Towards Helicopter Simulation with an Index-1 Differential-Algebraic Equations System

Efficient Time Integration Methods

Elena Kremser

August 7, 2017

Master's thesis in Mathematics (M.Sc.)

University of Cologne
Mathematical Institute

First reviewer: Prof. Dr. Angela Kunoth
Second reviewer: Dr. rer. nat. Margrit Klitz
Technical advisor: Melven Röhrig-Zöllner

Abstract

Comprehensive helicopter simulation has been an important subject of research ever since the rise of the first helicopters. Numerous modeling approaches and software frameworks emerged, each with its individual advantages and drawbacks. However, a comprehensive approach that yields an efficient and effective simulation adaptable for research is still missing. Comprehensive here means that all relevant aspects are taken into account, e. g. vortex dynamics, rotor behavior, vibrations, etc. The German Aerospace Center (DLR) is working on a new solution using general state-space models for submodels of the helicopter which are coupled to a comprehensive model. This coupling yields an index-1 differential algebraic equation (DAE) system.

In this thesis, we analyze existing numerical algorithms for the solution of index-1 DAE systems and define a new familiy of methods that suits the challenges of helicopter simulation best. The challenges here consist in the interaction of large systems that are complex and individual in their behavior on their own. We focus our analysis on half-explicit Runge-Kutta (HERK) methods. Their explicit approach delivers the efficiency we seek. However, these methods are not able to handle stiff systems like the highly vibratory helicopter well. Stiff ordinary differential equation (ODE) problems can be solved by explicit exponential Runge-Kutta (EERK) methods. In order to apply them to index-1 DAE systems, we derive the new half-explicit exponential Runge-Kutta (HEERK) methods.

We test our HEERK methods in comparison to HERK methods on a mechanical model of the main rotor. The results show that HEERK methods need substantially fewer time steps than HERK methods for the same approximation quality. We also see that a good choice of submodels, where the stiff variables are solely placed in the state function, boosts this effect. So, HEERK methods constitute an essential step towards an effective real-time simulation in comprehensive rotorcraft simulation.

Acknowledgments

I thank Prof. Dr. Angela Kunoth, University of Cologne, and Dr. Achim Basermann, department leader of the High Performance Computing division of the German Aerospace Center (DLR) in Cologne, for their cooperation which made this Master's thesis possible in the first place.
A thank you also goes to my second reviewer Dr. Margrit Klitz (DLR) for her continuous support and improvement suggestions.
I also want to thank my technical advisor Melven Röhrig-Zöllner (DLR). His visions and ideas and our constant dialogue substantially contributed to the direction and the contents of my thesis.
A special thanks goes to my fiancé Jan Kohlwey who supports and encourages me no matter what.

Contents

Introduction

Today's helicopters are invaluable in emergency medical assistance, search and rescue, military missions and tourism. The first helicopters, as we know them today, emerged in the early 20^{th} century [AHMEC]. Compared to other aircraft, helicopters are of special interest because of their flight attributes: little space for take-off and landing, a vertical climb flight, flight is possible in all directions (forward, backward and lateral) and helicopters can even hover which makes their possible applications so manifold.

However, negative examples in history also show that it is important to study helicopters accurately. Helicopters like the SIKORSKY UH-60 or the SIKORSKY S76 showed unexpected unstable flight behavior after construction which had to be experimentally corrected resulting in high expenses [Bi09]. Furthermore, there are still numerous fatal accidents with helicopters due to unforeseen behavior of the helicopter in turbulent situations [HSAT16].

Due to these failures, the advantages of a sophisticated simulation tool for helicopters cannot be stressed enough: it provides a better understanding of the forces acting on the components of the helicopter in any flight situation and helps preventing accidents. Additionally, the simulation driven design of helicopters and their components renders the development of new models much less expensive.

Helicopter simulation codes have existed since the invention of computers. In the 1970's, the first generation of codes (C81, REXOR and others) emerged. They exhibited several limitations. Mostly, particular types of helicopters or only particular parts were simulated. Hence, the analysis was not comprehensive. The realization of these limitations then triggered a second generation of codes in the early 1980's (2GCHAS, RCAS, CAMRAD II amongst others). The new codes were characterized by a higher modifiability through separation of structural and aerodynamic models and a building-block structure [Jo13].

In spite of the variety of available codes, there is no solution that suits the requirements for research of the German Aerospace Center (DLR) well. Either the software is too expensive and not adaptable (CAMRAD II) or not sophisticated enough (HOST). Thus, the DLR works at designing its own solution for research. Hence, the project VAST – Versatile Aeromechanics Simulation Tool – was introduced as a cooperation of the Institute of Flight Systems (FT) at DLR Brunswick and the High Performance Computing department (HPC) of DLR Cologne. The aim of the project is an independent simulation code that is easily adaptable to related problems, that performs better than existing codes and that accurately predicts a helicopter's behavior – even in real time. In order to fulfill this challenging goal, the VAST software framework is built on two main pillars:

1. Modularity: the individual components of the entire model shall be easily interchangeable. Thus, changes concerning one part of the simulation can be implemented without changing

the others. This way, new designs of helicopters can be integrated smoothly in the framework improving their development while lowering the costs.

2. Complexity: the complex interdependencies of the individual components shall be considered as elaborately as possible. The modeling error when mapping the reality to physical equations can never be entirely eliminated. However, through the choice of more accurate numerical methods, the approximation of the arising systems can be improved.

Helicopters consist of several systems with different behavior that interact with each other. This is one of the main challenges that we have to take into account for their simulation. Additionally to extensive overview books like 'Helicopter Theory' [Jo12], 'Bramwell's Helicopter Dynamics' [Br01], 'Helicopter Flight Dynamics' [Pa07] and 'Basic Helicopter Aerodynamics' [SN11], there are numerous papers that investigate specific aspects of helicopters. To start with, the elastic blades of the rotor are among the most interesting parts of the helicopter. They are essential for the flight behavior and already form a quite sophisticated system themselves; see [BKMS17]. Most importantly, the helicopter has to be able to fly stable. Thus, the analysis of rotorcraft stability is an important topic [BN01, BW10, Fr86, Pe94]. Instabilities occur due to the interference of vortices with the rotor blades. These vortices are modeled by a Partial Differential Equation (PDE) system. Additionally, we need to take into account the fuselage of the helicopter. Even if, at first glance, these parts show very different behavior, they are not independent from each other. For instance, the vortices that are generated by the spinning of the blades introduce a downward force on the fuselage. Such interactions always have to be considered when modeling the system 'helicopter' as a whole [BK93].

As mathematicians, we tend to favor a symbolic system of equations which describes the system in its entirety. On the computing side, this would mean a monolithic model of aerodynamic and structural dynamic interactions by a set of partial differential equations that are simultaneously treated by a single solver. However, as pointed out in [Wa05] this approach lays our aim of modularity to waste: if we implement minor changes or improvements in the aerodynamic or structural solver this would require a complete update of the computer program. Furthermore, the derivation of these equations is not straightforward [Wa05]. Last but not least, our modular approach allows us to reuse already existing codes in the DLR. Therefore, we model the whole system as a collection of individual systems with coupling terms that establish the reciprocal physical influences and obtain an explicitly loosely coupled system; see [GSJJ13] for different coupling methods. The system we will work with has the general form

$$\left\{ \begin{array}{rcl} \dot{\mathbf{x}}(t) & = & f(t,\mathbf{x}(t),\mathbf{y}(t)) \\ \mathbf{y}(t) & = & g(t,\mathbf{x}(t),\mathbf{y}(t)) \end{array} \right. , \tag{$\star$}$$

where $\mathbf{x} : \mathbb{R}_+ \to \mathbb{R}^n$, $\mathbf{y} : \mathbb{R}_+ \to \mathbb{R}^m$, $f : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $g : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m$, $t \in \mathbb{R}_+$ and $m,n \in \mathbb{N}$. The system $(\star)$ is called general form of a state-space model. We assume that the

gradient $\frac{\partial g}{\partial \mathbf{y}}(t, \mathbf{x}(t), \mathbf{y}(t))$ is regular around the solution which makes the present system an index-1 differential-algebraic equation (DAE) system (cf. [HW96]).

This thesis focuses on the mathematical derivation of a suitable numerical method for the solution of a system ($\star$). The full specification of the actual equations is ongoing work in the DLR and goes beyond the scope of this thesis. However, we want to provide a preliminary assessment of existing and new algorithms for the simulation of general state-space models.

The main contributions of the author are the following:

- We conduct a detailed linear stability analysis of the first order explicit exponential Runge-Kutta (EERK) method. Although similar linear approaches have been considered in [CM02, MZ13, OP16, Zh17b], they do not provide a precise analysis of the stability function, which we do. Experimentally, we provide upper bounds for the time step size when applying the exponential Euler to an ODE with linear stiff and linear nonstiff terms. Additionally, we emphasize the advantages of the exponential Euler in comparison to the ordinary Euler for this setting.
- We create stability plots of EERK methods of orders 1–4 which are similar to the well-known stability plots for ordinary explicit Runge-Kutta (ERK) methods of orders 1–4. To the best of our knowledge, these plots are not present in the literature so far.
- We give a straight forward, comprehensible consistency analysis of half-explicit Runge-Kutta (HERK) methods of first and second order. Comparable results can be found in Arnold et al. [ASW93].
- Using the findings from our comprehensible consistency analysis of HERK methods, we newly define half-explicit exponential Runge-Kutta (HEERK) methods. These methods are very valuable in our setting, where we deal with a linear stiff part and a nonlinear nonstiff part in our ODE system. To the best of our knowledge, these methods have not been used before. For the first and second order HEERK methods we conduct a consistency analysis based on the comprehensible analysis of HERK methods for first and second orders.
- Last but not least, we highlight the advantages of HEERK methods in comparison to HERK methods by using a simplified stiff rotor model which exhibits a similar behavior to what we expect from a full helicopter system. Here, we use the mechanical model of Sanches et al. [SMBA11] as a starting point.

The remainder of this thesis is organized as follows: Section 1 contains a short introduction to helicopter simulation. We present our approach for the helicopter model and a test model that we will use for our numerical tests. In Section 2, we analyze suitable time integration approaches for our system. Here, we start with standard *Runge-Kutta methods* for the solution of ODEs (Subsection 2.1). Then we analyze so-called *half-explicit Runge-Kutta methods* for the solution of

index-1 DAEs (Subsection 2.2). Subsequently, we introduce *exponential Runge-Kutta methods* (Subsection 2.3) which work with a more advanced formulation of our system. In Subsection 2.4, we define *half-explicit exponential Runge-Kutta methods* in analogy to the common half-explicit Runge-Kutta methods of Subsection 2.2. Finally, in Section 3, we compare the described methods with respect to stability, consistency and run time. Some tests are executed on small problems, whereas the main result is obtained through a simulation with the simplified rotorcraft model from Section 1.2. We conclude this thesis with a summary and the numerous possibilities for further research.

1. Approach

1.1. Derivation of coupled index-1 DAE system

In contrast to analyzing specific interesting parts of a helicopter, we are interested in the helicopter's behavior as a whole. Here, we need to account for the complexity of the individual parts of the helicopter in order to obtain a comprehensive approach. Hence, we treat various subsystems individually. For each physical subsystem i, we consider a model

$$\begin{aligned} \dot{\mathbf{x}}_i(t) &= f_i(t, \mathbf{x}_i(t), \mathbf{u}_i(t)) \\ \mathbf{y}_i(t) &= g_i(t, \mathbf{x}_i(t), \mathbf{u}_i(t)) \end{aligned}, \tag{1.1.1}$$

where $t \in \mathbb{R}_+$ denotes the time variable, $\mathbf{x}_i \in \mathbb{R}^{n_i},\ n_i \in \mathbb{N}$ denotes the state vector of system i, $\mathbf{y}_i \in \mathbb{R}^{m_i},\ m_i \in \mathbb{N}$ denotes the output vector and $\mathbf{u}_i \in \mathbb{R}^{q_i},\ q_i \in \mathbb{N}$ denotes the input vector which consists of combinations of outputs of other subsystems. The functions $f_i : \mathbb{R}_+ \times \mathbb{R}^{n_i} \times \mathbb{R}^{q_i} \to \mathbb{R}^{n_i}$ and $g_i : \mathbb{R}_+ \times \mathbb{R}^{n_i} \times \mathbb{R}^{q_i} \to \mathbb{R}^{m_i}$ describe the behavior of model i. We only want to consider ordinary differential equations (ODEs). For models that contain partial differential equations (PDEs), we apply a discretization on the differential operators to obtain ODEs as well.

A combined model of all parts has the form

$$\begin{cases} \dot{\mathbf{x}}(t) = f(t, \mathbf{x}(t), \mathbf{y}(t)) \\ \mathbf{y}(t) = g(t, \mathbf{x}(t), \mathbf{y}(t)) \end{cases}, \tag{$\star$}$$

where $\mathbf{x} : \mathbb{R}_+ \to \mathbb{R}^n$, $\mathbf{y} : \mathbb{R}_+ \to \mathbb{R}^m$, $f : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $g : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m$, $n = \sum_i n_i$ and $m = \sum_i m_i$. Here, $\mathbf{x}$ denotes the *global state vector*, $\mathbf{y}$ denotes the *global output vector* and f and g arise when constructing the model by combining all f_i and g_i of (1.1.1), respectively. We will call the first part of this system the *dynamic part* and the second part will be called the *algebraic part*.

With the definition

$$\bar{g}(t, \mathbf{x}(t), \mathbf{y}(t)) := g(t, \mathbf{x}(t), \mathbf{y}(t)) - \mathbf{y}(t), \tag{1.1.2}$$

we can also write system ($\star$) as

$$\begin{cases} \dot{\mathbf{x}}(t) &= f(t, \mathbf{x}(t), \mathbf{y}(t)) \\ 0 &= \bar{g}(t, \mathbf{x}(t), \mathbf{y}(t)) \end{cases}. \tag{$\star\star$}$$

This form is often discussed in the literature, where it is either called semi-explicit DAE system (e. g. [BT99]) or index-1 DAE system (e. g. [ASW93, HLR89]). We will use the latter term.

Since our system reflects physical behavior, it is reasonable to assume that the implicit function theorem (see Theorem A.1.1) is applicable to $\bar{g}(t, \mathbf{x}(t), \mathbf{y}(t)) = 0$. This implies that the coupling

terms $\mathbf{y}(t) = g(t,\mathbf{x}(t),\mathbf{y}(t))$ or $\bar{g}(t,\mathbf{x}(t),\mathbf{y}(t)) = 0$ could be resolved for $\mathbf{y}$ as

$$\mathbf{y}(t) = \bar{g}^{-1}(0;t,\mathbf{x}(t)) =: G(t,\mathbf{x}). \tag{1.1.3}$$

The assumption that the implicit function theorem is applicable to $\bar{g}(t,\mathbf{x}(t),\mathbf{y}(t)) = 0$ requires $\frac{\partial g}{\partial \mathbf{y}}(t,\mathbf{x}(t),\mathbf{y}(t))$ to be regular near the solution which we will take as granted from here onwards.

Inserting (1.1.3) into the first part of ($\star$), $\dot{\mathbf{x}}(t) = f(t,\mathbf{x}(t),\mathbf{y}(t))$, we obtain an ordinary differential equations system

$$\dot{\mathbf{x}}(t) = f(t,\mathbf{x}(t),G(t,\mathbf{x}(t))) =: F(t,\mathbf{x}(t)). \tag{1.1.4}$$

This system, which is also called state-space form in [BT99, HW96], then is easily solvable with standard methods like Runge-Kutta methods. In Appendix B, we give an example, where we formulate a problem in a coupled index-1 form ($\star$) and in a state-space form (1.1.4) .

In general, the computation of the implicitly given function G is not trivial. Hence, we are interested in methods that deal with the specific formulation ($\star\star$). Furthermore, the methods that we consider should not contain derivatives of the functions f and g higher than first order: since we do not have a symbolic representation of ($\star$), an analytic gradient calculation is not possible and its approximation is expensive [GW08].

Additionally, we need consistent initial values at initial time t_0 for the system (cf. [HW96, BT99]), i. e.

$$\begin{aligned} \mathbf{x}(t_0) &= \mathbf{x}_0 \text{ with } \mathbf{x}_0 \in \mathbb{R}^n, \\ \mathbf{y}(t_0) &= \mathbf{y}_0 \text{ with } \mathbf{y}_0 \in \mathbb{R}^m, \\ 0 &= \bar{g}(t_0,\mathbf{x}_0,\mathbf{y}_0). \end{aligned}$$

As we only want to consider physical behavior, we will always assume that initial values of this kind are either completely prescribed or given t_0 and $\mathbf{x}_0$, we can be find $\mathbf{y}_0$ by $\mathbf{y}_0 = \bar{g}^{-1}(0;t_0,\mathbf{x}_0)$.

Since in helicopter simulation the blades of the helicopter spin fast and we need to simulate behavior over longer periods, we need a 'fast' time integration method. Therefore, we only consider explicit time integration schemes. For a good reproduction of the physical behavior, we need time steps that relate to 1 or 2 degrees of a revolution of the rotor (resulting in 360 or 180 time steps per revolution, respectively). Furthermore, we choose a fixed time step size, since a Fast Fourier Transform is applied to parts of the data subsequently.

The rotor blades can be modeled through parabolic PDEs, which we discretize in order to obtain our ODE system. However, this makes the system display stiff behavior even with our small time steps. Stiffness is a term that has no unique mathematical definition. It dates from a paper by Curtiss and Hirschfelder from 1952 [CH52]. Most generally, a system is stiff if it requires very small step sizes for a stable solution with explicit methods, while implicit methods deliver suitable

results with a much higher step size (cf. [Sp96] (Definition 2.1)). Several definitions of stiffness can be found in [La91].

