. Ferrari. PLUTO: A Numerical Code for Computational Astrophysics. *The Astrophysical Journal Supplement Series*, 170(1):228–242, 2007.

[186] B. Minchev and W. M. Wright. A review of exponential integrators for first order semi-linear problems. Technical report, Department of Mathematical Science, Norwegian University of Science and Technology, Trondheim, Norway, 2005.

[187] C. Moler and C. Van Loan. Nineteen Dubious Ways to Compute the Exponential of a Matrix, Twenty-Five Years Later. *SIAM Review*, 45(1):3–49, 2003.

[188] B. C. Moore. Principal Component Analysis in Linear Systems: Controllability, Observability, and Model Reduction. *IEEE Transactions on Automatic Control*, 26(1):17–32, 1981.

[189] K. W. Morton and D. F. Mayers. *Numerical Solution of Partial Differential Equations: An Introduction*. Cambridge University Press, 2nd edition, 2005.

[190] M. Müller. On the POD Method: An Abstract Investigation with Applications to Reduced-Order Modeling and Suboptimal Control. Master's thesis, Institute for Numerical and Applied Mathematics, University of Göttingen, Kassel, Germany, 2008.

[191] A. Nasikun, C. Brandt, and K. Hildebrandt. Fast Approximation of Laplace-Beltrami Eigenproblems. *Computer Graphics Forum*, 37(5):121–134, 2018.

[192] S.-B. Nouri. *Advanced Model-Order Reduction Techniques for Large-Scale Dynamical Systems*. PhD thesis, Department of Electronics, Carleton University, Ottawa, Canada, 2014.

[193] G. Obinata and B. D. O. Anderson. *Model Reduction for Control System Design*. Springer, London, 1st edition, 2001.

[194] A. Odabasioglu, M. Celik, and L. T. Pileggi. PRIMA: Passive Reduced-Order Interconnect Macromodeling Algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(8):645–654, 1998.

[195] S. O. Ojo, S. Grivet-Talocia, and M. Paggi. Model order reduction applied to heat conduction in photovoltaic modules. *Composite Structures*, 119:477–486, 2015.

[196] S. O'Sullivan. A class of high-order Runge-Kutta-Chebyshev stability polynomials. *Journal of Computational Physics*, 300:665–678, 2015.

[197] S. O'Sullivan. Factorized Runge-Kutta-Chebyshev Methods. *Journal of Physics: Conference Series*, 837:012020, 2017.

[198] S. O'Sullivan and T. P. Downes. An explicit scheme for multifluid magnetohydrodynamics. *Monthly Notices of the Royal Astronomical Society*, 366(4):1329–1336, 2006.

[199] M. Ovsjanikov, A. M. Bronstein, M. M. Bronstein, and L. G. Guibas. Shape Google: a computer vision approach to isometry invariant shape retrieval. In *Proceedings of the 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops 2009*, pages 320–327, 2009.

[200] H. K. F. Panzer. *Model Order Reduction by Krylov Subspace Methods with Global Error Bounds and Automatic Choice of Parameters*. PhD thesis, Faculty of Mechanical Engineering, Technical University of Munich, Germany, 2014.

[201] H. K. F. Panzer, T. Wolf, and B. Lohmann. A Strictly Dissipative State Space Representation of Second Order Systems. *at-Automatisierungstechnik*, 60(7):392–396, 2012.

[202] W. P. Pape. Temperatur des Grundwassers. In *Jahresbericht 2008 des Hessischen Landesamtes für Umwelt und Geologie*, pages 59–69. Hessisches Landesamt für Umwelt und Geologie, 2009.

[203] S. Parisotto. *Anisotropic variational models and PDEs for inverse imaging problems*. PhD thesis, Cambridge Centre for Analysis, University of Cambridge, United Kingdom, 2006.

[204] S. Parisotto, L. Calatroni, A. Bugeau, N. Papadakis, and C.-B. Schönlieb. Variational Osmosis for Non-Linear Image Fusion. *IEEE Transactions on Image Processing*, 29:5507–5516, 2020.

[205] S. Parisotto, L. Calatroni, M. Caliari, C.-B. Schönlieb, and J. Weickert. Anisotropic osmosis filtering for shadow removal in images. *Inverse Problems*, 35(5):054001, 2019.

[206] S. Parisotto, L. Calatroni, and C. Daffara. Efficient Osmosis Filtering of Thermal-Quasi Reflectography Images for Cultural Heritage. *arXiv e-print*, arXiv:1704.04052, 2017.