Certainly, we will encounter problems when using explicit methods on a stiff system (cf. [La73]). In order to deal with the stiffness for our solver, we assume that it is possible to split every $f_i(t,\mathbf{x}_i(t),\mathbf{u}_i(t))$ into a linear stiff part $S_i \in \mathbb{R}^{n_i \times n_i}$ and a nonlinear nonstiff part $\tilde{f}_i(t,\mathbf{x}_i(t),\mathbf{u}_i(t))$. Combining all subsystems we obtain

$$\dot{\mathbf{x}}(t) = S\mathbf{x} + \tilde{f}(t,\mathbf{x}(t),\mathbf{y}(t)), \tag{1.1.5}$$

where $S \in \mathbb{R}^{n \times n}$ denotes a constant matrix with the matrices S_i on its diagonal, and $\tilde{f} : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ denotes a possibly nonlinear function that contains the nonstiff part of f, i. e. it has a sufficiently small Lipschitz constant [HO10]. The assumption that S contains the stiff part can either be given by the model or we can linearize f in a neighborhood of the solution. In total, we obtain a system

$$\left\{ \begin{array}{lcl} \dot{\mathbf{x}}(t) & = & S\mathbf{x} + \tilde{f}(t,\mathbf{x}(t),\mathbf{y}(t)) \\ \mathbf{y}(t) & = & g(t,\mathbf{x}(t),\mathbf{y}(t)) \end{array} \right. . \tag{$\star\star\star$}$$

1.2. Mechanical helicopter model

So far, we have obtained an impression of the challenges of helicopter simulation and we have presented our strategy for modeling such a system. In the following, we construct a model that represents a simplified part of the helicopter. For our model, we build on the work of Sanches et al. [SMBA11] who model a simplified rotor fixed to a fuselage (see Figure 1.2.1). The fuselage is considered to be a rigid body connected to a rotor hub with 4 blades which are represented by a concentrated mass located at a distance b from the point B [SMBA11].

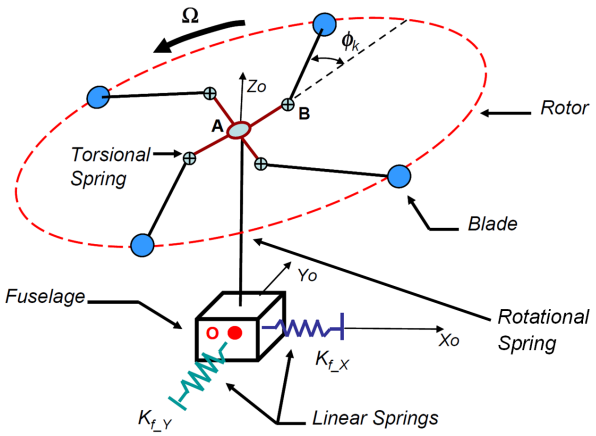


Figure 1.2.1: Simplified rotor fixed to a fuselage (adapted sketch from [SMBA11] (Figure 1))

We now present the model from [SMBA11] and show our approach in order to make their model a stiff coupled DAE model. For a more detailed derivation of our model and an explanation of all variables see Appendix C.

The governing equation of the system in [SMBA11] reads

$$M\ddot{\mathbf{u}}(t) + G\dot{\mathbf{u}}(t) + K\mathbf{u}(t) = F. \tag{1.2.2}$$

with

$$\mathbf{u}(t) = (x_{Fus}(t), y_{Fus}(t), \varphi_1(t), \varphi_2(t), \varphi_3(t), \varphi_4(t))^T \in \mathbb{R}^6 \tag{1.2.3}$$

and 6×6-matrices M, G and K and vector $F \in \mathbb{R}^6$ given in Subsection C.1. Here, $x_{Fus}(t)$ and $y_{Fus}(t)$ denote the longitudinal and transversal displacement of the fuselage, respectively, and $\varphi_i(t), i = 1, \ldots 4$ denotes the lead-lag angle of the i^{th} blade; cf. Equations (1) – (8) and Table 1 in [SMBA11] and Appendix C.

With

$$\mathbf{v}(t) := (\dot{x}_{Fus}(t), \dot{y}_{Fus}(t), \dot{\varphi}_1(t), \dot{\varphi}_2(t), \dot{\varphi}_3(t), \dot{\varphi}_4(t))^T \in \mathbb{R}^6, \tag{1.2.4}$$

we transform system (1.2.2) into a 12-dimensional first order system in the variables $\mathbf{u}(t)$ and $\mathbf{v}(t)$:

$$\begin{cases} \dot{\mathbf{u}}(t) = & \mathbf{v}(t) \\ \dot{\mathbf{v}}(t) = & -M^{-1}G\mathbf{v}(t) - M^{-1}K\mathbf{u}(t) + M^{-1}F. \end{cases} \tag{1.2.5}$$

In order to convert this model into a stiff system, we allow the connecting mast to have a small play provided by an additional spring at both ends of the mast. This yields a slightly more complex system that can be written as a coupled index-1 DAE system with the stiff part provided by the mast's behavior. We expect such a system to exhibit similar features as more complex systems for helicopters. We start with introducing the changes of the advanced model into the monolithic ODE model and state the index-1 DAE model afterwards.

We need additional constants and variables to model the occurring behavior. Let K_s denote the stiffness coefficient of the mast, and let $\varphi_{RH}(t)$ denote the rotational angle of the rotor head. Then we define

$$\begin{aligned} \omega_i(t) &:= \dot{\varphi}_i(t),\ i = 1, \ldots, 4, \\ \omega_{RH}(t) &:= \dot{\varphi}_{RH}(t), \\ \alpha_{RH}(t) &:= \dot{\omega}_{RH}(t), \\ \omega_s &:= \sqrt{\frac{K_s}{4(a+b)^2 m_b + 4I_{zb}}}, \end{aligned} \tag{1.2.6}$$

$$r_{sb} := \frac{b(a+b)m_b + I_{zb}}{4(a+b)^2 m_b + 4I_{zb}},$$

where $a,\ b,\ m_b$ and I_{zb} are constants of the model; see Table C.1.

Our state vector $\mathbf{u}(t)$ obtains the additional entry $\varphi_{RH}(t)$ and the matrices M, G and K and the vector F need to be adapted; see Subsection C.2. We denote the new matrices by M_{mon}, G_{mon} and K_{mon} and the new right side by F_{mon}.

With the adapted vector

$$\mathbf{u}_{mon}(t) = (x_{Fus}(t), y_{Fus}(t), \varphi_1(t), \varphi_2(t), \varphi_3(t), \varphi_4(t), \varphi_{RH}(t))^T \in \mathbb{R}^7, \tag{1.2.7}$$

the advanced system reads

$$M_{mon}\ddot{\mathbf{u}}_{mon}(t) + G_{mon}\dot{\mathbf{u}}_{mon}(t) + K_{mon}\mathbf{u}_{mon}(t) = F_{mon}. \tag{1.2.8}$$

With

$$\mathbf{v}_{mon}(t) = (\dot{x}_{Fus}(t), \dot{y}_{Fus}(t), \dot{\varphi}_1(t), \dot{\varphi}_2(t), \dot{\varphi}_3(t), \dot{\varphi}_4(t), \dot{\varphi}_{RH}(t))^T \in \mathbb{R}^7, \tag{1.2.9}$$

we transform system (1.2.8) into a first order system in the variables $\mathbf{u}_{mon}(t)$ and $\mathbf{v}_{mon}(t)$:

$$\begin{cases} \dot{\mathbf{u}}_{mon}(t) = & \mathbf{v}_{mon}(t) \\ \dot{\mathbf{v}}_{mon}(t) = & -M_{mon}^{-1}G_{mon}\mathbf{v}_{mon}(t) - M_{mon}^{-1}K_{mon}\mathbf{u}_{mon}(t) + M_{mon}^{-1}F_{mon}. \end{cases}$$

Defining

$$\mathbf{x}_{mon} := \begin{pmatrix} \mathbf{u}_{mon}(t) \\ \mathbf{v}_{mon}(t) \end{pmatrix} \tag{1.2.10}$$

with equations (1.2.7) and (1.2.9), we obtain a 14-dimensional linear monolithic system

$$\dot{\mathbf{x}}_{mon} = \begin{pmatrix} \mathbf{O}_{7\times 7} & \mathbf{I}_{7\times 7} \\ -M_{mon}^{-1}K_{mon} & -M_{mon}^{-1}G_{mon} \end{pmatrix} \cdot \mathbf{x}_{mon} + \begin{pmatrix} \mathbf{0}_7 \\ M_{mon}^{-1}F_{mon} \end{pmatrix} =: f_{mon}(t, \mathbf{x}_{mon}(t)), \tag{1.2.11}$$

where $\mathbf{O}_{7\times 7}$ denotes the 7×7-zero matrix, $\mathbf{I}_{7\times 7}$ denotes the 7×7-identity matrix and $\mathbf{0}_7$ denotes the 7-dimensional zero vector.

For the index-1 DAE model, we split the monolithic model into two submodels. The first submodel contains the original model while the second submodel mirrors the behavior of the mast. Again, the underlying matrices have to be adapted. The definitions of M_{DAE}, K_{DAE}, G_{DAE} and F_{DAE} are given in Subsection C.3.

With definitions (1.2.3) of $\mathbf{u}(t)$ and (1.2.4) of $\mathbf{v}(t)$, we obtain

$$\mathbf{x}_1(t) = \begin{pmatrix} \mathbf{u}(t) \\ \mathbf{v}(t) \end{pmatrix} = \begin{pmatrix} x_{Fus}(t) \\ y_{Fus}(t) \\ \varphi_1(t) \\ \varphi_2(t) \\ \varphi_3(t) \\ \varphi_4(t) \\ \dot{x}_{Fus}(t) \\ \dot{y}_{Fus}(t) \\ \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \\ \omega_4(t) \end{pmatrix} \in \mathbb{R}^{12} \tag{1.2.12}$$

as the state vector of the first model. The model has the inputs $\varphi_{RH}(t)$, $\omega_{RH}(t)$ and $\alpha_{RH}(t)$ and the outputs $\dot{\omega}_1(t), \dot{\omega}_2(t), \dot{\omega}_3(t)$ and $\dot{\omega}_4(t)$.

Similar to system (1.2.5), this yields

$$\dot{\mathbf{x}}_1(t) = f_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)) = \begin{pmatrix} \mathbf{O}_{6\times 6} & \mathbf{I}_{6\times 6} \\ -M_{DAE}^{-1} K_{DAE} & -M_{DAE}^{-1} G_{DAE} \end{pmatrix} \mathbf{x}_1(t) + \begin{pmatrix} \mathbf{0}_6 \\ M_{DAE}^{-1} F_{DAE} \end{pmatrix}, \tag{1.2.13}$$

where $\mathbf{O}_{6\times 6}$ denotes the 6×6-zero matrix, $\mathbf{I}_{6\times 6}$ denotes the 6×6-identity matrix and $\mathbf{0}_6$ denotes the 6-dimensional zero vector. The dependence on the inputs $\mathbf{y}_2(t) = (\varphi_{RH}(t), \omega_{RH}(t), \alpha_{RH}(t))^T$ is manifested in the definitions of M_{DAE}, K_{DAE}, G_{DAE} and F_{DAE}.

We see that the outputs of the first model $(\dot{\omega}_1(t), \dot{\omega}_2(t), \dot{\omega}_3(t), \dot{\omega}_4(t))$ are the time derivatives of the 9^{th} to 12^{th} entries of $\mathbf{x}_1(t)$. Hence, in order to state the algebraic function $g_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t))$ explicitly, we need to calculate $-M_{DAE}^{-1} K_{DAE}$ and $-M_{DAE}^{-1} G_{DAE}$. The particulars of the computation of $g_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t))$ are given in Subsection C.3.

From the first model we do not extract a stiff linear part. Hence,

$$S_1 := \mathbf{O}_{12\times 12} \tag{1.2.14}$$

and

$$\tilde{f}_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)) := f_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)). \tag{1.2.15}$$

The governing equation of the second model reads

$$\dot{\omega}_{RH}(t) = \omega_s^2(\Omega t - \varphi_{RH}(t)) - r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t). \tag{1.2.16}$$

Here, we have the inputs $\dot{\omega}_1(t), \dot{\omega}_2(t), \dot{\omega}_3(t), \dot{\omega}_4(t)$ and the outputs $\varphi_{RH}(t)$, $\omega_{RH}(t)$ and $\alpha_{RH}(t)$. With the local state vector

$$\mathbf{x}_2(t) := \begin{pmatrix} \varphi_{RH}(t) \\ \omega_{RH}(t) \end{pmatrix} \in \mathbb{R}^2, \tag{1.2.17}$$

equation (1.2.16) becomes

$$\dot{\mathbf{x}}_2(t) = \begin{pmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{pmatrix} \mathbf{x}_2(t) + \begin{pmatrix} 0 \\ \omega_s^2 \Omega t - r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t) \end{pmatrix}.$$

Since ω_s^2 (defined in (1.2.6)) is the variable which is mainly responsible for the stiffness of the system, we want it to be solely part of the first linear summand of the differential equations system. So far, it also appears in the second line of the above equation. Hence, we redefine our variables. Let

$$\begin{aligned} \mathbf{x}_2^{(1)} &:= \bar{\varphi}_{RH}(t) := \varphi_{RH}(t) - \Omega t, \\ \mathbf{x}_2^{(2)} &:= \bar{\omega}_{RH}(t) := \dot{\bar{\varphi}}_{RH}(t) = \bar{\varphi}_{RH}(t) - \Omega, \end{aligned} \tag{1.2.18}$$

which yields

$$\dot{\mathbf{x}}_2(t) = \begin{pmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{pmatrix} \mathbf{x}_2(t) + \begin{pmatrix} 0 \\ -r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t) \end{pmatrix}.$$

Accordingly, we define

$$S_2 := \begin{pmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{pmatrix} \tag{1.2.19}$$

and

$$\tilde{f}_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t)) := \begin{pmatrix} 0 \\ -r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t) \end{pmatrix}. \tag{1.2.20}$$

In this formulation, the variable ω_s is not part of the nonlinear part $\tilde{f}_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t))$.

The local output vector and hence the local output function $g_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t))$ is given by

$$\mathbf{y}_2(t) = \begin{pmatrix} \varphi_{RH}(t) \\ \omega_{RH}(t) \\ \alpha_{RH}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{x}_2^{(1)}(t) + \Omega t \\ \mathbf{x}_2^{(2)}(t) + \Omega \\ \dot{\mathbf{x}}_2^{(2)}(t) \end{pmatrix} =: g_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t)). \tag{1.2.21}$$

We can now collect the global model. Using the definitions of $\mathbf{x}_1(t)$ and $\mathbf{x}_2(t)$ in equations (1.2.12)

and (1.2.18), we obtain the global state vector

$$\mathbf{x}(t) = \begin{pmatrix} \mathbf{x}_1(t) \\ \mathbf{x}_2(t) \end{pmatrix} = \begin{pmatrix} x_{Fus}(t) \\ y_{Fus}(t) \\ \varphi_1(t) \\ \varphi_2(t) \\ \varphi_3(t) \\ \varphi_4(t) \\ \dot{x}_{Fus}(t) \\ \dot{y}_{Fus}(t) \\ \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \\ \omega_4(t) \\ \bar{\varphi}_{RH}(t) \\ \bar{\omega}_{RH}(t) \end{pmatrix}. \tag{1.2.22}$$

With the definitions of $\mathbf{y}_1(t)$ and $\mathbf{y}_2(t)$, our global output vector reads

$$\mathbf{y}(t) = \begin{pmatrix} \mathbf{y}_1(t) \\ \mathbf{y}_2(t) \end{pmatrix} = \begin{pmatrix} \dot{\omega}_1(t) \\ \dot{\omega}_2(t) \\ \dot{\omega}_3(t) \\ \dot{\omega}_4(t) \\ \varphi_{RH}(t) \\ \omega_{RH}(t) \\ \alpha_{RH}(t) \end{pmatrix}. \tag{1.2.23}$$

The global functions $f_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t))$ and $g_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t))$ arise from equations (1.2.14), (1.2.15), (1.2.19), (1.2.20), (C.3.5) and (1.2.21):

$$\begin{aligned} f_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t)) &:= S\mathbf{x}(t) + \tilde{f}_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t)) \qquad (1.2.24) \\ &= \begin{pmatrix} S_1 & \mathbf{O}_{12\times 2} \\ \mathbf{O}_{2\times 12} & S_2 \end{pmatrix} \mathbf{x}(t) + \begin{pmatrix} \tilde{f}_1(t, \mathbf{x}(t), \mathbf{y}(t)) \\ \tilde{f}_2(t, \mathbf{x}(t), \mathbf{y}(t)) \end{pmatrix}, \\ g_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t)) &:= \begin{pmatrix} g_1(t, \mathbf{x}(t), \mathbf{y}(t)) \\ g_2(t, \mathbf{x}(t), \mathbf{y}(t)) \end{pmatrix}. \qquad (1.2.25) \end{aligned}$$

Now we have the advanced stiff model in monolithic and in DAE representation. In the DAE representation, we extracted a stiff linear part and obtained a system of the form ($\star\star\star$). In Section 3.4, we use this model to evaluate the algorithm defined in Section 2.4.

2. Numerical methods for index-1 DAEs

In Section 1.1, we explained the approach that we take towards helicopter simulation: we model the governing equations as a coupled index-1 DAE system. In this section, we examine numerical methods that can be applied to our system.

There are two popular approaches for the numerical solution of index-1 differential algebraic equations: ε-embedding methods [BT99, HW96] and half-explicit Runge-Kutta methods [BT99, ASW93, HLR89], also called state space form methods [BT99, HW96].

In order to apply ε-embedding methods, we need to consider the singularly perturbed problem (SPP)

$$\begin{cases} \dot{\mathbf{x}}(t) &= f(t,\mathbf{x}(t),\mathbf{y}(t)) \\ \varepsilon\dot{\mathbf{y}}(t) &= \bar{g}(t,\mathbf{x}(t),\mathbf{y}(t)) \end{cases}, \tag{SPP}$$

where $0 < \varepsilon \ll 1$ ([BT99] (p. 5)). For $\varepsilon = 0$, we obtain system ($\star\star$). Despite ε, we have a system of ODEs which can be solved by slightly adapted standard methods; see [HW96] (pp. 374/5). However for small ε, the arising system is stiff which makes it necessary to use implicit solution schemes; see [BT99] (Section 2.2.2). Alternatively, implicit-explicit methods [Bo07] could be applied. When using fully implicit methods, we need to solve a generally nonlinear system of equations in every stage. This results in a significantly higher run time of the algorithms, which is not suitable for our needs.

Instead, half-explicit Runge-Kutta methods do not exhibit these limitations since they are explicit methods. Thus, we want to analyze them in more detail.

We start this section with introducing general Runge-Kutta methods in Subsection 2.1, where we briefly state well-known consistency and stability results. Subsequently, we present the derivation of half-explicit Runge-Kutta methods in Subsection 2.2. Here, we conduct a comprehensible consistency analysis of first and second order methods. In order to deal with the earlier described stiffness of our system, we introduce explicit exponential Runge-Kutta methods in Subsection 2.3 and transfer the findings for half-explicit Runge-Kutta methods on these exponential integrators in order to obtain half-explicit exponential Runge-Kutta methods in Subsection 2.4. An overview of these topics is given in Table 2.1.

		First order	Second order
ERK	Consistency	[HW96, DB08, DR06]	
	Stability		
HERK	Consistency	[ASW93] & Subsection 2.2.1	[ASW93] & Subsection 2.2.2
	Stability	translates from ERK	
EERK	Consistency	[HO10] & Subsection 2.3.3.1	[HO10] & Subsection 2.3.4
	Stability	Subsection 2.3.3.2	n.a.
HEERK	Consistency	Subsection 2.4.1	Subsection 2.4.2
	Stability	translates from EERK	n.a.

Table 2.1: Summary of Section 2

2.1. Explicit Runge-Kutta (ERK) methods

Runge-Kutta (RK) methods have now been used and advanced for over 100 years. Their evolution started in 1895 when Carl Runge published a paper proposing more elaborate schemes than the Euler method for the numerical solution of differential equations. The different stages that the development has taken are outlined in [Bu96] or [BW96].

Runge-Kutta methods are one-step methods for the numerical solution of an ODE

$$\begin{cases} \dot{\mathbf{x}}(t) &= f(t,\mathbf{x}(t)), \quad \mathbf{x} \in \mathbb{R}^n \\ \mathbf{x}(0) &= \mathbf{x}_0 \end{cases} . \tag{2.1.1}$$

Every step consists of s stages. Explicit Runge-Kutta (ERK) methods only use previous time step approximations, while for implicit methods, a generally nonlinear system of equations has to be solved; see [DB08] for a more detailed introduction to Runge-Kutta methods. In this work, we only deal with explicit methods since implicit methods are too costly for larger helicopter systems.

An explicit s-stage Runge-Kutta method is given by the Butcher-Tableau

$$\begin{array}{c|ccccc} 0 & 0 & & & & \\ c_2 & a_{21} & 0 & & & \\ c_3 & a_{31} & a_{32} & \ddots & & \\ \vdots & \vdots & \vdots & \ddots & 0 & \\ c_s & a_{s,1} & a_{s,2} & \dots & a_{s,s-1} & 0 \\ \hline & b_1 & b_2 & \dots & b_{s-1} & b_s \end{array} .$$

Here, a_{ij}, b_i and c_i are the coefficients of the method. They can be stored in a matrix $A := (a_{ij})_{i=1,\dots,s;j=1,\dots,s}$ and vectors $\mathbf{b} := (b_j)_{j=1,\dots,s}$ and $\mathbf{c} := (c_i)_{i=1,\dots,s}$. For a given simulation

end time T and a prescribed number of time steps N, the corresponding algorithm then reads

Algorithm 2.1.2: s-Stage Explicit Runge-Kutta Method

Input : $\mathbf{x}_0, T, N, f, A, \mathbf{b}, \mathbf{c}$.

Output: $\mathbf{x}$

$\mathbf{x} \leftarrow \mathbf{x}_0$

$h \leftarrow \frac{T}{N}$

for $\ell \leftarrow 1$ *to* $N+1$ **do**

  for $i \leftarrow 1$ *to* s **do**

    $\mathbf{k}_i \leftarrow f\Big(t + c_i \cdot h, \mathbf{x} + h \cdot \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j\Big)$;

  end

  $\mathbf{x} \leftarrow \mathbf{x} + h \cdot \sum_{j=1}^{s} b_j \cdot \mathbf{k}_j$;

  $t \leftarrow t + h$;

end

Here, h is the constant step width throughout the whole algorithm and the intermediate results $\mathbf{k}_i := \mathbf{k}_i(t_n, \mathbf{x}_n)$ for $i = 1, \ldots, s$ denote the internal stages of every time step. Here, $\mathbf{x}_n$ is the approximation of $\mathbf{x}(t_n)$ obtained by the algorithm and $t_n := t_0 + n \cdot h$ denotes the time after n time steps. For the analysis of Runge-Kutta methods we need the notion of the increment function.

Definition 2.1.3 (Increment function ([Ku13] (Equation (3.2.1)), cf. [DB08] (Lemma 4.4)))**.**
The main iteration of a Runge-Kutta method is given by

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h \cdot \sum_{j=1}^{s} b_j \cdot \mathbf{k}_j(t_n, \mathbf{x}_n) =: \mathbf{x}_n + h \cdot \Phi(t_n, \mathbf{x}_n).$$

The function $\Phi(t_n, \mathbf{x}_n)$ is called *increment function*.

In general, we are interested in algorithms that yield 'good' approximations. In order to quantify the quality of an algorithm, we need the definitions of consistency, convergence and stability and their connection.

Definition 2.1.4 (Consistency ([Ku13] (Definition 3.2.6), cf. [DB08]))**.**
Let

$$\tau(h) := \frac{\mathbf{x}(t_{n+1}) - \mathbf{x}_n}{h} - \Phi(t_n, \mathbf{x}_n)$$

denote the *local truncation error*.
A method is called *consistent* if

$$\lim_{h \to 0} \tau(h) = 0.$$

A method is called *consistent of order k* if

$$\tau(h) = O(h^k) \text{ for } h \to 0.$$

In contrary to the local truncation error for consistency, we need to analyze the global truncation error for convergence.

Definition 2.1.5 (Convergence ([Ku13]))**.**
Let $\mathbf{x}_h(t)$ denote the approximation of $\mathbf{x}(t)$ obtained by a one-step method. The method is called convergent if the *global truncation error*

$$\varepsilon(t,h) := \mathbf{x}_h(t) - \mathbf{x}(t)$$

converges to 0 for $h \to 0$.

In order to proof a consistency order k of Runge-Kutta methods, we can compare the Taylor expansion to k^{th} order of $\mathbf{x}'(t)$ with the Taylor expansion to k^{th} order of the increment function $\Phi(t,\mathbf{x})$. If their difference is of k^{th} order, then the method has consistency order k. However, this result is purely asymptotic. It does not say how small h needs to be for specific problems in order for the methods to converge of order k. To this end, we additionally need the notion of stability.

In order to analyze the stability of one-step methods applied to an ODE of the form (2.1.1), we consider a one-dimensional Dahlquist test equation

$$\dot{x}(t) = \lambda x(t) =: f(t,x(t)), \tag{2.1.6}$$

where $t \in \mathbb{R}_+$, $\lambda \in \mathbb{C}$, $x : \mathbb{R} \to \mathbb{R}$. We can bring our underlying method in the form

$$x_{n+1} = \phi(h\lambda)x_n \tag{2.1.7}$$

with a stability function ϕ. Then the method is stable in the region

$$\{z \in \mathbb{C} : |\phi(z)| \leq 1\},$$

hence, for those $h \in \mathbb{R}_+$ such that $|\phi(h\lambda)| \leq 1$ (cf. [HW96] (Chapter IV.2) or [DR06] (Section 11.9.3)).

This is a linear stability analysis and certainly has shortcomings. However, nonlinear stability theory is much more sophisticated (cf. [La91], Chapter 7), which makes the nonlinear stability results rare. Futhermore, the findings for the linear analysis have proven to be quite good estimates even for nonlinear systems [HR07].

In total, we obtain convergence, if a method is consistent and stable.

Lemma 2.1.8 (adapted from [La91] (Chapter 7) and [Ku13])**.**
If a method is consistent of order k and stable, then it is convergent of order k or short

$$\text{Consistency} + \text{Stability} \Leftrightarrow \text{Convergence}.$$

Remark 2.1.9 ([Ku13] (Theorem 3.3.5), [DR06] (Theorem 11.25))**.**
For the stability of explicit one-step methods it is sufficient to show a Lipschitz condition of the increment function in $\mathbf{x}$, i. e. it exists $M \in \mathbb{R}$, such that

$$||\Phi(t,\mathbf{x}) - \Phi(t,\tilde{\mathbf{x}})|| \leq M||\mathbf{x} - \tilde{\mathbf{x}}||, \quad \mathbf{x},\ \tilde{\mathbf{x}} \in \mathbb{R}^n.$$

2.1.1. Consistency

The derivation of consistency conditions for various orders of Runge-Kutta methods has been "an interesting challenge" [Bu96] since their formulation. Consistency conditions for Runge-Kutta methods of orders one to four can be found in [DB08]. We will not deal with this subject for ordinary Runge-Kutta methods in this thesis.

2.1.2. Stability

The analysis of stability in this subsection is based on [Fr08] (Chapter 10, pp. 58–59). See also [Bu96, DB08]. Starting from an autonomous one-dimensional ODE of the form

$$\dot{x}(t) = f(x(t)), \quad x \in \mathbb{R},$$

an s-stage Runge-Kutta method is given by the iteration

$$k_i = f\Big(x_n + h\sum_{j=1}^{s} a_{ij}k_j\Big), \quad i = 1,\ldots,s \tag{2.1.10a}$$

$$x_{n+1} = x_n + h\sum_{i=1}^{s} b_i k_i, \tag{2.1.10b}$$

where the coefficients a_{ij} and b_i can be stored in a matrix $A := (a_{ij})_{i=1,\ldots,s;j=1,\ldots,s}$ and a vector $\mathbf{b} := (b_i)_{i=1,\ldots,s}$. In order to determine the stability region of the method, we consider the Dahlquist test equation (2.1.6), which we insert into (2.1.10a) to obtain

$$k_i = \lambda\Big(x_n + h\sum_{j=1}^{s} a_{ij}k_j\Big).$$

With $\mathbf{k} = (k_1, \ldots, k_s)^T$ and $\mathbf{e} = (1, \ldots, 1)^T \in \mathbb{R}^s$, this transforms to

$$\begin{aligned} & \mathbf{k} = \lambda (x_n \mathbf{e} + hA\mathbf{k}) \\ \Leftrightarrow\ & \mathbf{k} = (I - h\lambda A)^{-1} \lambda x_n \mathbf{e}. \end{aligned} \tag{2.1.11}$$

In order to obtain an equation of the form (2.1.7) we insert (2.1.11) into (2.1.10b):

$$\begin{aligned} x_{n+1} &= x_n + h\mathbf{b}^T \mathbf{k} \\ &= x_n + h\mathbf{b}^T (I - h\lambda A)^{-1} \lambda x_n \mathbf{e} \\ &= \left[1 + h\lambda \mathbf{b}^T (I - h\lambda A)^{-1} \mathbf{e}\right] x_n. \end{aligned}$$

Hence, using $z := h\lambda$, the stability function reads

$$\phi(z) = 1 + z\mathbf{b}^T (I - zA)^{-1} \mathbf{e}. \tag{2.1.12}$$

Lemma 2.1.13 (cf. [Fr08] (Chapter 10, pp. 58–59))**.**
For an n-dimensional problem, the derivation of the stability function can be reduced to finding the stability function for n one-dimensional problems.

Proof.
We consider the problem

$$\dot{\mathbf{x}}(t) = L\mathbf{x}(t) =: f(\mathbf{x}(t)), \quad \mathbf{x} \in \mathbb{R}^n,\ L \in \mathbb{R}^{n \times n}. \tag{2.1.14}$$

Case 1: Let L be diagonalizable.
Let $U \in \mathbb{R}^{n \times n}$ contain all eigenvectors of L and let $\Lambda \in \mathbb{R}^{n \times n}$ have the eigenvalues $\lambda_i, i = 1, \ldots, n$ of L on its diagonal, such that $U^{-1} L U = \Lambda$. Define

$$\hat{\mathbf{x}}_n := U^{-1} \mathbf{x}_n \tag{2.1.15a}$$

$$\hat{\mathbf{k}}_i := U^{-1} \mathbf{k}_i, \quad i = 1, \ldots s \tag{2.1.15b}$$

and insert (2.1.15a) and (2.1.15b) into (2.1.10). For (2.1.10a), we obtain

$$\begin{aligned} \mathbf{k}_i \quad &= L\Big(\mathbf{x}_n + h \sum_{j=1}^{s} a_{ij} \mathbf{k}_j\Big) \\ \Leftrightarrow U\hat{\mathbf{k}}_i &= LU\hat{\mathbf{x}}_n + h \sum_{j=1}^{s} a_{ij} LU \hat{\mathbf{k}}_j \\ \Leftrightarrow \hat{\mathbf{k}}_i \quad &= U^{-1} LU \hat{\mathbf{x}}_n + h \sum_{j=1}^{s} a_{ij} U^{-1} LU \hat{\mathbf{k}}_j \end{aligned}$$

$$\Leftrightarrow \hat{\mathbf{k}}_i \quad = \Lambda\hat{\mathbf{x}}_n + h\sum_{j=1}^{s} a_{ij}\Lambda\hat{\mathbf{k}}_j.$$

Similarly, for (2.1.10b), we get

$$\begin{aligned}
\mathbf{x}_{n+1} \quad &= \mathbf{x}_n + h\sum_{i=1}^{s} b_i\mathbf{k}_i \\
\Leftrightarrow U\hat{\mathbf{x}}_{n+1} &= U\hat{\mathbf{x}}_n + h\sum_{i=1}^{s} b_i U\hat{\mathbf{k}}_i \\
\Leftrightarrow \hat{\mathbf{x}}_{n+1} \quad &= \hat{\mathbf{x}}_n + h\sum_{i=1}^{s} b_i\hat{\mathbf{k}}_i.
\end{aligned}$$

We obtain n independent problems, since for any $\ell = 1,\ldots,n$, the ℓ^{th} entries of $\hat{\mathbf{k}}_i$ and $\hat{\mathbf{x}}_{n+1}$ only depend on the ℓ^{th} entries of $\hat{\mathbf{x}}_n$ and all $\hat{\mathbf{k}}_j$, $j = 1,\ldots,s$, namely

$$\begin{aligned}
\hat{\mathbf{k}}_i^{(\ell)} &= \lambda_\ell\hat{\mathbf{x}}_n^{(\ell)} + h\sum_{j=1}^{s} a_{ij}\lambda_\ell\hat{\mathbf{k}}_j^{(\ell)} \\
\hat{\mathbf{x}}_{n+1}^{(\ell)} &= \hat{\mathbf{x}}_n^{(\ell)} + h\sum_{i=1}^{s} b_i\hat{\mathbf{k}}_i^{(\ell)}.
\end{aligned}$$

Case 2: Let L be non-diagonalizable.
For non-diagonalizable L the situation is more complex. We can bring L in Jordan normal form (see Lemma A.2.2). Then, certain results can be shown. A detailed analysis is given in [Sp98] (Chapters 4–6). □

If we now insert the Butcher-Tableau for classical Runge-Kutta methods into the stability function (2.1.12), we obtain the following results.

Corollary 2.1.16 (cf. [HW96] (Theorem 2.2))**.**
The stability function of an s-stage explicit Runge-Kutta method of consistency order k, with $k \leq s$, is given by

$$\phi(z) = \sum_{i=0}^{k} \frac{z^i}{i!}.$$

Corollary 2.1.17 (cf. [HW96] (Theorem 2.2))**.**
The stability functions $\phi_1(z)$ to $\phi_4(z)$ of the explicit Runge-Kutta methods of orders 1–4 are given by

$$\phi_1(z) = 1 + z, \tag{2.1.18a}$$

$$\phi_2(z) = 1 + z + \frac{1}{2}z^2, \tag{2.1.18b}$$

$$\phi_3(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3, \tag{2.1.18c}$$

$$\phi_4(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4. \tag{2.1.18d}$$

These methods are then stable in the regions $\{z \in \mathbb{C} : |\phi_i(z)| \leq 1\}$ for $i = 1, \ldots, 4$, which we visualize in Figure 2.1.19.

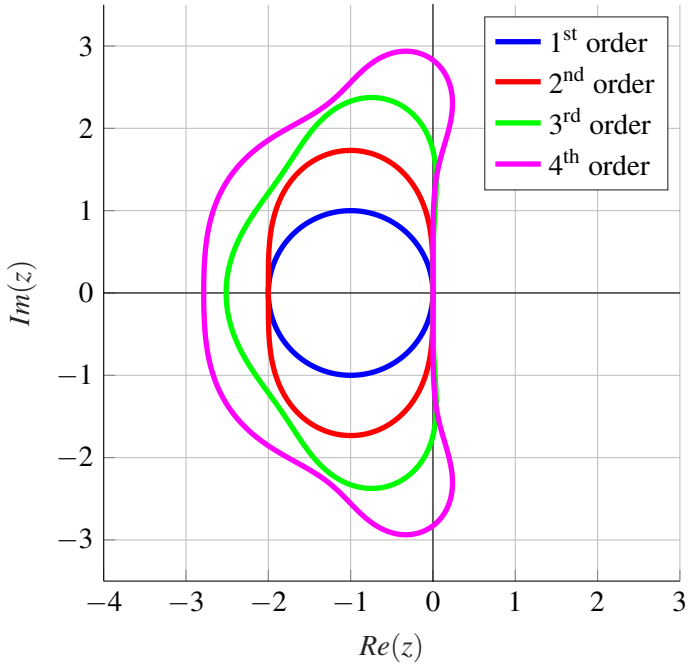


Figure 2.1.19: Stability regions of explicit Runge-Kutta methods of orders 1 to 4 (author's graphic, Matlab code combined from [RKM] and [Tr11], see also e. g. [Bu96] (Figure 1)).

2.2. Half-Explicit Runge-Kutta (HERK) methods

We presented explicit Runge-Kutta methods in the last subsection. Now we address the derivation of Half-Explicit Runge-Kutta (HERK) methods from ERK methods. Therefore, we consider our system

$$\left\{ \begin{array}{rcl} \dot{\mathbf{x}}(t) & = & f(t, \mathbf{x}(t), \mathbf{y}(t)) \\ 0 & = & \bar{g}(t, \mathbf{x}(t), \mathbf{y}(t)) \end{array} \right. \tag{$\star\star$}$$

that we strive to solve, intrinsically.

For the dynamic part, we want to use an explicit Runge-Kutta method. For the algebraic part, we need to find a root of the function $\bar{g}$. From (1.1.3) we know that $\mathbf{y}(t)$ is implicitly given by the solution of

$$\mathbf{y}(t) = \bar{g}^{-1}(0; t, \mathbf{x}(t)).$$

If the function $\bar{g}^{-1}$ is known, the s-stage method reads

Algorithm 2.2.1: s-Stage Half-Explicit Runge-Kutta Method

Input : $\mathbf{x}_0, \mathbf{y}_0, T, N, f, \bar{g}^{-1}, A, \mathbf{b}, \mathbf{c}$.