[207] B. N. Parlett. *The Symmetric Eigenvalue Problem*. SIAM, 1998.

[208] G. Patané. wFEM heat kernel: Discretization and applications to shape analysis and retrieval. *Computer Aided Geometric Design*, 30(3):276–295, 2013.

[209] G. Patané. Accurate and Efficient Computation of Laplacian Spectral Distances and Kernels. *Computer Graphics Forum*, 36(1):184–196, 2017.

[210] G. Patané. An Introduction to Laplacian Spectral Distances and Kernels: Theory, Computation, and Applications. In *ACM SIGGRAPH 2017 Courses*, 2017.

[211] D. W. Peaceman and J. H. H. Rachford. The Numerical Solution of Parabolic and Elliptic Differential Equations. *Journal of the Society for Industrial and Applied Mathematics*, 3(1):28–41, 1955.

[212] K. Pearson. LIII. On Lines and Planes of Closest Fit to Systems of Points in Space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[213] A. Pentland and J. Williams. Good Vibrations: Modal Dynamics for Graphics and Animation. *ACM Computer Graphics*, 23(3):207–214, 1989.

[214] F. H. Pereira, S. L. L. Verardi, and S. I. Nabeta. A fast algebraic multigrid preconditioned conjugate gradient solver. *Applied Mathematics and Computation*, 179(1):344–351, 2006.

[215] P. Pérez, M. Gangnet, and A. Blake. Poisson Image Editing. *ACM Transactions on Graphics*, 22(3):313–318, 2003.

[216] L. Pernebo and L. M. Silverman. Model Reduction via Balanced State Space Representations. *IEEE Transactions on Automatic Control*, 27(2):382–387, 1982.

[217] P. Perona and J. Malik. Scale-Space and Edge Detection Using Anisotropic Diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[218] J. Phillips, L. Daniel, and L. M. Silveira. Guaranteed Passive Balancing Transformations for Model Order Reduction. In *Proceedings of the 39th Annual Design Automation Conference*, pages 52–57. IEEE Computer Society Press, 2002.

[219] L. T. Pillage and R. A. Rohrer. Asymptotic Waveform Evaluation for Timing Analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 9(4):352–366, 1990.

[220] R. Pinnau. Model Reduction via Proper Orthogonal Decomposition. In W. H. A. Schilders, H. A. van der Vorst, and J. Rommes, editors, *Model Order Reduction: Theory, Research Aspects and Applications*, pages 95–109. Springer, Berlin, Heidelberg, 2008.

[221] S. Prajna. POD Model Reduction with Stability Guarantee. In *Proceedings of the 42nd IEEE International Conference on Decision and Control*, volume 5, pages 5254–5258, 2003.

[222] Z.-Q. Qu. *Model Order Reduction Techniques with Applications in Finite Element Analysis*. Springer, London, 1st edition, 2004.

[223] M. Raissi. Deep Hidden Physics Models: Deep Learning of Nonlinear Partial Differential Equations. *Journal of Machine Learning Research*, 19(25):1–24, 2018.

[224] M. Raissi and G. E. Karniadakis. Hidden physics models: Machine learning of nonlinear partial differential equations. *Journal of Computational Physics*, 357:125–141, 2018.

[225] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[226] M. Reuter, S. Biasotti, D. Giorgi, G. Patanè, and M. Spagnuolo. Discrete Laplace-Beltrami operators for shape analysis and segmentation. *Computers & Graphics*, 33(3):381–390, 2009.

[227] M. Reuter, F.-E. Wolter, and N. Peinecke. Laplace-Beltrami spectra as "Shape-DNA" of surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006.

[228] L. F. Richardson. IX. The Approximate Arithmetical Solution by Finite Differences of Physical Problems involving Differential Equations, with an Application to the Stresses in a Masonry Dam. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 210(459-470):307–357, 1910.

[229] L. F. Richardson and J. A. Gaunt. VIII. The Deferred Approach to the Limit. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 226(636-646):299–361, 1927.

[230] J. D. Roberts. Linear model reduction and solution of the algebraic Riccati equation by use of the sign function. *International Journal of Control*, 32(4):677–687, 1980.

[231] E. Rodolà, S. R. Bulo, T. Windheuser, M. Vestner, and D. Cremers. Dense Non-Rigid Shape Correspondence Using Random Forests. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 4177–4184, 2014. KIDS dataset freely available at `https://vision.in.tum.de/data/datasets/kids`.

[232] S. Rother and M. Beitelschmidt. Load Snapshot Decomposition to Consider Heat Radiation in Thermal Model Order Reduction. *IFAC-PapersOnLine*, 51(2):667–672, 2018.