Output: $\mathbf{x}, \mathbf{y}$

1 $\mathbf{x} \leftarrow \mathbf{x}_0$

2 $\mathbf{y} \leftarrow \mathbf{y}_0$

3 $h \leftarrow \frac{T}{N}$

4 **for** $\ell \leftarrow 1$ *to* $N+1$ **do**

5 **for** $i \leftarrow 1$ *to* s **do**

6 $\mathbf{y} \leftarrow \bar{g}^{-1}\Big(0, t + c_i \cdot h, \mathbf{x} + h \cdot \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j\Big)$;

7 $\mathbf{k}_i \leftarrow f\Big(t + c_i \cdot h, \mathbf{x} + h \cdot \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j, \mathbf{y}\Big)$;

8 **end**

9 $\mathbf{x} \leftarrow \mathbf{x} + h \cdot \sum_{j=1}^{s} b_j \cdot \mathbf{k}_j$;

10 $t \leftarrow t + h$;

11 **end**

In this setting, we calculate the exact solution to

$$\mathbf{y} = g\Big(t + c_i \cdot h, \mathbf{x} + h \cdot \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j, \mathbf{y}\Big)$$

in line 6. In order to simplify the algorithm, we use a simplified Newton method for finding a cheap approximation of $\mathbf{y}$. For this purpose, we substitute line 6 by a simplified Newton call of the form

$$\mathbf{y} \leftarrow \text{Newton}\Big(t + c_i \cdot h, \mathbf{x} + h \cdot \sum_{j=1}^{i-1} a_{ij}\mathbf{k}_j, \mathbf{y}, g, \partial_{\mathbf{y}} g, \varepsilon(h)\Big),$$

where the Newton method is executed until an approximation of $\mathbf{y}$ with maximum defect $\varepsilon(h)$ is reached. For more details on Newton methods see Definitions A.3.1, A.3.2 and Remark A.3.3. For the remainder of the thesis we mean the simplified Newton method whenever we refer to a Newton method.

It should be noted that we need the gradient $\partial_{\mathbf{y}} g(t, \mathbf{x}, \mathbf{y})$ for the Newton call. As mentioned in the introduction, it is not trivial to obtain gradients in large systems that are not given symbolically. However, these gradients need to be approximated somehow in order to be able to apply our method. For our numerical tests in Section 3.4, we calculate the gradient analytically, since we have a symbolic representation of the system there.

Obviously, when evaluating the algebraic equation $0 = \bar{g}(t, \mathbf{x}(t), \mathbf{y}(t))$ exactly or solving it for $\mathbf{y}$

as in (1.1.4), we do not need to investigate the convergence orders of the underlying methods, since they are equivalent to the convergence orders of these methods for ODEs. However, for approximations of $\mathbf{y}(t) = g(t, \mathbf{x}(t), \mathbf{y}(t))$ e. g. by a Newton method, we need to know how many Newton steps we need in every iteration or to which accuracy we need to apply the Newton method in order to sustain the convergence order of the underlying Runge-Kutta method. The number of needed Newton steps is analyzed in [ASW93]. Instead here, we focus on the accuracy to which our output vector $\mathbf{y}$ needs to be computed by the Newton method. We investigate this question using the example of the first and second order half-explicit Runge-Kutta methods which are given in Algorithms 2.2.2 and 2.2.9. The respective findings of [ASW93] are given in Remarks 2.2.6 and 2.2.16.

2.2.1. First order consistency

The Butcher-Tableau of the only consistent first order explicit Runge-Kutta method (also known as the explicit Euler method) is given by

$$\begin{array}{c|c} 0 & \\ \hline & 1 \end{array}.$$

Combining the explicit Euler method with one Newton call per iteration yields a one-stage half-explicit Runge-Kutta method as in Algorithm 2.2.2.

Algorithm 2.2.2: One-Stage Half-Explicit Runge-Kutta Method

Input : $\mathbf{x}_0, \mathbf{y}_0, T, N, f, g, \partial_{\mathbf{y}} g$

Output: $\mathbf{x}, \mathbf{y}$

1 $\mathbf{x} \leftarrow \mathbf{x}_0$

2 $\mathbf{y} \leftarrow \mathbf{y}_0$

3 $h \leftarrow \frac{T}{N}$

4 **for** $\ell \leftarrow 1$ *to* $N+1$ **do**

5 $\quad \mathbf{y} \leftarrow \text{Newton}(\mathbf{x}, \mathbf{y}, g, \partial_{\mathbf{y}} g, \varepsilon(h))$;

6 $\quad \mathbf{k}_1 \leftarrow f(t, \mathbf{x}, \mathbf{y})$;

7 $\quad \mathbf{x} \leftarrow \mathbf{x} + h \cdot \mathbf{k}_1$;

8 $\quad t \leftarrow t + h$;

9 **end**

We now investigate the consistency of this algorithm.

Theorem 2.2.3 (Author's contribution)**.**
If the Newton method is solved to an accuracy of $O(h)$ in line 5 when we apply Algorithm 2.2.2 to $(\star)$, then the half-explicit Euler method is consistent of first order.

Proof.
We consider the system

$$\begin{cases} \dot{\mathbf{x}}(t) &= f(t,\mathbf{x}(t),\mathbf{y}(t)) \\ \mathbf{y}(t) &= g(t,\mathbf{x}(t),\mathbf{y}(t)) \end{cases}. \tag{$\star$}$$

Locally, we have a unique solution

$$\mathbf{y}(t) = \bar{g}^{-1}(0;t,\mathbf{x}(t))$$

as in (1.1.3).

However, we commit an error when applying the Newton method. Hence, we only get an inexact version $\tilde{g}^{-1}$ instead of $\bar{g}^{-1}$ for $\mathbf{y}$ and subsequently a defective $\tilde{\mathbf{y}}$, which has an error $\triangle\mathbf{y}$ in comparison to the correct $\mathbf{y}$, i. e.

$$\tilde{\mathbf{y}} := \tilde{g}^{-1}(0;t,\mathbf{x}) = \bar{g}^{-1}(0;t,\mathbf{x}) + \triangle\mathbf{y} = \mathbf{y} + \triangle\mathbf{y}.$$

In order to observe the consistency error, we need to compare the Taylor expansion of the difference operator with the Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$. First, we calculate the Taylor expansion of the difference operator up to first order

$$\begin{aligned} \mathbf{x}'(t) &= \frac{\mathbf{x}(t+h)-\mathbf{x}(t)}{h} \\ &= \mathbf{x}'(t) + O(h) \\ &= f(t,\mathbf{x},\mathbf{y}) + O(h). \end{aligned}$$

Hence,

$$\mathbf{x}'(t) = f(t,\mathbf{x},\mathbf{y}) + O(h). \tag{2.2.4}$$

The increment function $\Phi(t,\mathbf{x},\mathbf{y})$ reads

$$\Phi(t,\mathbf{x},\mathbf{y}) = \mathbf{k}_1(t,\mathbf{x},\tilde{\mathbf{y}}) = f(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}).$$

The corresponding Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$ under the assumption that $\triangle\mathbf{y}$ is at least of order $O(h)$ reads:

$$\Phi(t,\mathbf{x},\mathbf{y}) = f(t,\mathbf{x},\mathbf{y}) + \triangle\mathbf{y}\frac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y}) + O\big((\triangle\mathbf{y})^2\big).$$

So we obtain

$$\Phi(t,\mathbf{x},\mathbf{y}) = f(t,\mathbf{x},\mathbf{y}) + O(h). \tag{2.2.5}$$

The comparison of (2.2.4) and (2.2.5) yields the desired result. If $\triangle \mathbf{y} = O(h)$, then the method is consistent of first order. □

The respective result of Arnold et al. ([ASW93]) reads

Remark 2.2.6 ([ASW93] (Example 2.a))**.**
For a first order half-explicit Runge-Kutta method, at least one simplified Newton step is required to achieve the desired accuracy.

2.2.2. Second order consistency

The Butcher-Tableau of a second order ERK is given by

$$\begin{array}{c|cc} 0 & & \\ c_2 & a_{21} & \\ \hline & b_1 & b_2 \end{array}.$$

Corollary 2.2.7 ([DB08] (Lemma 4.16 & Theorem 4.18))**.**
A two-stage ERK method is consistent of second order if

$$\begin{aligned} b_1 + b_2 &= 1, \\ b_2 c_2 &= \tfrac{1}{2}, \\ b_2 a_{21} &= \tfrac{1}{2}. \end{aligned} \tag{2.2.8}$$

Now, we combine a two-stage Runge-Kutta method with Newton steps in each internal stage and obtain Algorithm 2.2.9.

Algorithm 2.2.9: Two-Stage Half-Explicit Runge-Kutta Method

Input : $\mathbf{x}_0, \mathbf{y}_0, T, N, f, g, \partial_{\mathbf{y}} g, a_{21}, b_1, b_2, c_2$
Output: $\mathbf{x}, \mathbf{y}$

$\mathbf{x} \leftarrow \mathbf{x}_0$
$\mathbf{y} \leftarrow \mathbf{y}_0$
$h \leftarrow \frac{T}{N}$
for $\ell \leftarrow 1$ *to* $N+1$ **do**
  $\mathbf{y} \leftarrow \text{Newton}(\mathbf{x}, \mathbf{y}, g, \partial_{\mathbf{y}} g, \varepsilon(h))$;
  $\mathbf{k}_1 \leftarrow f(t, \mathbf{x}, \mathbf{y})$;
  $\mathbf{y} \leftarrow \text{Newton}(t + c_2 h, \mathbf{x} + a_{21} h \cdot \mathbf{k}_1, \mathbf{y}, g, \partial_{\mathbf{y}} g, \varepsilon(h))$;
  $\mathbf{k}_2 \leftarrow f(t + c_2 h, \mathbf{x} + a_{21} h \cdot \mathbf{k}_1, \mathbf{y})$;
  $\mathbf{x} \leftarrow \mathbf{x} + h \cdot (b_1 \cdot \mathbf{k}_1 + b_2 \cdot \mathbf{k}_2)$;
  $t \leftarrow t + h$;
end

Theorem 2.2.10 (Author's contribution)**.**
If the order conditions (2.2.8) are fulfilled and the Newton method is solved to an accuracy of $O(h^2)$ in lines 5 and 7 when we apply Algorithm 2.2.9 to $(\star)$, then the half-explicit two-stage Runge-Kutta method is consistent of second order.

Proof.
We consider the system

$$\begin{cases} \dot{\mathbf{x}}(t) &= f(t,\mathbf{x}(t),\mathbf{y}(t)) \\ \mathbf{y}(t) &= g(t,\mathbf{x}(t),\mathbf{y}(t)) \end{cases} . \qquad (\star)$$

Locally, we have a unique solution

$$\mathbf{y}(t) = \bar{g}^{-1}(0;t,\mathbf{x}(t))$$

as in (1.1.3).

However, we commit an error when applying the Newton method. Hence, we only get an inexact version $\tilde{g}^{-1}$ instead of $\bar{g}^{-1}$ for $\mathbf{y}$ and subsequently a defective $\tilde{\mathbf{y}}_1$ in line 5, which has an error $\triangle\mathbf{y}_1$ in comparison to the correct $\mathbf{y}$, namely

$$\tilde{\mathbf{y}}_1 := \tilde{g}^{-1}(0;t,\mathbf{x}) = \bar{g}^{-1}(0;t,\mathbf{x}) + \triangle\mathbf{y}_1 = \mathbf{y} + \triangle\mathbf{y}_1.$$

For the Newton call in line 7, we already use our defective $\tilde{\mathbf{y}}_1$ from the first Newton call. We need to compute $\mathbf{y}$ corresponding to $t = t + c_2 h$ and $\mathbf{x} = \mathbf{x} + a_{21} h f(t,\mathbf{x},\tilde{\mathbf{y}}_1)$. We denote the occurring error by $\triangle\mathbf{y}_2$ and obtain the defective $\tilde{\mathbf{y}}_2$ as

$$\begin{aligned} \tilde{\mathbf{y}}_2 &:= \tilde{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\tilde{\mathbf{y}}_1)) \\ &= \bar{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\tilde{\mathbf{y}}_1)) + \triangle\mathbf{y}_2. \end{aligned}$$

In order to observe the consistency error, we need to compare the Taylor expansion of the difference operator with the Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$. First, we calculate the Taylor expansion of the difference operator up to second order:

$$\begin{aligned} \mathbf{x}'(t) &= \frac{\mathbf{x}(t+h)-\mathbf{x}(t)}{h} \\ &= \mathbf{x}'(t) + \tfrac{h}{2}\mathbf{x}''(t) + O(h^2) \\ &= f(t,\mathbf{x},\mathbf{y}) + \tfrac{h}{2}\left[\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial f}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y})\cdot f(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot \mathbf{y}'(t)\right] + O(h^2) \\ &= f(t,\mathbf{x},\mathbf{y}) + h\left[\tfrac{1}{2}\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y}) + \tfrac{1}{2}\tfrac{\partial f}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y})\cdot f(t,\mathbf{x},\mathbf{y}) + \tfrac{1}{2}\tfrac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot \mathbf{y}'(t)\right] + O(h^2). \end{aligned}$$

So we have

$$\mathbf{x}'(t) = f(t,\mathbf{x},\mathbf{y}) + h\left[\tfrac{1}{2}\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y}) + \tfrac{1}{2}\tfrac{\partial f}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y})\cdot f(t,\mathbf{x},\mathbf{y}) + \tfrac{1}{2}\tfrac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot \mathbf{y}'(t)\right] + O(h^2) \quad (2.2.11)$$

Here, we can further calculate $\mathbf{y}'(t)$ from (1.1.3) and obtain

$$\mathbf{y}'(t) = \tfrac{\partial \bar{g}^{-1}}{\partial t}(0;t,\mathbf{x}) + \tfrac{\partial \bar{g}^{-1}}{\partial \mathbf{x}}(0;t,\mathbf{x})\cdot \mathbf{x}'(t) = \tfrac{\partial \bar{g}^{-1}}{\partial t}(0;t,\mathbf{x}) + \tfrac{\partial \bar{g}^{-1}}{\partial \mathbf{x}}(0;t,\mathbf{x})\cdot f(t,\mathbf{x},\mathbf{y}). \quad (2.2.12)$$

The increment function $\Phi(t,\mathbf{x},\mathbf{y})$ reads

$$\begin{aligned}
\Phi(t,\mathbf{x},\mathbf{y}) &= b_1\mathbf{k}_1(t,\mathbf{x},\tilde{\mathbf{y}}_1) + b_2\mathbf{k}_2(t,\mathbf{x},\tilde{\mathbf{y}}_2)\\
&= b_1 f(t,\mathbf{x},\tilde{\mathbf{y}}_1) + b_2 f\big(t+c_2h, \mathbf{x}+a_{21}hf(t,\mathbf{x},\tilde{\mathbf{y}}_1), \tilde{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\tilde{\mathbf{y}}_1))\big)\\
&= b_1 f(t,\mathbf{x},\tilde{\mathbf{y}}_1)\\
&\quad + b_2 f\big(t+c_2h, \mathbf{x}+a_{21}hf(t,\mathbf{x},\tilde{\mathbf{y}}_1), \bar{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\tilde{\mathbf{y}}_1)) + \triangle\mathbf{y}_2\big)\\
&= b_1 f(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1) + b_2 f\big(t+c_2h, \mathbf{x}+a_{21}hf(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1),\\
&\quad \bar{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1)) + \triangle\mathbf{y}_2\big).
\end{aligned}$$

In order to simplify the notation in the Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$, we bring some calculations forward. Under the assumption that $\triangle\mathbf{y}_1$ and $\triangle\mathbf{y}_2$ are at least of order $O(h)$, we have

$$\begin{aligned}
h\cdot f(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1) &= h[f(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1 + O(\triangle\mathbf{y}_1^2)]\\
&= hf(t,\mathbf{x},\mathbf{y}) + h\cdot O(h)\tfrac{\partial f}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y}) + h\cdot O(h^2)\\
&= hf(t,\mathbf{x},\mathbf{y}) + O(h^2)\\
\Rightarrow h\cdot f(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1) &= hf(t,\mathbf{x},\mathbf{y}) + O(h^2) \qquad (2.2.13)
\end{aligned}$$

and

$$\begin{aligned}
\bar{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1)) &= \bar{g}^{-1}(0;t,\mathbf{x}) + c_2h\tfrac{\partial \bar{g}^{-1}}{\partial t}(0;t,\mathbf{x})\\
&\quad + \tfrac{\partial \bar{g}^{-1}}{\partial \mathbf{x}}(0;t,\mathbf{x})\cdot a_{21}hf(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1) + O(h^2)\\
&= \mathbf{y} + c_2h\tfrac{\partial \bar{g}^{-1}}{\partial t}(0;t,\mathbf{x})\\
&\quad + a_{21}h\tfrac{\partial \bar{g}^{-1}}{\partial \mathbf{x}}(0;t,\mathbf{x})f(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1) + O(h^2)\\
&\overset{(2.2.13)}{=} \mathbf{y} + c_2h\tfrac{\partial \bar{g}^{-1}}{\partial t}(0;t,\mathbf{x})\\
&\quad + a_{21}h\tfrac{\partial \bar{g}^{-1}}{\partial \mathbf{x}}(0;t,\mathbf{x})f(t,\mathbf{x},\mathbf{y}) + O(h^2),
\end{aligned}$$

which yields

$$\begin{array}{ll} \bar{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1)) & = \quad a_{21}h\frac{\partial\bar{g}^{-1}}{\partial\mathbf{x}}(0;t,\mathbf{x})f(t,\mathbf{x},\mathbf{y}) \\ +\triangle\mathbf{y}_2-\mathbf{y} & \quad +c_2h\frac{\partial\bar{g}^{-1}}{\partial t}(0;t,\mathbf{x})+\triangle\mathbf{y}_2+O(h^2). \end{array} \tag{2.2.14}$$