[233] C. W. Rowley. Model reduction for fluids, using balanced proper orthogonal decomposition. *International Journal of Bifurcation and Chaos*, 15(03):997–1013, 2005.

[234] W. J. Rugh. *Linear system theory.* Prentice Hall Upper Saddle River, N.J., 2nd edition, 1996.

[235] R. M. Rustamov. Laplace-Beltrami Eigenfunctions for Deformation InvariantShape Representation. In A. Belyaev and M. Garland, editors, *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, pages 225–233. Eurographics Association, 2007.

[236] J. B. Rutzmoser. *Model Order Reduction for Nonlinear Structural Dynamics: Simulation-free Approaches*. PhD thesis, Faculty of Mechanical Engineering, Technical University of Munich, Germany, 2017.

[237] Y. Saad. Analysis of Some Krylov Subspace Approximations to the Matrix Exponential Operator. *SIAM Journal on Numerical Analysis*, 29(1):209–228, 1992.

[238] Y. Saad. *Iterative Methods For Sparse Linear Systems*. SIAM, 2nd edition, 2003.

[239] J. Saak. *Efficient Numerical Solution of Large Scale Algebraic Matrix Equations in PDE Control and Model Order Reduction*. PhD thesis, Faculty of Mathematics, Chemnitz University of Technology, Germany, 2009.

[240] J. Saak and P. Benner. Efficient solution of large scale Lyapunov and Riccati equations arising in model order reduction problems. *Proceedings in Applied Mathematics and Mechanics*, 8(1):10085–10088, 2008.

[241] K. Salah. A novel model order reduction technique based on artificial intelligence. *Microelectronics Journal*, 65:58–71, 2017.

[242] B. Salimbahrami and B. Lohmann. Krylov Subspace Methods in Linear Model Order Reduction: Introduction and Invariance Properties. Technical report, Institute of Automation, University of Bremen, Germany, 2002.

[243] B. Salimbahrami and B. Lohmann. Order reduction of large scale second-order systems using Krylov subspace methods. *Linear Algebra and its Applications*, 415(2-3):385–405, 2006.

[244] V. K. Saul'yev. *Integration of Equations of Parabolic Type by the Method of Nets*. Oxford, Pergamon Press, 1964.

[245] F. Sauvigny. *Partial Differential Equations 2: Functional Analytic Methods*. Springer, London, 2 edition, 2012.

[246] H. Schaeffer. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473(2197):20160446, 2017.

[247] W. E. Schiesser. *The Numerical Method of Lines: Integration of Partial Differential Equations*. Academic Press, 1991.

[248] W. H. A. Schilders. Introduction to Model Order Reduction. In W. H. A. Schilders, H. A. van der Vorst, and J. Rommes, editors, *Model Order Reduction: Theory, Research Aspects and Applications*, pages 3–32. Springer, Berlin, Heidelberg, 2008.

[249] T. Schneidereit and M. Breuß. Solving Ordinary Differential Equations using Artificial Neural Networks - A study on the solution variance. *Proceedings of the Conference Algoritmy*, pages 21–30, 2020.

[250] C.-B. Schönlieb. *Partial Differential Equation Methods for Image Inpainting*. Cambridge University Press, 1st edition, 2015.

[251] R. C. Selga, B. Lohmann, and R. Eid. Stability Preservation in Projection-based Model Order Reduction of Large Scale Systems. *European Journal of Control*, 18(2):122–32, 2012.

[252] Y. Shamash. Linear system reduction using Pade approximation to allow retention of dominant modes. *International Journal of Control*, 21(2):257–272, 1975.

[253] L. S. Shen and J. W. Ramsey. An investigation of transient, two-dimensional coupled heat and moisture flow in the soil surrounding a basement wall. *International Journal of Heat and Mass Transfer*, 31(7):1517–1527, 1988.

[254] Y. Shi, L. Li, and C. H. Liang. Multidomain pseudospectral time-domain algorithm based on super-time-stepping method. *IEE Proceedings - Microwaves, Antennas and Propagation*, 153(1):55–60, 2006.

[255] D. H. Shin, D. H. Kim, and M. S. Song. The steepest descent method and the conjugate gradient method for slightly non-symmetric, positive definite matrices. *Communications of the Korean Mathematical Society*, 9(2):439–448, 1994.

[256] S. Shokoohi, L. M. Silverman, and P. M. Van Dooren. Linear Time-Variable Systems: Balancing and Model Reduction. *IEEE Transactions on Automatic Control*, 28(8):810–822, 1983.

[257] R. B. Sidje. Expokit: A Software Package forComputing Matrix Exponentials. *ACM Transactions on Mathematical Software*, 24(1):130–156, 1998.

[258] E. D. Sifakis and J. Barbic. FEM Simulation of 3D Deformable Solids: A Practitioner's Guide to Theory, Discretization and Model Reduction. In *ACM SIGGRAPH 2012 Courses*, pages 20:1–20:50, 2012.

[259] L. M. Silveira, M. Kamon, I. Elfadel, and J. White. A Coordinate-Transformed Arnoldi Algorithm for Generating Guaranteed Stable Reduced-Order Models of RLC Circuits. In *Computer-Aided Design, International Conference on*, pages 288–294, 1996.

[260] V. Simoncini. A New Iterative Method for Solving Large-Scale Lyapunov Matrix Equations. *SIAM Journal on Scientific Computing*, 29(3):1268–1288, 2007.

[261] J. Šindler, A. Suleng, T. J. Olsen, and P. Bárta. Krylov Model Order Reduction of a Thermal Subsea Model. *International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, 7(5):842–849, 2013.

[262] J. Sirignano and K. Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.

[263] L. Sirovich. Turbulence and the dynamics of coherent structures part I: Coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987.

[264] G. D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*. Clarendon Press, Oxford, 3 edition, 1985.

[265] T. Ström. On Logarithmic Norms. *SIAM Journal on Numerical Analysis*, 12(5):741–753, 1975.

[266] T.-J. Su and R. R. Craig. Model Reduction and Control of Flexible Structures Using Krylov Vectors. *Journal of Guidance, Control, and Dynamics*, 14(2):260–267, 1991.

[267] J. Sun, M. Ovsjanikov, and L. Guibas. A Concise and Provably Informative Multi-Scale Signature Based on Heat Diffusion. In *Proceedings of the Symposium on Geometry Processing*, volume 28, pages 1383–1392. Eurographics Association, 2009.

[268] G. Söderlind. The logarithmic norm. History and modern theory. *BIT Numerical Mathematics*, 46(3):631–652, 2006.

[269] O. Tatebe. The Multigrid Preconditioned Conjugate Gradient Method. In N. D. Melson, S. F. McCormick, and T. A. Manteuffel, editors, *Sixth Copper Mountain Conference on Multigrid Methods*, pages 621–634. NASA, 1993.

[270] M. S. Tombs and I. Postlethwaite. Truncated balanced realization of a stable non-minimal state-space system. *International Journal of Control*, 46(4):1319–1330, 1987.

[271] M. Torrilhon and R. Jeltsch. Essentially optimal explicit Runge-Kutta methods with application to hyperbolic-parabolic equations. *Numerische Mathematik*, 106(2):303–334, 2007.

[272] L. N. Trefethen. Finite Difference and Spectral Methods for Ordinary and Partial Differential Equations. Cornell University, USA, 1996.

[273] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1st edition, 1997.

[274] A. Treuille, A. Lewis, and Z. Popović. Model Reduction for Real-Time Fluids. In *ACM SIGGRAPH 2006 Papers*, pages 826–834, 2006.

[275] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 1st edition, 2000.

[276] B. Unger. Impact of Discretization Techniques on Nonlinear Model Reduction and Analysis of the Structure of the POD Basis. Master's thesis, Department of Mathematics, Virginia Tech, Blacksburg, USA, 2013.

[277] B. Vaidya, D. Prasad, A. Mignone, P. Sharma, and L. Rickler. Scalable explicit implementation of anisotropic diffusion with Runge-Kutta-Legendre super-time-stepping. *Monthly Notices of the Royal Astronomical Society*, 472(3):3147–3160, 2017.

[278] J. van den Eshof and M. Hochbruck. Preconditioning Lanczos Approximations to the Matrix Exponential. *SIAM Journal on Scientific Computing*, 27(4):1438–1457, 2006.

[279] P. J. van der Houwen. Explicit Runge-Kutta Formulas with Increased Stability Boundaries. *Numerische Mathematik*, 20(2):149–164, 1972.

[280] P. J. van der Houwen. The development of Runge-Kutta methods for partial differential equations. *Applied Numerical Mathematics*, 20(3):261–272, 1996.

[281] P. J. van der Houwen and B. P. Sommeijer. On the Internal Stability of Explicit, m-Stage Runge-Kutta Methods for Large m-Values. *Journal of Applied Mathematics and Mechanics*, 60(10):479–485, 1980.