The corresponding Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$ under the assumptions that $\triangle\mathbf{y}_1$ and $\triangle\mathbf{y}_2$ are at least of order $O(h)$ and the conditions (2.2.8) are fulfilled reads

$$\begin{aligned} \Phi(t,\mathbf{x},\mathbf{y}) &= b_1\left(f(t,\mathbf{x},\mathbf{y})+\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1\right) \\ &\quad +b_2\Big[f(t,\mathbf{x},\mathbf{y})+c_2h\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+a_{21}h\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})f(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1) \\ &\quad +\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\left(\bar{g}^{-1}(0;t+c_2h,\mathbf{x}+a_{21}hf(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1))+\triangle\mathbf{y}_2-\mathbf{y}\right)\Big]+O(h^2) \\ &\overset{(2.2.14)}{=}(b_1+b_2)f(t,\mathbf{x},\mathbf{y})+b_2c_2h\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+b_2a_{21}h\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})f(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}_1) \\ &\quad +b_2\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\left(c_2h\tfrac{\partial\bar{g}^{-1}}{\partial t}(0;t,\mathbf{x})+\tfrac{\partial\bar{g}^{-1}}{\partial\mathbf{x}}(0;t,\mathbf{x})\cdot a_{21}hf(t,\mathbf{x},\mathbf{y})+\triangle\mathbf{y}_2\right) \\ &\quad +b_1\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1+O(h^2) \\ &\overset{(2.2.13)}{=}(b_1+b_2)f(t,\mathbf{x},\mathbf{y})+b_2c_2h\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+b_2a_{21}h\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})f(t,\mathbf{x},\mathbf{y}) \\ &\quad +b_2\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\left(c_2h\tfrac{\partial\bar{g}^{-1}}{\partial t}(0;t,\mathbf{x})+a_{21}h\tfrac{\partial\bar{g}^{-1}}{\partial\mathbf{x}}(0;t,\mathbf{x})f(t,\mathbf{x},\mathbf{y})+\triangle\mathbf{y}_2\right) \\ &\quad +b_1\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1+O(h^2) \\ &\overset{(2.2.8)}{=} f(t,\mathbf{x},\mathbf{y})+h\Big[\tfrac{1}{2}\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})f(t,\mathbf{x},\mathbf{y}) \\ &\quad +\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\left(\tfrac{\partial\bar{g}^{-1}}{\partial t}(0;t,\mathbf{x})+\tfrac{\partial\bar{g}^{-1}}{\partial\mathbf{x}}(0;t,\mathbf{x})f(t,\mathbf{x},\mathbf{y})\right)\Big] \\ &\quad +\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot(b_1\triangle\mathbf{y}_1+b_2\triangle\mathbf{y}_2)+O(h^2) \\ &\overset{(2.2.12)}{=}f(t,\mathbf{x},\mathbf{y})+h\Big[\tfrac{1}{2}\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})f(t,\mathbf{x},\mathbf{y})+\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\mathbf{y}'(t)\Big] \\ &\quad +\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot(b_1\triangle\mathbf{y}_1+b_2\triangle\mathbf{y}_2)+O(h^2). \end{aligned}$$

So, we have

$$\begin{aligned} \Phi(t,\mathbf{x},\mathbf{y}) = f(t,\mathbf{x},\mathbf{y})+h\Big[\tfrac{1}{2}\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})f(t,\mathbf{x},\mathbf{y})+\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\mathbf{y}'(t)\Big] \\ +\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot(b_1\triangle\mathbf{y}_1+b_2\triangle\mathbf{y}_2)+O(h^2). \end{aligned} \tag{2.2.15}$$

Subtracting (2.2.11) from (2.2.15), we obtain

$$\Phi(t,\mathbf{x},\mathbf{y})-\mathbf{x}'(t)=\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot(b_1\triangle\mathbf{y}_1+b_2\triangle\mathbf{y}_2)+O(h^2).$$

Hence, in order for the method to be consistent of second order, we need $\triangle\mathbf{y}_1+\triangle\mathbf{y}_2=O(h^2)$.

Now, if we solve both Newton methods to second order accuracy, we obtain $\triangle \mathbf{y}_1 = O(h^2)$ and $\triangle \mathbf{y}_2 = O(h^2)$ which yields $\triangle \mathbf{y}_1 + \triangle \mathbf{y}_2 = O(h^2)$. □

The respective result by [ASW93] reads

Remark 2.2.16 ([ASW93] (Example 2.b))**.**
For a second order half-explicit Runge-Kutta methods, at least one Newton step is required after the computation of $\mathbf{k}_1$ (line 7 in Algorithm 2.2.9) and also at least one Newton step is needed after the computation of the new $\mathbf{x}$ (which is equivalent to the Newton call in line 5 of Algorithm 2.2.9).

2.2.3. Note on stability

When applying half-explicit methods, we solve the algebraic part of ($\star$) in every iteration up to the accuracy that is equal to the convergence order that the method shall achieve. However, this means that only the error in the state variable $\mathbf{x}$ is propagated by the iteration of the method. Thus, we only need to analyze the stability properties of the explicit methods, which then translate to the respective half-explicit methods. For a more detailed analysis see [HLR89] (Theorem 3.1).

2.3. Explicit Exponential Runge-Kutta (EERK) methods

In Subsection 1.1, we stated that the dynamic part of our DAE system ($\star$) can be split into a linear stiff part and a nonlinear nonstiff part (cf. equation (1.1.5)). Additionally to fully implicit methods, Implicit-Explicit (IMEX) methods (also called linearly implicit methods) and exponential integrators are known for their ability to deal with this setting. IMEX methods are helpful whenever an ODE consists of a stiff part which needs to be integrated implicitly and a nonstiff part for which an explicit procedure is preferable (for more information on IMEX methods consult among others [ARW93, CFN01, Bo07, Ko08, BR09, BFR16, ZSB16]). For the application of exponential integrators the ODE needs to contain a linear part and a nonlinear part. Explicit methods require the nonlinear part to not display stiff behavior, whereas the linear part may be stiff. Since we have a linear stiff part in our setting, we want to focus on explicit exponential integrators on the basis of the work of Hochbruck and Ostermann [HO05a, HO05b, HO10].

General exponential methods have been studied since the middle of the 20^{th} century, whereas exponential Runge-Kutta methods evolved in the 1970's; see [HO10] (Section 6) for more detailed historical remarks. Originally, these methods were not considered to be efficient, since the exponential of a matrix has to be calculated. With the evolution of more efficient techniques for computing a matrix exponential, the exponential integrators have been revisited [Zh17b] and are a subject of recent research for applications e. g. in biochemistry [Zh17a] and ecology [OP16].

In this subsection, we first derive Explicit Exponential Runge-Kutta (EERK) methods. We give an overview of consistency in Subsection 2.3.1 and supply approaches to compute the numerical stability of these methods in Subsection 2.3.2. For first and second order methods, we investigate the consistency and for the first order methods we additionally examine stability properties in more detail (Subsections 2.3.3 and 2.3.4). Finally, we supply stability region plots for EERK methods of first to fourth order in Subsection 2.3.5.

We consider the ODE

$$\begin{cases} \dot{\mathbf{x}}(t) &= S\mathbf{x}(t) + \tilde{f}(t, \mathbf{x}(t)) \\ \mathbf{x}(t_0) &= \mathbf{x}_0 \end{cases} \tag{2.3.1}$$

similar to (1.1.5).

For an analytic solution of this problem, we would start with the linear homogeneous problem

$$\dot{\mathbf{x}}(t) = S\mathbf{x}(t),$$

the solution of which is given by

$$\mathbf{x}(t) = e^{St} \cdot \mathbf{C}, \quad \mathbf{C} := \mathbf{C}(t) \in \mathbb{R}^n.$$

Here, the matrix exponential e^{St} is defined as in Lemma A.2.3. The variation of the constant then provides a solution for the inhomogeneous problem. Inserting $\mathbf{x}(t) = e^{St} \cdot \mathbf{C}(t)$ into (2.3.1), we obtain

$$\begin{aligned} & Se^{St} \cdot \mathbf{C}(t) + e^{St} \cdot \mathbf{C}'(t) &&= Se^{St} \cdot \mathbf{C}(t) + \tilde{f}(t, \mathbf{x}(t)) \\ \Leftrightarrow \quad & e^{St} \cdot \mathbf{C}'(t) &&= \tilde{f}(t, \mathbf{x}(t)) \\ \Leftrightarrow \quad & \mathbf{C}'(t) &&= e^{-St} \cdot \tilde{f}(t, \mathbf{x}(t)). \end{aligned}$$

Since $\mathbf{x}(t_0) = e^{St_0} \cdot \mathbf{C}(t_0) \stackrel{!}{=} \mathbf{x}_0$, $\mathbf{C}(t)$ additionally has to fulfill the initial condition

$$\mathbf{C}(t_0) = e^{-St_0}\mathbf{x}_0.$$

In integral form, we can express $\mathbf{C}(t)$ as

$$\mathbf{C}(t) = \int_{t_0}^{t} e^{-S\tau} \cdot \tilde{f}(\tau, \mathbf{x}(\tau)) d\tau + e^{-St_0}\mathbf{x}_0.$$

Altogether, we obtain

$$\mathbf{x}(t) = e^{St} \cdot \left(\int_{t_0}^{t} e^{-S\tau} \cdot \tilde{f}(\tau, \mathbf{x}(\tau)) d\tau + e^{-St_0}\mathbf{x}_0 \right) = \int_{t_0}^{t} e^{S(t-\tau)} \cdot \tilde{f}(\tau, \mathbf{x}(\tau)) d\tau + e^{S(t-t_0)}\mathbf{x}_0.$$

for the analytical solution $\mathbf{x}(t)$.

Choosing $t_0 = t_n,\ \mathbf{x}_0 = \mathbf{x}_n = \mathbf{x}(t_n)$ and $t = t_n + h =: t_{n+1}$ (then $t - t_0 = t_n + h - t_n = h$), we obtain

$$\mathbf{x}(t_n + h) = \int_{t_n}^{t_n+h} e^{S(t_n+h-\tau)} \cdot \tilde{f}(\tau, \mathbf{x}(\tau)) d\tau + e^{Sh}\mathbf{x}_n.$$

Substituting $\bar{\tau}(\tau) = \tau - t_n$ and considering

- $\frac{d\bar{\tau}}{d\tau} = 1 \Leftrightarrow d\bar{\tau} = d\tau$,
- $\tau = \bar{\tau} + t_n$,
- $\bar{\tau}(t_n) = 0$ and $\bar{\tau}(t_n + h) = h$ for the integration limits,

we obtain the solution at time $t_{n+1} := t_n + h$ based on the prior solution $\mathbf{x}_n = \mathbf{x}(t_n)$ as

$$\mathbf{x}(t_n + h) = \int_0^h e^{S(h-\bar{\tau})} \cdot \tilde{f}(\bar{\tau} + t_n, \mathbf{x}(\bar{\tau} + t_n)) d\bar{\tau} + e^{Sh}\mathbf{x}_n.$$

When approximating the integral by quadrature rules, we obtain

$$\mathbf{x}(t_n + h) = \mathbf{x}_{n+1} = e^{hS}\mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)\tilde{f}(t_n + c_i h, \mathbf{x}(t_n + c_i h)) \tag{2.3.2}$$

with weights

$$\mathbf{b}_i(hS) = \int_0^1 e^{h(1-\theta)S}\ell_i(\theta) d\theta \in \mathbb{R}^{n\times n},$$

where

$$\ell_i(\theta) = \prod_{\substack{m=1 \\ m\neq i}}^{s} \frac{\theta - c_m}{c_i - c_m}, i = 1, \ldots, s$$

are the Lagrange interpolation polynomials. Since the unknown function $\mathbf{x}$ appears in (2.3.2), we further need to approximate $\mathbf{x}(t_n + c_i h)$ using the relation (2.3.2) once again with $t_n + c_i h$ instead of $t_n + h$ resulting in

$$\mathbf{x}(t_n + c_i h) \approx e^{c_i hS}\mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hS)\widetilde{F}_{nj} =: X_{ni},$$

where the $\widetilde{F}_{nj}$ are approximations of $\tilde{f}(t_n + c_j h, \mathbf{x}(t_n + c_j h))$. Hence

$$\widetilde{F}_{nj} = \tilde{f}(t_n + c_j h, X_{nj}).$$

Altogether, this yields the general form of the exponential Runge-Kutta methods

$$\mathbf{x}_{n+1} = \chi(hS)\mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)\widetilde{F}_{ni}, \tag{2.3.3a}$$

$$X_{ni} = \chi_i(hS)\mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hS)\widetilde{F}_{nj}, \tag{2.3.3b}$$

$$\widetilde{F}_{nj} = \tilde{f}(t_n + c_j h, X_{nj}), \tag{2.3.3c}$$

where

$$\begin{aligned} \sum_{i=1}^{s} \mathbf{b}_i(z) &= \frac{\chi(z)-1}{z} \\ \sum_{j=1}^{s} \mathbf{a}_{ij}(z) &= \frac{\chi_i(z)-1}{z}, i = 1,\dots,s \end{aligned} \tag{2.3.4}$$

and the coefficients χ and χ_i are given by

$$\begin{aligned} \chi(z) &= e^z, \\ \chi_i(z) &= e^{c_i z}, i = 1,\dots,s. \end{aligned} \tag{2.3.5}$$

The respective Butcher-Tableau to this explicit exponential method is

$$\begin{array}{c|cccc|c} c_1 & 0 & & & & \chi_1(hS) \\ c_2 & \mathbf{a}_{21}(hS) & & & & \chi_2(hS) \\ \vdots & \vdots & \ddots & \ddots & & \vdots \\ c_s & \mathbf{a}_{s,1}(hS) & \dots & \mathbf{a}_{s,s-1}(hS) & & \chi_s(hS) \\ \hline & \mathbf{b}_1(hS) & \dots & \mathbf{b}_{s-1}(hS) & \mathbf{b}_s(hS) & \chi(hS) \end{array} .$$

Here $c_1,\dots,c_s$ denote the nodes in time (as for usual RK methods). The matrices $\chi_i(hS)$ are introduced to accommodate for the linear stiff part. They all become the identity matrix for $S = 0$; see equations (2.3.5). A further difference to ordinary Runge-Kutta methods are the coefficients $\mathbf{a}_{ij}(hS)$ and $\mathbf{b}_j(hS)$. These are $n \times n$-matrices and not scalar anymore. Furthermore, these matrices depend on the stiff matrix S and on the step size h. However, for a fixed time step size, they only need to be computed at the beginning of the execution of an algorithm.

For $S \to 0$, the coefficient matrices $\mathbf{a}_{ij}(hS)$ and $\mathbf{b}_j(hS)$ converge to a diagonal matrix with identical values on the diagonal. The resulting method is equivalent to an ordinary Runge-Kutta method with coefficients b_i equal to one diagonal entry of $\mathbf{b}_j(0)$ and a_{ij} equal to one diagonal entry of $\mathbf{a}_{ij}(0)$. This resulting method is called the *underlying Runge-Kutta method* (cf. [HO10] (p. 220)).

Additionally for explicit methods,

$$\chi_1(z) = 1 \text{ and } \mathbf{a}_{ij}(z) = 0, 1 \leq i \leq j \leq s$$

have to hold.

When solving the conditions (2.3.4) for the coefficients $\chi(hS)$ and $\chi_i(hS)$ as

$$\chi(hS) = 1 + \sum_{i=1}^{s} \mathbf{b}_i(hS) \cdot hS$$
$$\chi_i(hS) = 1 + \sum_{j=1}^{s} \mathbf{a}_{ij}(hS) \cdot hS, i = 1, \ldots, s$$

and inserting them into (2.3.3a)

$$\begin{aligned}
\mathbf{x}_{n+1} &= \chi(hS)\mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)\widetilde{F}_{ni} \\
&= \Big(1 + \sum_{i=1}^{s} \mathbf{b}_i(hS) \cdot hS\Big)\mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)\widetilde{F}_{ni} \\
&= \mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)(\widetilde{F}_{ni} + S\mathbf{x}_n)
\end{aligned}$$

and (2.3.3b)

$$\begin{aligned}
X_{ni} &= \chi_i(hS)\mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hS)\widetilde{F}_{nj} \\
&= \Big(1 + \sum_{j=1}^{s} \mathbf{a}_{ij}(hS) \cdot hS\Big)\mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hS)\widetilde{F}_{nj} \\
&= \mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hS)(\widetilde{F}_{ni} + S\mathbf{x}_n),
\end{aligned}$$

we obtain a simpler formulation

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)(\widetilde{F}_{ni} + S\mathbf{x}_n), \tag{2.3.6a}$$

$$X_{ni} = \mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hS)(\widetilde{F}_{nj} + S\mathbf{x}_n), \tag{2.3.6b}$$

$$\widetilde{F}_{ni} = \tilde{f}(t_n + c_i h, X_{ni}), \tag{2.3.6c}$$

which can again be displayed in a more familiar looking Butcher-Tableau without the χ-functions:

$$\begin{array}{c|cccc} c_1 & 0 & & & \\ c_2 & \mathbf{a}_{21}(hS) & & & \\ \vdots & \vdots & \ddots & \ddots & \\ c_s & \mathbf{a}_{s,1}(hS) & \dots & \mathbf{a}_{s,s-1}(hS) & \\ \hline & \mathbf{b}_1(hS) & \dots & \mathbf{b}_{s-1}(hS) & \mathbf{b}_s(hS) \end{array}.$$

We will use this Butcher-Tableau for our algorithms in the following. If we insert (2.3.6b) into (2.3.6c), the iteration in (2.3.6) changes to

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)(\widetilde{F}_{ni} + S\mathbf{x}_n), \tag{2.3.7a}$$

$$\widetilde{F}_{ni} = \tilde{f}\Big(\mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hS)(\widetilde{F}_{nj} + S\mathbf{x}_n)\Big). \tag{2.3.7b}$$

Remark 2.3.8.
Since our matrix S is obtained by combining several submodels, it has block diagonal form. This lowers the effort for the computation of e^{hS} and $e^{h(1-\theta)S}$. This is an additional reason why exponential integrators are interesting for system ($\star\star\star$).
It should be noted that the coefficients of the methods depend on the step width h. For variable step widths, the computation effort for the coefficients grows. However, for a fixed step width – as we have in mind – we only need to calculate the coefficients once at the beginning of the method.

2.3.1. Consistency

In order to simplify the statement of the order conditions, we need the following definitions; see (2.10), (2.17) and (2.37) in [HO10]:

$$\varphi_k(hS) := \int_0^1 e^{h(1-\theta)S}\frac{\theta^{k-1}}{(k-1)!}d\theta, \quad k \geq 1, \tag{2.3.9a}$$

$$\varphi_{k,\ell}(hS) := \varphi_k(c_\ell hS), \tag{2.3.9b}$$

$$\psi_k(hS) := \varphi_k(hS) - \sum_{i=1}^{m} \mathbf{b}_i(hS)\frac{c_i^{k-1}}{(k-1)!}, \tag{2.3.9c}$$

$$\psi_{k,\ell}(hS) := \varphi_{k,\ell}(hS)c_\ell^k - \sum_{j=1}^{\ell-1} \mathbf{a}_{\ell j}(hS)\frac{c_j^{k-1}}{(k-1)!}. \tag{2.3.9d}$$

For reasons of notational simplicity, we will occasionally drop the argument hS of the above functions.