[282] O. van Kaick, H. Zhang, G. Hamarneh, and D. Cohen-Or. A Survey on Shape Correspondence. *Computer Graphics Forum*, 30(6):1681–1707, 2011.

[283] J. Van Lent. *Multigrid methods for time-dependent partial differential equations.* PhD thesis, Department of Electrical Engineering, Catholic University of Leuven, Leuven, Belgium, 2006.

[284] S. R. S. Varadhan. On the Behavior of the Fundamental Solution of the Heat Equation with Variable Coefficients. *Communications on Pure and Applied Mathematics*, 20(2):431–455, 1967.

[285] A. Vaxman, M. Ben-Chen, and C. Gotsman. A Multi-Resolution Approach to Heat Kernels on Discrete Surfaces. *ACM Transactions on Graphics*, 29(4):121, 2010.

[286] J. G. Verwer. Explicit Runge-Kutta methods for parabolic partial differential equations. *Applied Numerical Mathematics*, 22(1-3):359–379, 1996.

[287] J. G. Verwer, W. H. Hundsdorfer, and B. P. Sommeijer. Convergence Properties of the Runge-Kutta-Chebyshev Method. *Numerische Mathematik*, 57(1):157–178, 1990.

[288] M. Vestner, R. Litman, E. Rodolà, A. M. Bronstein, and D. Cremers. Product manifold filter: Non-rigid shape correspondence via kernel density estimation in the product space. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[289] O. Vogel, K. Hagenburg, J. Weickert, and S. Setzer. A Fully Discrete Theory for Linear Osmosis Filtering. In A. Kuijper, K. Bredies, T. Pock, and H. Bischof, editors, *Scale Space and Variational Methods in Computer Vision*, volume 7893 of *LNCS*, pages 368–379. Springer, Berlin, Heidelberg, 2013.

[290] C. von Tycowicz, C. Schulz, H.-P. Seidel, and K. Hildebrandt. An Efficient Construction of Reduced Deformable Objects. *ACM Transactions on Graphics*, 32(6):213, 2013.

[291] Z. Wang, D. Xiao, F. Fang, R. Govindan, C. C. Pain, and Y. Guo. Model identification of reduced order fluid dynamics systems using deep learning. *International Journal for Numerical Methods in Fluids*, 86(4):255–268, 2018.

[292] T. Washizawa. On the Behavior of the Residuals in Conjugate Gradient Method. *Applied Mathematics*, 1:211–214, 2010.

[293] J. Weickert. *Anisotropic Diffusion in Image Processing.* B.G. Teubner Stuttgart, 1st edition, 1998.

[294] J. Weickert, S. Grewenig, C. Schroers, and A. Bruhn. Cyclic Schemes for PDE-Based Image Analysis. *International Journal of Computer Vision*, 118(3):275–299, 2015.

[295] J. Weickert, K. Hagenburg, M. Breuß, and O. Vogel. Linear Osmosis Models for Visual Computing. In A. Heyden, F. Kahl, C. Olsson, M. Oskarsson, and X.-C. Tai, editors, *Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 8081 of *LNCS*, pages 26–39. Springer, Berlin, Heidelberg, 2013.

[296] J. Weickert, B. M. ter Haar Romeny, and M. A. Viergever. Efficient and Reliable Schemes for Nonlinear Diffusion Filtering. 7(3):398–410, 1998.

[297] J. Weickert, K. J. Zuiderveld, B. M. ter Haar Romeny, and W. J. Niessen. Parallel Implementations of AOS Schemes: A Fast Way of Nonlinear Diffusion Filtering. In *Proceedings of International Conference on Image Processing*, volume 3, pages 396–399. IEEE Computer Society Press, 1997.

[298] P. Wesseling. Introduction to multigrid methods. Technical report, Institute for Computer Applications in Science and Engineering, NASA, 1995.

[299] K. Willcox and J. Peraire. Balanced Model Reduction via the Proper Orthogonal Decomposition. *AIAA Journal*, 40(11):2323–2330, 2002.

[300] X. Xie, H. Zheng, S. Jonckheere, A. van de Walle, B. Pluymers, and W. Desmet. Adaptive model reduction technique for large-scale dynamical systems with frequency-dependent damping. *Computer Methods in Applied Mechanics and Engineering*, 332:363–381, 2018.

[301] F. Zhang and E. R. Hancock. Graph spectral image smoothing using the heat kernel. *Pattern Recognition*, 41(11):3328–3342, 2008.

[302] H. Zhang, O. van Kaick, and R. Dyer. Spectral Mesh Processing. *Computer Graphics Forum*, 29(6):1865–1894, 2010.