As mentioned before, exponential integrators have been used for some time. A rigorous analysis

and a comprehensive overview is found in [HO10]. This includes an overview of consistency order conditions for explicit exponential Runge-Kutta methods of orders 1-4 which we recite here.

Number	Order	Order condition
1	1	$\psi_1(hS) = 0$
2	2	$\psi_2(hS) = 0$
3	2	$\psi_{1,i}(hS) = 0, \text{ for } i = 1,\dots,s$
4	3	$\psi_3(hS) = 0$
5	3	$\sum_{i=1}^{s} \mathbf{b}_i(hS) J_1 \psi_{2,i}(hS) = 0$
6	4	$\psi_4(hS) = 0$
7	4	$\sum_{i=1}^{s} \mathbf{b}_i(hS) J_1 \psi_{3,i}(hS) = 0$
8	4	$\sum_{i=1}^{s} \mathbf{b}_i(hS) J_1 \sum_{j=2}^{i-1} \mathbf{a}_{ij}(hS) J_1 \psi_{2,j}(hS) = 0$
9	4	$\sum_{i=1}^{s} \mathbf{b}_i(hS) J_2 \psi_{2,i}(hS) = 0$

Table 2.3.10: Order conditions for EERK methods of orders 1–4 ([HO10] (Table 2.2))

Here, $J_i,\ i \in \{1,2\}$ denote bounded operators. It should be noted that in order for a method to have a certain consistency, all conditions prior to the required consistency order have to be fulfilled. Thus, in order to proof that a method has third order, we need to show that conditions 1 to 5 hold.

2.3.2. Stability

We want to analyze the stability of explicit exponential Runge-Kutta methods of orders 1–4 in an analogous way as for ordinary Runge-Kutta methods. For exponential Runge-Kutta methods, we consider a problem of the form

$$\dot{\mathbf{x}}(t) = S\mathbf{x}(t) + \tilde{f}(\mathbf{x}(t)), \quad \mathbf{x} \in \mathbb{R}^n,\ S \in \mathbb{R}^{n\times n}. \tag{2.3.11}$$

As in (2.1.14), we approximate the nonlinear function $\tilde{f}(\mathbf{x}(t))$ with a linear function $L\mathbf{x}(t)$. Our test problem then reads

$$\dot{\mathbf{x}}(t) = S\mathbf{x}(t) + L\mathbf{x}(t), \quad \mathbf{x} \in \mathbb{R}^n,\ S \in \mathbb{R}^{n\times n},\ L \in \mathbb{R}^{n\times n}. \tag{2.3.12}$$

In the one-dimensional case, we set $S := \sigma \in \mathbb{C}$ and $L := \lambda \in \mathbb{C}$, which yields

$$\dot{x}(t) = \sigma x(t) + \lambda x(t), \quad x \in \mathbb{R}. \tag{2.3.13}$$

In Section 2.1.2, we saw that the stability region is two-dimensional for the test equation (2.1.6). So here, the stability region will be four-dimensional in the general case. This problem can be handled by fixing or restraining certain parts of σ and λ as we will see.

Analyses with the same approach yet with other focuses are carried out in e. g. [BKV98, FD99, CM02, MZ09, MZ13, OP16, Zh17b]. We shortly summarize their findings and distinct our aim. In [BKV98], the stability of different exponential integration methods like the Adams-Moulton and Adams-Bashforth schemes (not exponential RK methods) is analyzed. In order to deal with the above mentioned problem, Beylkin et. al [BKV98] consider σ to be fixed and real (negative). In contrast, Fornberg and Driscoll [FD99] analyze the stability of implicit-explicit Adams-Moulton and Adams-Bashforth schemes for purely imaginary values of σ and λ. Cox and Matthews [CM02] then build on these works and consider σ and λ to take real (negative) values for Adams-Moulton and Adams-Bashforth schemes but also for a second order exponential Runge-Kutta method. A real negative σ is also the subject of investigation of Owolabi and Patidar [OP16] (fourth order EERK) and Zhu [Zh17b] (second order EERK). In our case, we expect σ to be purely imaginary (with high modulus). As we will see later on, this setting is comparably critical to a real negative σ but not yet analyzed. Maset and Zennaro [MZ09, MZ13] analyze the conditional and unconditional stability properties of exponential Runge-Kutta methods. Their approach is very close to our approach. However, they focus on the asymptotic behavior of the stability function. From their results one cannot directly see how to choose a step size h for a certain setting.

We now start our analysis of the linear one-dimensional stability function. As in Section 2.1.2, we insert the test equation (2.3.13) into the iteration (2.3.7b) and obtain

$$\widetilde{F}_{ni} = \lambda \left(x_n + h \sum_{j=1}^{s} a_{ij}(h\sigma)(\widetilde{F}_{nj} + \sigma x_n) \right). \tag{2.3.14}$$

Let $\widetilde{\mathbf{F}} := (\widetilde{F}_{n1}, \dots, \widetilde{F}_{ns})^T \in \mathbb{R}^s$, $A(h\sigma) := (a_{ij}(h\sigma))_{i=1,\dots,s;j=1,\dots,s} \in R^{s\times s}$, $\mathbf{e} := (1,\dots,1)^T \in \mathbb{R}^s$ and $\mathbf{b}(h\sigma) := (b_1(h\sigma),\dots,b_s(h\sigma))^T \in \mathbb{R}^s$. Then,

$$\begin{aligned} \widetilde{\mathbf{F}} &= \lambda x_n \mathbf{e} + h\lambda A(h\sigma)\widetilde{\mathbf{F}} + h\lambda A(h\sigma)\sigma x_n \mathbf{e} \\ \Leftrightarrow (I - h\lambda A(h\sigma))\widetilde{\mathbf{F}} &= (\lambda I + h\lambda\sigma A(h\sigma)) x_n \mathbf{e} \\ \Leftrightarrow \widetilde{\mathbf{F}} &= \lambda (I - h\lambda A(h\sigma))^{-1} (I + h\sigma A(h\sigma)) \mathbf{e} x_n. \end{aligned}$$

Inserting this into (2.3.7a), we obtain

$$\begin{aligned} x_{n+1} &= x_n + h\sum_{i=1}^{m} b_i(h\sigma)(\widetilde{F}_{ni} + \sigma x_n) \\ &= x_n + h\mathbf{b}(h\sigma)^T\widetilde{\mathbf{F}} + h\mathbf{b}(h\sigma)^T\mathbf{e}\sigma x_n \\ &= x_n + h\mathbf{b}(h\sigma)^T\lambda(I - h\lambda A(h\sigma))^{-1}(I + h\sigma A(h\sigma))\mathbf{e}x_n + h\sigma\mathbf{b}(h\sigma)^T\mathbf{e}x_n \\ &= \left[I + h\lambda\mathbf{b}(h\sigma)^T(I - h\lambda A(h\sigma))^{-1}(I + h\sigma A(h\sigma))\mathbf{e} + h\sigma\mathbf{b}(h\sigma)^T\mathbf{e}\right]x_n. \end{aligned}$$

Hence,

$$\phi(h\sigma, h\lambda) = I + h\lambda\mathbf{b}(h\sigma)^T(I - h\lambda A(h\sigma))^{-1}(I + h\sigma A(h\sigma))\mathbf{e} + h\sigma\mathbf{b}(h\sigma)^T\mathbf{e}$$

or with $w := h\sigma$ and $z := h\lambda$, we write

$$\phi(w,z) = I + z\mathbf{b}(w)^T(I - zA(w))^{-1}(I + wA(w))\mathbf{e} + w\mathbf{b}(w)^T\mathbf{e}. \tag{2.3.15}$$

Before continuing the stability analysis, we would like to obtain a similar result to Lemma 2.1.13 for our n-dimensional test problem (2.3.12). Unfortunately, this is not as straight forward as in the ordinary RK case. A similar result can only be shown, if S and L commute, which does not generally occur in real applications. However, as Hundsdorfer states, the step size predictions obtained by the study of the scalar test equation are surprisingly accurate in practical applications which makes their in depth study valuable anyway [HR07]. Therefore, we now content ourselves with the scalar stability analysis in this thesis.

2.3.3. First order

The only one-stage explicit exponential Runge-Kutta method is called the exponential Euler method (cf. [HO10] (Example 2.12)). Its Butcher-Tableau is given by:

$$\begin{array}{c|c} 0 & \\ \hline & \varphi_1(hS) \end{array}.$$

The algorithm then reads

Algorithm 2.3.16: One-Stage Explicit Exponential Runge-Kutta Method

Input : $\mathbf{x}_0, T, N, S, \tilde{f}, \mathbf{b}_1(hS)$

Output: $\mathbf{x}$

$\mathbf{x} \leftarrow \mathbf{x}_0$

$h \leftarrow \frac{T}{N}$

for $\ell \leftarrow 1$ *to* $N+1$ **do**

  $\widetilde{F} \leftarrow \tilde{f}(t, \mathbf{x})$;

  $\mathbf{x} \leftarrow \mathbf{x} + h \cdot \mathbf{b}_1(hS)(\widetilde{F} + S\mathbf{x})$;

  $t \leftarrow t + h$;

end

2.3.3.1. Consistency

Lemma 2.3.17 (cf. [HO10] (Table 2.2))**.**
The explicit exponential Euler method is consistent of first order if $\psi_1(hS) = 0$.

Proof.
First, we use definition (2.3.9c) in order to dissolve the condition $\psi_1(hS) = 0$:

$$\begin{aligned} \psi_1(hS) &= \varphi_1(hS) - \sum_{i=1}^{1} \mathbf{b}_i(hS) \frac{c_i^{1-1}}{(1-1)!} \\ &= \varphi_1(hS) - \mathbf{b}_1(hS) \overset{!}{=} 0 \\ \Leftrightarrow \mathbf{b}_1(hS) &= \varphi_1(hS). \end{aligned} \tag{2.3.18}$$

In order to observe the consistency error, we need to compare the Taylor expansion of the difference operator with the Taylor expansion of the increment function $\Phi(t, \mathbf{x})$. First, we calculate the Taylor expansion of the difference operator up to first order:

$$\begin{aligned} \mathbf{x}'(t) &= \frac{\mathbf{x}(t+h) - \mathbf{x}(t)}{h} \\ &= \mathbf{x}'(t) + O(h) \\ &= S\mathbf{x} + \tilde{f}(t, \mathbf{x}) + O(h). \end{aligned}$$

The increment function $\Phi(t, x)$ reads

$$\Phi(t, x) = \mathbf{b}_1(hS)(\widetilde{F} + S\mathbf{x}) \overset{(2.3.18)}{=} \varphi_1(hS)(\tilde{f}(t, \mathbf{x}) + S\mathbf{x}).$$

We know that the exponential function in $\varphi_1(hS)$ (see definition (2.3.9a)) has a power series representation (see Lemma A.1.6) as

$$\varphi_1(hS) = \int_0^1 e^{h(1-\theta)S} d\theta = \int_0^1 \sum_{k=0}^{\infty} \frac{(h(1-\theta)S)^k}{k!} d\theta.$$

By Theorem A.1.7 and Remark A.1.8, we can interchange summation and integration in the following Taylor expansion of the increment function $\Phi(t,\mathbf{x})$:

$$\begin{aligned}
\Phi(t,\mathbf{x}) &= \varphi_1(hS)(\tilde{f}(t,\mathbf{x}) + S\mathbf{x}) \\
&= \int_0^1 \sum_{k=0}^{\infty} \frac{(h(1-\theta)S)^k}{k!} d\theta \cdot (\tilde{f}(t,\mathbf{x}) + S\mathbf{x}) \\
&= \int_0^1 I + h \cdot \sum_{k=1}^{\infty} \frac{h^{k-1}((1-\theta)S)^k}{k!} d\theta \cdot (\tilde{f}(t,\mathbf{x}) + S\mathbf{x}) \\
&= \left[\int_0^1 I d\theta + O(h) \right] (\tilde{f}(t,\mathbf{x}) + S\mathbf{x}) \\
&= \tilde{f}(t,\mathbf{x}) + S\mathbf{x} + O(h).
\end{aligned}$$

Since $\Phi(t,\mathbf{x}) - \mathbf{x}'(t) = O(h)$, the method is convergent of first order. □

2.3.3.2. Stability

For the analysis of stability, we use the one-dimensional test equation in (2.3.13) and the definitions $w := h\sigma$ and $z := h\lambda$.

Using definition (2.3.9a), we can then explicitly calculate $\varphi_1(h\sigma)$:

$$\varphi_1(h\sigma) = \varphi_1(w) = \int_0^1 e^{(1-\theta)w} d\theta = \left[-\frac{1}{w} e^{(1-\theta)w} \right]_0^1 = -\frac{1}{w} + \frac{1}{w} e^w.$$

Inserted in (2.3.15), we obtain the stability function

$$\begin{aligned}
\phi(w,z) &= 1 + zb(w) + wb(w)) \\
&= 1 + (z+w)b(w) \\
&= 1 + (z+w)\varphi_1(w)
\end{aligned}$$

$$= 1 + (z+w)\left(-\frac{1}{w} + \frac{1}{w}e^w\right)$$
$$= 1 + \frac{z+w}{w}(-1+e^w).$$

So, for the explicit exponential Euler method, the stability function reads

$$\phi(w,z) = 1 + \frac{z+w}{w}(-1+e^w). \tag{2.3.19}$$

Theorem 2.3.20 (Author's contribution)**.**
The stability region of the explicit exponential Euler method is given by

$$\{w,z \in \mathbb{C} : |\phi(w,z)| \leq 1\},$$

with

$$\begin{aligned} |\phi(w,z)|^2 = e^{2w_{re}} &+ \frac{(1-2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}})}{w_{re}^2 + w_{im}^2}(z_{re}^2 + z_{im}^2) \\ &+ 2\frac{w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) + w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2} z_{re} \\ &+ 2\frac{w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2} z_{im}. \end{aligned} \tag{2.3.21}$$

Proof.
Let

$$\begin{aligned} \sigma &:= \sigma_{re} + i\sigma_{im}, \\ \lambda &:= \ell_{re} + i\ell_{im}, \\ w &:= w_{re} + iw_{im} := h\sigma_{re} + ih\sigma_{im}, \\ z &:= z_{re} + iz_{im} := h\ell_{re} + ih\ell_{im}. \end{aligned}$$

Additionally, let $w \neq 0 \in \mathbb{C}$. We now calculate $|\phi(w,z)|$. For the modulus of a complex statement, we need to know its imaginary and its real part, which we determine first by inserting the definitions of z and w into (2.3.19) and sorting the summands:

$$\begin{aligned} \phi(w,z) &= 1 + \frac{z+w}{w}(-1+e^w) \\ &= 1 + \frac{z_{re} + w_{re} + i(z_{im} + w_{im})}{w_{re} + iw_{im}}(-1 + e^{w_{re}}(\cos(w_{im}) + i\sin(w_{im}))) \\ &= 1 + \frac{\big(z_{re} + w_{re} + i(z_{im} + w_{im})\big) \cdot (w_{re} - iw_{im})}{w_{re}^2 + w_{im}^2}(-1 + e^{w_{re}}(\cos(w_{im}) + i\sin(w_{im}))) \end{aligned}$$

$$
\begin{aligned}
&= 1 + \frac{(z_{re}+w_{re})w_{re} + (z_{im}+w_{im})w_{im} + i(-(z_{re}+w_{re})w_{im} + (z_{im}+w_{im})w_{re})}{w_{re}^2 + w_{im}^2} \\
&\quad \cdot (-1 + e^{w_{re}}(\cos(w_{im}) + i\sin(w_{im}))) \\
&= \left[1 - \frac{(z_{re}+w_{re})w_{re} + (z_{im}+w_{im})w_{im}}{w_{re}^2 + w_{im}^2}(1 - e^{w_{re}}\cos(w_{im}))\right. \\
&\quad \left. - \frac{-(z_{re}+w_{re})w_{im} + (z_{im}+w_{im})w_{re}}{w_{re}^2 + w_{im}^2} e^{w_{re}}\sin(w_{im})\right] \\
&\quad + i\left[\frac{(z_{re}+w_{re})w_{re} + (z_{im}+w_{im})w_{im}}{w_{re}^2 + w_{im}^2} e^{w_{re}}\sin(w_{im})\right. \\
&\quad \left. - \frac{-(z_{re}+w_{re})w_{im} + (z_{im}+w_{im})w_{re}}{w_{re}^2 + w_{im}^2}(1 - e^{w_{re}}\cos(w_{im}))\right].
\end{aligned}
$$

In order to simplify the notation, let

$$
a := \frac{(z_{re}+w_{re})w_{re} + (z_{im}+w_{im})w_{im}}{w_{re}^2 + w_{im}^2} = \frac{z_{re}w_{re} + z_{im}w_{im}}{w_{re}^2 + w_{im}^2} + 1 = \frac{w_{re}}{w_{re}^2 + w_{im}^2} z_{re} + \frac{w_{im}}{w_{re}^2 + w_{im}^2} z_{im} + 1
$$

and

$$
b := \frac{-(z_{re}+w_{re})w_{im} + (z_{im}+w_{im})w_{re}}{w_{re}^2 + w_{im}^2} = \frac{-z_{re}w_{im} + z_{im}w_{re}}{w_{re}^2 + w_{im}^2} = \frac{-w_{im}}{w_{re}^2 + w_{im}^2} z_{re} + \frac{w_{re}}{w_{re}^2 + w_{im}^2} z_{im}.
$$

Then,

$$
\begin{aligned}
\phi(w,z) = &\left[1 - a(1 - e^{w_{re}}\cos(w_{im})) - be^{w_{re}}\sin(w_{im})\right] \\
&+ i\left[ae^{w_{re}}\sin(w_{im}) - b(1 - e^{w_{re}}\cos(w_{im}))\right].
\end{aligned} \tag{2.3.22}
$$

If we take the modulus of (2.3.22), we have:

$$
|\phi(w,z)| = \sqrt{\left[1 - a(1 - e^{w_{re}}\cos(w_{im})) - be^{w_{re}}\sin(w_{im})\right]^2 + \left[ae^{w_{re}}\sin(w_{im}) - b(1 - e^{w_{re}}\cos(w_{im}))\right]^2}.
$$

We want to know for which $h > 0$ and which $w,\ z$ we have $|\phi(w,z)| \leq 1$. Hence, we only need to consider the term under the root, since for $x \in \mathbb{R}_+$, we know that $\sqrt{x} \leq 1$ if and only if $x \leq 1$:

$$
\begin{aligned}
|\phi(w,z)|^2 &= 1 + a^2(1 - e^{w_{re}}\cos(w_{im}))^2 + b^2 e^{2w_{re}}\sin^2(w_{im}) \\
&\quad - 2\left(a(1 - e^{w_{re}}\cos(w_{im})) + be^{w_{re}}\sin(w_{im})\right) + 2ab(1 - e^{w_{re}}\cos(w_{im}))e^{w_{re}}\sin(w_{im}) \\
&\quad + a^2 e^{2w_{re}}\sin^2(w_{im}) - 2abe^{w_{re}}\sin(w_{im})(1 - e^{w_{re}}\cos(w_{im})) + b^2(1 - e^{w_{re}}\cos(w_{im}))^2 \\
&= 1 + a^2(1 - e^{w_{re}}\cos(w_{im}))^2 + b^2 e^{2w_{re}}\sin^2(w_{im}) \\
&\quad - 2\left(a(1 - e^{w_{re}}\cos(w_{im})) + be^{w_{re}}\sin(w_{im})\right) \\
&\quad + a^2 e^{2w_{re}}\sin^2(w_{im}) + b^2(1 - e^{w_{re}}\cos(w_{im}))^2
\end{aligned}
$$

$$
\begin{aligned}
&= 1 + a^2(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}\cos^2(w_{im})) + b^2 e^{2w_{re}}\sin^2(w_{im}) \\
&\quad - 2\left(a(1 - e^{w_{re}}\cos(w_{im})) + be^{w_{re}}\sin(w_{im})\right) \\
&\quad + a^2 e^{2w_{re}}\sin^2(w_{im}) + b^2(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}\cos^2(w_{im})) \\
&= 1 + a^2(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - 2\left(a(1 - e^{w_{re}}\cos(w_{im})) + be^{w_{re}}\sin(w_{im})\right) \\
&\quad + b^2(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) \\
&= 1 + (a^2 + b^2)(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - 2\left(a(1 - e^{w_{re}}\cos(w_{im})) + be^{w_{re}}\sin(w_{im})\right).
\end{aligned}
$$

Now,

$$
\begin{aligned}
a^2 + b^2 &= \left(\frac{z_{re}w_{re} + z_{im}w_{im}}{w_{re}^2 + w_{im}^2} + 1\right)^2 + \left(\frac{-z_{re}w_{im} + z_{im}w_{re}}{w_{re}^2 + w_{im}^2}\right)^2 \\
&= \frac{z_{re}^2 w_{re}^2 + 2z_{re}w_{re}z_{im}w_{im} + z_{im}^2 w_{im}^2}{(w_{re}^2 + w_{im}^2)^2} + 2\frac{z_{re}w_{re} + z_{im}w_{im}}{w_{re}^2 + w_{im}^2} + 1 \\
&\quad + \frac{z_{re}^2 w_{im}^2 + -2z_{re}w_{im}z_{im}w_{re} + z_{im}^2 w_{re}^2}{(w_{re}^2 + w_{im}^2)^2} \\
&= \frac{1}{w_{re}^2 + w_{im}^2}(z_{re}^2 + z_{im}^2) + 2\frac{w_{re}}{w_{re}^2 + w_{im}^2}z_{re} + 2\frac{w_{im}}{w_{re}^2 + w_{im}^2}z_{im} + 1.
\end{aligned}
$$

In order to obtain circle equations (see Lemma 2.3.27 and Theorem 2.3.29), we need to sort the summands by terms in z_{re}^2, z_{im}^2, z_{re} and z_{im}. We have:

$$
\begin{aligned}
|\phi(w,z)|^2 &= 1 + (a^2 + b^2)(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - 2\left(a(1 - e^{w_{re}}\cos(w_{im})) + be^{w_{re}}\sin(w_{im})\right) \\
&= 1 + \left(\frac{1}{w_{re}^2 + w_{im}^2}(z_{re}^2 + z_{im}^2) + 2\frac{w_{re}}{w_{re}^2 + w_{im}^2}z_{re} + 2\frac{w_{im}}{w_{re}^2 + w_{im}^2}z_{im} + 1\right) \\
&\quad \cdot (1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) \\
&\quad - 2\left(\left(\frac{w_{re}}{w_{re}^2 + w_{im}^2}z_{re} + \frac{w_{im}}{w_{re}^2 + w_{im}^2}z_{im} + 1\right)(1 - e^{w_{re}}\cos(w_{im})) \right. \\
&\quad \left. + \left(\frac{-w_{im}}{w_{re}^2 + w_{im}^2}z_{re} + \frac{w_{re}}{w_{re}^2 + w_{im}^2}z_{im}\right)e^{w_{re}}\sin(w_{im})\right) \\
&= 1 + (1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - 2(1 - e^{w_{re}}\cos(w_{im})) \\
&\quad + \frac{(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}})}{w_{re}^2 + w_{im}^2}(z_{re}^2 + z_{im}^2) \\
&\quad + \frac{2w_{re}(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - 2w_{re}(1 - e^{w_{re}}\cos(w_{im})) + 2w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2}z_{re} \\
&\quad + \frac{2w_{im}(1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - 2w_{im}(1 - e^{w_{re}}\cos(w_{im})) - 2w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2}z_{im}
\end{aligned}
$$

$$
\begin{aligned}
&= e^{2w_{re}} + \frac{(1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}})}{w_{re}^2+w_{im}^2}(z_{re}^2+z_{im}^2)\\
&\quad + 2\frac{w_{re}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})+w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{re}\\
&\quad + 2\frac{w_{im}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})-w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{im}.
\end{aligned}
$$

So,

$$
\begin{aligned}
|\phi(w,z)|^2 = e^{2w_{re}} &+ \frac{(1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}})}{w_{re}^2+w_{im}^2}(z_{re}^2+z_{im}^2)\\
&+ 2\frac{w_{re}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})+w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{re}\\
&+ 2\frac{w_{im}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})-w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{im}.
\end{aligned}
$$

□

Before starting the analysis of the function given in (2.3.21) with respect to w_{re} and w_{im}, we start with a peculiar case (a purely imaginary σ, where $w_{im} = \sigma h$ is a multiple of 2π) that we can leave out of our considerations subsequently:

Lemma 2.3.23 (Author's contribution)**.**
Let $h \in \mathbb{R}_+$ be the step size. For $w_{re} = 0$ and $w_{im} = 2k\pi, k \in \mathbb{N}$, we have $\phi(w,z) = 1$ for all $z \in \mathbb{C}$.

Proof.
Consider equation (2.3.21). For $w_{re} = 0$ and $w_{im} = 2k\pi$, we have $e^{w_{re}} = 1, e^{2w_{re}} = 1, \cos(w_{im}) = 1$ and $\sin(w_{im}) = 0$. Hence

$$
\begin{aligned}
|\phi(w,z)|^2 &= 1 + \frac{(1-2\cdot 1\cdot 1+1)}{0^2+(2k\pi)^2}(z_{re}^2+z_{im}^2) + 2\frac{0\cdot(-1\cdot 1+1)+2k\pi\cdot 1\cdot 0}{0^2+(2k\pi)^2}z_{re}\\
&\quad + 2\frac{2k\pi\cdot(-1\cdot 1+1)-0\cdot 1\cdot 0}{0^2+(2k\pi)^2}z_{im}\\
&= 1.
\end{aligned}
$$

□

Now, we analyze the case when $w_{im} \neq 2k\pi, k \in \mathbb{N}$. We want to show that the stability regions given by $\{z \in \mathbb{C} : |\phi(w,z)| \leq 1\}$, where $|\phi(w,z)|$ is given in (2.3.21), are circles. Therefore, we first need to state the formulation of a circle in the complex plane.

Lemma 2.3.24. (cf. [Ru08])
Let $\alpha, z \in \mathbb{C}$ and $\beta, \gamma \in \mathbb{R}$. A circle in the complex plane is given by

$$\alpha\bar{z} + \bar{\alpha}z = \beta z\bar{z} + \gamma, \tag{2.3.25}$$

where $\frac{\alpha}{\beta}$ denotes the center and $\frac{1}{\beta}$ denotes the radius. Additionally,

$$\alpha\bar{\alpha} - 1 = \beta\gamma \tag{2.3.26}$$

has to be fulfilled.

Proof.
In general, a circle is given by

$$|z - o| = r,$$

where $o \in \mathbb{C}$ denotes the center of the circle and $r \in \mathbb{R}$ its radius. Squaring the equation, we obtain

$$(z - o)(\bar{z} - \bar{o}) = r^2.$$

With $\beta := \frac{1}{r}$ and $\alpha := \beta o$, this equation transforms to

$$\begin{aligned} & \left(z - \frac{\alpha}{\beta}\right)\left(\bar{z} - \frac{\bar{\alpha}}{\beta}\right) && = \frac{1}{\beta^2} \\ \overset{\cdot\beta}{\Leftrightarrow} & \left(z - \frac{\alpha}{\beta}\right)(\beta\bar{z} - \bar{\alpha}) && = \frac{1}{\beta} \\ \Leftrightarrow\ & \beta z\bar{z} - \alpha\bar{z} - \bar{\alpha}z + \frac{\alpha\bar{\alpha}}{\beta} - \frac{1}{\beta} = 0 \end{aligned}$$

With $\gamma := \frac{\alpha\bar{\alpha}-1}{\beta}$ (condition in (2.3.26)), we obtain the equation (2.3.25). □

Lemma 2.3.27 (Author's contribution)**.**
Let $z = z_{re} + iz_{im}$ and $\alpha = \alpha_{re} + i\alpha_{im}$. Then equation (2.3.25) is equivalent to

$$0 = -2\alpha_{re}z_{re} - 2\alpha_{im}z_{im} + \beta(z_{re}^2 + z_{im}^2) + \gamma. \tag{2.3.28}$$

Proof.

$$\begin{aligned} 0 &= \alpha\bar{z} + \bar{\alpha}z - \beta z\bar{z} - \gamma \\ &= (\alpha_{re} + i\alpha_{im})(z_{re} - iz_{im}) + (\alpha_{re} - i\alpha_{im})(z_{re} + iz_{im}) - \beta(z_{re}^2 + z_{im}^2) - \gamma \\ &= 2\alpha_{re}z_{re} + 2\alpha_{im}z_{im} - \beta(z_{re}^2 + z_{im}^2) - \gamma. \end{aligned}$$

□

With these circle equations we obtain the following theorem

Theorem 2.3.29 (Author's contribution)**.**
Let $w \neq 2k\pi i,\ k \in \mathbb{N}$. The stability regions of the explicit exponential Euler method are circles. For given w the circles have their center at

$$\left(-\frac{w_{re}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})+w_{im}e^{w_{re}}\sin(w_{im})}{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}, -\frac{w_{im}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})-w_{re}e^{w_{re}}\sin(w_{im})}{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}\right) \in \mathbb{C}$$

and their radius is

$$\frac{|w|}{\sqrt{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}}.$$

Proof.
Consider equation (2.3.21):

$$\begin{aligned}|\phi(w,z)|^2 = e^{2w_{re}} &+ \frac{(1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}})}{w_{re}^2+w_{im}^2}(z_{re}^2+z_{im}^2)\\ &+ \frac{2w_{re}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})+2w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{re}\\ &+ \frac{2w_{im}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})-2w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{im}.\end{aligned}$$

We are interested in the circle for $|\phi(w,z)|^2 = 1$. Hence, putting $|\phi(w,z)|^2 = 1$ and bringing everything on one side we obtain:

$$\begin{aligned}0 = e^{2w_{re}} - 1 &+ \frac{(1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}})}{w_{re}^2+w_{im}^2}(z_{re}^2+z_{im}^2)\\ &+ \frac{2w_{re}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})+2w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{re}\\ &+ \frac{2w_{im}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})-2w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2+w_{im}^2}z_{im}.\end{aligned}$$

Comparing this to the circle equation

$$0 = \beta(z_{re}^2+z_{im}^2) - 2\alpha_{re}z_{re} - 2\alpha_{im}z_{im} + \gamma$$

from Lemma 2.3.27, we could think of

$$\begin{aligned}\tilde{\beta} &= \frac{(1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}})}{w_{re}^2+w_{im}^2}\\ \tilde{\gamma} &= e^{2w_{re}} - 1\end{aligned}$$

$$\tilde{\alpha}_{re} = -\frac{w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) + w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2}$$
$$\tilde{\alpha}_{im} = -\frac{w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2}.$$

We need $\tilde{\alpha}\bar{\tilde{\alpha}} - \tilde{\beta}\tilde{\gamma} = 1$ (see (2.3.26)) in order for this to be the real coefficients of the circle. At the moment, we have:

$$\begin{aligned}
\tilde{\alpha}\bar{\tilde{\alpha}} &= \tilde{\alpha}_{re}^2 + \tilde{\alpha}_{im}^2 \\
&= \left(-\frac{w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) + w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2}\right)^2 \\
&\quad + \left(-\frac{w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2}\right)^2 \\
&= \frac{1}{(w_{re}^2 + w_{im}^2)^2}\Big(w_{re}^2(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}})^2 + 2w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}})w_{im}e^{w_{re}}\sin(w_{im}) \\
&\quad + w_{im}^2 e^{2w_{re}}\sin^2(w_{im}) + w_{im}^2(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}})^2 \\
&\quad - 2w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}})w_{re}e^{w_{re}}\sin(w_{im}) + w_{re}^2 e^{2w_{re}}\sin^2(w_{im})\Big) \\
&= \frac{1}{(w_{re}^2 + w_{im}^2)^2}\Big(w_{re}^2\big(e^{2w_{re}}\cos^2(w_{im}) - 2e^{w_{re}}\cos(w_{im})e^{2w_{re}} + e^{4w_{re}} + e^{2w_{re}}\sin^2(w_{im})\big) \\
&\quad + w_{im}^2\big(e^{2w_{re}}\cos^2(w_{im}) - 2e^{w_{re}}\cos(w_{im})e^{2w_{re}} + e^{4w_{re}} + e^{2w_{re}}\sin^2(w_{im})\big)\Big) \\
&= \frac{e^{2w_{re}}\cos^2(w_{im}) - 2e^{w_{re}}\cos(w_{im})e^{2w_{re}} + e^{4w_{re}} + e^{2w_{re}}\sin^2(w_{im})}{w_{re}^2 + w_{im}^2} \\
&= \frac{e^{2w_{re}} - 2e^{w_{re}}\cos(w_{im})e^{2w_{re}} + e^{4w_{re}}}{w_{re}^2 + w_{im}^2} \\
&= e^{2w_{re}}\frac{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}{w_{re}^2 + w_{im}^2}
\end{aligned}$$

and

$$\tilde{\beta}\tilde{\gamma} = \frac{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}{w_{re}^2 + w_{im}^2} \cdot (e^{2w_{re}} - 1).$$

Hence,

$$\tilde{\alpha}\bar{\tilde{\alpha}} - \tilde{\beta}\tilde{\gamma} = \frac{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}{w_{re}^2 + w_{im}^2}.$$

The function $\tilde{\eta}(w_{re}, w_{im}) := 1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}$ is greater than or equal to 0 for all $w_{re}, w_{im} \in \mathbb{R}$. It has its minima at $(w_{re}, w_{im}) = (0, 2k\pi),\ k \in \mathbb{N}$ where $\tilde{\eta}(0, 2k\pi) = 0$. However,

these values for w_{re} and w_{im} are excluded by our requirements. Hence, $\tilde{\eta}(w_{re}, w_{im}) > 0$ for all $(w_{re}, w_{im}) \neq (0, 2k\pi)$, $k \in \mathbb{N}$. In order to meet requirement (2.3.26), we divide our coefficients $\tilde{\alpha}, \tilde{\beta}$ and $\tilde{\gamma}$ by

$$\eta := \sqrt{\frac{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}{w_{re}^2 + w_{im}^2}} > 0$$

and obtain new equations for α, β and γ as $\alpha = \frac{\tilde{\alpha}}{\eta}, \beta = \frac{\tilde{\beta}}{\eta}$ and $\gamma = \frac{\tilde{\gamma}}{\eta}$.

Let

$$\begin{aligned}
\beta &= \frac{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}{w_{re}^2 + w_{im}^2} \cdot \frac{\sqrt{w_{re}^2 + w_{im}^2}}{\sqrt{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}} \\
&= \sqrt{\frac{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}{w_{re}^2 + w_{im}^2}}, \\
\gamma &= (e^{2w_{re}} - 1) \cdot \frac{\sqrt{w_{re}^2 + w_{im}^2}}{\sqrt{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}}, \\
\alpha_{re} &= -\frac{w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) + w_{im}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2} \cdot \frac{\sqrt{w_{re}^2 + w_{im}^2}}{\sqrt{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}} \\
&= -\frac{w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) + w_{im}e^{w_{re}}\sin(w_{im})}{\sqrt{w_{re}^2 + w_{im}^2}\sqrt{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}}, \\
\alpha_{im} &= -\frac{w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - w_{re}e^{w_{re}}\sin(w_{im})}{w_{re}^2 + w_{im}^2} \cdot \frac{\sqrt{w_{re}^2 + w_{im}^2}}{\sqrt{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}} \\
&= -\frac{w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - w_{re}e^{w_{re}}\sin(w_{im})}{\sqrt{w_{re}^2 + w_{im}^2}\sqrt{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}}.
\end{aligned}$$

Then,

$$\alpha\bar{\alpha} - \beta\gamma = 1.$$

Note that $\eta = \sqrt{\tilde{\beta}}$. Hence, $\beta = \frac{\tilde{\beta}}{\sqrt{\tilde{\beta}}} = \sqrt{\tilde{\beta}} = \eta$. So, the center of the circle is located at $\frac{\alpha}{\beta} = \frac{\tilde{\alpha}}{\eta} \cdot \frac{1}{\beta} = \frac{\tilde{\alpha}}{\eta} \cdot \frac{1}{\eta} = \frac{\tilde{\alpha}}{\eta^2}$ which is

$$\begin{aligned}
\frac{\tilde{\alpha}}{\eta^2} = \Bigg[&\frac{-(w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) + w_{im}e^{w_{re}}\sin(w_{im}))}{w_{re}^2 + w_{im}^2} \\
&-i\frac{\big(w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - w_{re}e^{w_{re}}\sin(w_{im})\big)}{w_{re}^2 + w_{im}^2}\Bigg] \cdot \frac{w_{re}^2 + w_{im}^2}{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}
\end{aligned}$$

$$= \frac{-(w_{re}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) + w_{im}e^{w_{re}}\sin(w_{im}))}{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}}$$
$$- i\frac{(w_{im}(-e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}) - w_{re}e^{w_{re}}\sin(w_{im}))}{1 - 2e^{w_{re}}\cos(w_{im}) + e^{2w_{re}}},$$

i. e. at $\left(-\frac{w_{re}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})+w_{im}e^{w_{re}}\sin(w_{im})}{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}, -\frac{w_{im}(-e^{w_{re}}\cos(w_{im})+e^{2w_{re}})-w_{re}e^{w_{re}}\sin(w_{im})}{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}\right) \in \mathbb{C}$, and the radius is $\frac{1}{\beta} = \frac{\sqrt{w_{re}^2+w_{im}^2}}{\sqrt{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}} = \frac{|w|}{\sqrt{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}}$.

□

We now analyze the behavior of the stability region circles by analyzing their radius and center coordinates with respect to the variables w and z.

Lemma 2.3.30 (Author's contribution)**.**
If the damping increases, the stability region becomes larger.

Proof sketch.
The damping is given by w_{re}. For negative w_{re} with higher modulus, the damping increases. The radius of the stability region is given by $r(w_{re}, w_{im}) = \frac{|w|}{\sqrt{1-2e^{w_{re}}\cos(w_{im})+e^{2w_{re}}}}$. Figure 2.3.31 shows that the function $r(w_{re}, w_{im})$ is monotonically increasing for increasing $|w_{re}|$ and $w_{im} \leq -10, w_{re} \leq -10$.

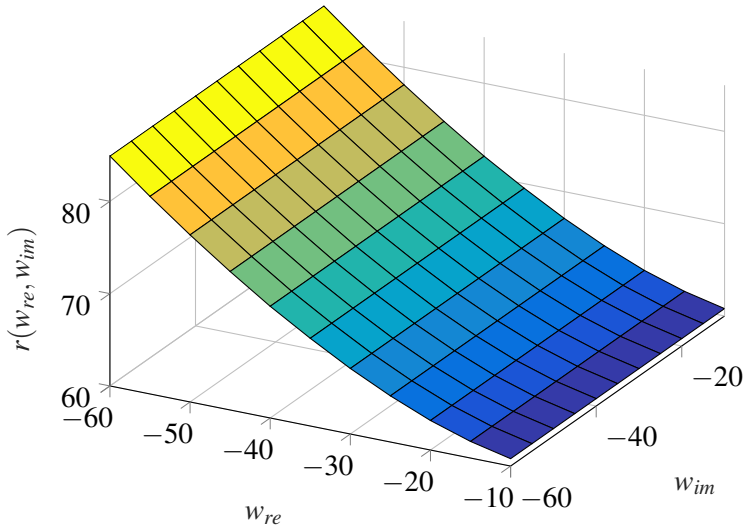


Figure 2.3.31: Radius function $r(w_{re}, w_{im})$ of stability region

□

Figure 2.3.32 visualizes Lemma 2.3.30. We observe that the stability region grows, when the real part of w has a higher modulus.

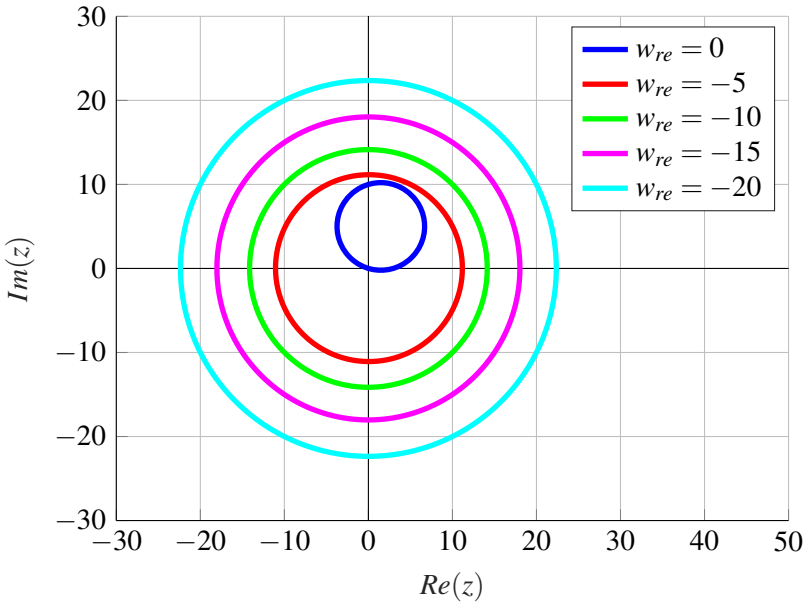


Figure 2.3.32: Growing radius of stability region for decreasing w_{re}

We additionally want to state some results for purely imaginary w ($w \in \mathbb{C} \setminus \mathbb{R}$ or $w_{re} = 0$) and for real w ($w \in \mathbb{R}$ or $w_{im} = 0$). We start with the former case.

Let $w_{re} = 0$, hence $w := iw_{im}, w_{im} \neq 0$. Inserting $e^w = e^{iw_{im}} = (\cos(w_{im}) + i\sin(w_{im}))$ into the stability function (2.3.19), we obtain

$$\phi(w,z) = 1 + \frac{z + iw_{im}}{iw_{im}} \left(-1 + e^{iw_{im}}\right) = 1 + \frac{z + iw_{im}}{iw_{im}} \left(-1 + \cos(w_{im}) + i\sin(w_{im})\right)$$

or equivalently when expanding the fraction $\frac{z+iw_{im}}{iw_{im}}$ by i

$$\phi(w,z) = 1 + \frac{w_{im} - iz}{w_{im}} \left(-1 + e^{iw}\right) = 1 + \frac{w_{im} - iz}{w_{im}} \left(-1 + \cos(w_{im}) + i\sin(w_{im})\right).$$

This can be further transformed into

$$\begin{aligned}
\phi(w,z) &= 1 + \frac{w_{im} - iz}{w_{im}} \left(-1 + \cos(w_{im}) + i\sin(w_{im})\right) \\
&= 1 + \left(-1 + \cos(w_{im}) + i\sin(w_{im})\right) - \frac{iz}{w_{im}} \left(-1 + \cos(w_{im}) + i\sin(w_{im})\right) \\
&= \cos(w_{im}) + i\sin(w_{im}) + z\left(\frac{i}{w_{im}} - \frac{i}{w_{im}}\cos(w_{im}) + \frac{1}{w_{im}}\sin(w_{im})\right).
\end{aligned} \tag{2.3.33}$$

As a first result we want to consider the limit case $w \to 0$.

Theorem 2.3.34 (cf. [HO10])**.**
Let $w_{re} = 0$. Then, $\lim_{w_{im} \to 0} \phi(w,z) = 1+z$.

Proof.
Consider the form (2.3.33):

$$\begin{aligned}\phi(w,z) &= \cos(w_{im}) + i\sin(w_{im}) + z\left(\frac{i}{w_{im}} - \frac{i}{w_{im}}\cos(w_{im}) + \frac{1}{w_{im}}\sin(w_{im})\right) \\ &= \cos(w_{im}) + i\sin(w_{im}) + z\left(i\frac{1-\cos(w_{im})}{w_{im}} + \frac{\sin(w_{im})}{w_{im}}\right).\end{aligned}$$

For $w_{im} \to 0$, also $w \to 0$. We know that $\cos(w_{im}) \overset{w_{im}\to 0}{\longrightarrow} 1$ and $\sin(w_{im}) \overset{w_{im}\to 0}{\longrightarrow} 0$. The appearing terms $\frac{1-\cos(w_{im})}{w_{im}}$ and $\frac{\sin(w_{im})}{w_{im}}$ need to be analyzed. We use L'Hôspital's rule (see Theorem A.1.2) and differentiate numerator and denominator of our fraction respectively:

- Since $\lim_{w_{im}\to 0} \frac{\sin(w_{im})}{1} = 0$, we obtain $\lim_{w_{im}\to 0} \frac{1-\cos(w)}{w} = 0$ and
- since $\lim_{w_{im}\to 0} \frac{\cos(w_{im})}{1} = 1$, we have $\lim_{w_{im}\to 0} \frac{\sin(w_{im})}{w_{im}} = 1$.

In total,

$$\lim_{w\to 0} \phi(w,z) = 1+0+z(0+1) = 1+z.$$

□

Remark 2.3.35.
For $w_{im} \to 0$, we obtain the stability function $\phi(z) = 1+z$, which is the stability function of the ordinary explicit Euler method.

Lemma 2.3.36 (Author's contribution)**.**
For $w_{re} = 0$, the stability circles have their center at $\left(-\frac{w_{im}}{2}\cot(\frac{w_{im}}{2}), -\frac{w_{im}}{2}\right) \in \mathbb{C}$ and their radius is $\frac{1}{\beta} = \frac{|w_{im}|}{\sqrt{2(1-\cos(w_{im}))}}$.

Proof.
Using Theorem 2.3.29, setting $w_{re} = 0$ and applying $\frac{\sin(w_{im})}{1-\cos(w_{im})} = \cot\left(\frac{w_{im}}{2}\right)$, we obtain the desired result. □

Lemma 2.3.37 (Author's contribution)**.**
For $w_{im} \to 2k\pi,\ k \in \mathbb{Z}\setminus\{0\}$, the radius goes to infinity.

Proof.
The radius of the circle is given by $\frac{|w_{im}|}{\sqrt{2(1-\cos(w_{im}))}}$. For $w_{im} \to 2k\pi,\ k \in \mathbb{Z}$, the denominator $\sqrt{2(1-\cos(w_{im}))}$ goes to zero, while $|w_{im}|$ stays bounded. Hence, the radius goes to infinity. □

Lemma 2.3.38 (Author's contribution)**.**
For an interval $[2k\pi, 2(k+1)\pi],\ k \in \mathbb{Z}\setminus\{-1,0\}$ the radius is smallest for $w_{im} = (2k+1)\pi,\ k \in \mathbb{Z}$. For the interval $[-2\pi, 2\pi]$, the radius is smallest at $w_{im} = 0$.

Proof.
The radius of the circle is given by $\frac{|w_{im}|}{\sqrt{2(1-\cos(w_{im}))}}$. For $w_{im} = (2k+1)\pi,\ k \in \mathbb{Z}$, we have $\cos(w_{im}) = \cos((2k+1)\pi) = -1$ which makes the denominator $\sqrt{2(1-\cos(w_{im}))}$ equal to 2. This is the highest value it can achieve. Hence, the radius is smallest at $w_{im} = (2k+1)\pi,\ k \in \mathbb{Z}$. However for the interval $[-2\pi, 2\pi]$, the smallest radius is achieved at $w_{im} = 0$, where it is equal to 1 which is smaller than $\frac{\pi}{2}$ which is the radius at $w_{im} = \pm\pi$. □

Lemma 2.3.39 (Author's contribution)**.**
For $w_{im} \to -\infty$, the center of the circle moves upwards. For $w_{im} \to \infty$, the center of the circle moves downwards.

Proof.
The vertical coordinate of the circle is given by $-\frac{w_{im}}{2}$. Hence, for $w_{im} \to -\infty$ this goes to plus infinity and for $w_{im} \to \infty$, it goes to minus infinity. □

Lemma 2.3.40 (Author's contribution)**.**
For positive (negative) w_{im}, the horizontal coordinate of the circle moves from minus infinity (plus infinity) to plus infinity (minus infinity) for every interval $[2k\pi, 2(k+1)\pi],\ k \in \mathbb{Z}\setminus\{-1,0\}$.

Proof.
The horizontal coordinate of the circle center is given by the function $h(w_{im}) := -\frac{w_{im}}{2}\cot(\frac{w_{im}}{2})$. It has roots at $w_{im} = (2k+1)\pi,\ k \in \mathbb{Z}$ and poles at $w_{im} = 2k\pi,\ k \in \mathbb{Z}\setminus\{0\}$, where for positive w_{im}

$$\lim_{w_{im}\uparrow 2k\pi} h(w_{im}) = \infty \text{ and}$$
$$\lim_{w_{im}\downarrow 2k\pi} h(w_{im}) = -\infty.$$

Furthermore,

$$\lim_{w_{im}\to 0} h(w_{im}) = -1.$$

A plot of $h(w_{im})$ is given in Figure 2.3.41.

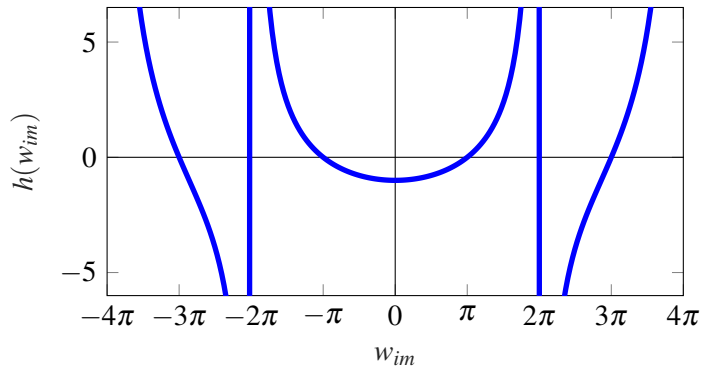


Figure 2.3.41: Plot of $h(w_{im})$

This means that for every $k_0 \in \mathbb{Z}$ the horizontal coordinate of the circle center starts to move from left to right starting from $w_{im} = 2k_0\pi + \varepsilon$ up to $w_{im} = 2(k_0+1)\pi - \varepsilon$ if $w_{im} > 0$. Since $h(w_{im})$ is an even function ($h(-w_{im}) = h(w_{im})$), this behavior is mirrored for negative w_{im}. □

Lemmas 2.3.37 to 2.3.40 are visualized in Figure 2.3.42. For w_{im} going from $-4\pi+\varepsilon$ to $-2\pi-\varepsilon$ we see that the radius becomes very large when w_{im} is close to a multiple of 2π (Lemma 2.3.37), and it is smallest towards the middle of the interval (Lemma 2.3.38). We can also observe how the circle center (and thus the circle) moves from far right to far left for w_{im} in this interval (Lemma 2.3.40), whereas the circle center moves downwards for increasing w_{im} (Lemma 2.3.39).

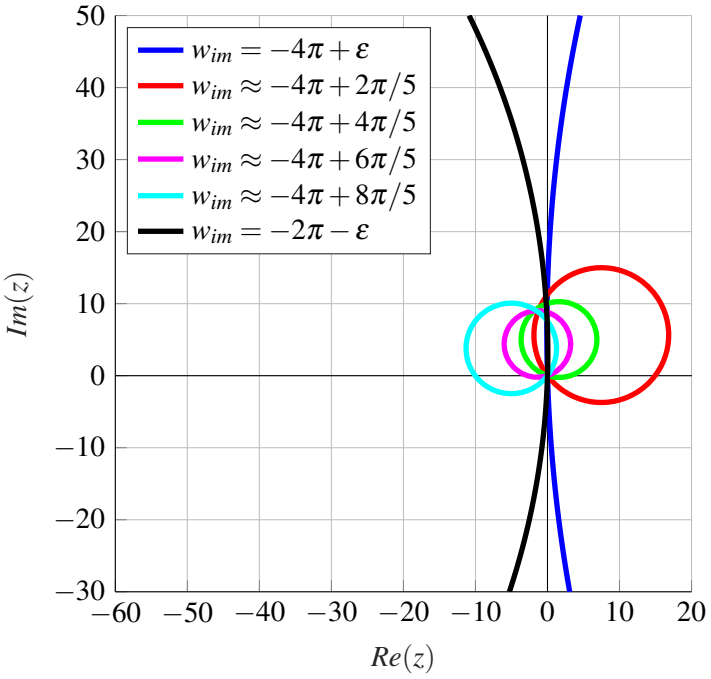


Figure 2.3.42: Growing radius of stability region for $w_{im} \to 2\pi$

The behavior of the stability regions for the exponential Euler method for w_{im} in the interval $[-2\pi, 0]$ is given in Figure 2.3.43. We observe that the stability region of the exponential Euler coincides with the stability region of the ordinary Euler for $w_{im} = 0$.

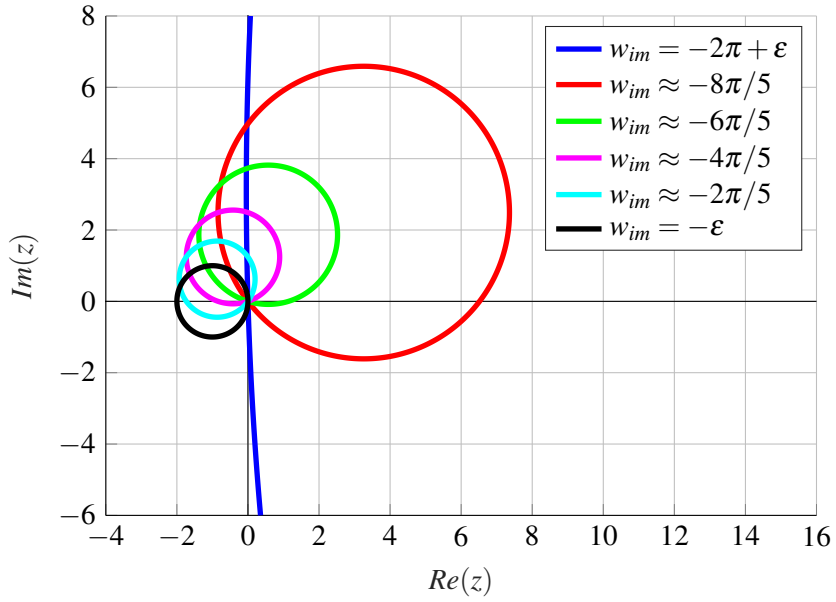


Figure 2.3.43: Behavior of the Stability regions for the interval $[-2\pi, 0]$

In total, the movement of the center of the circles is given in Figure 2.3.44 for w_{im} from -6π to 0. For positive w_{im} this behavior is mirrored at the real axis.

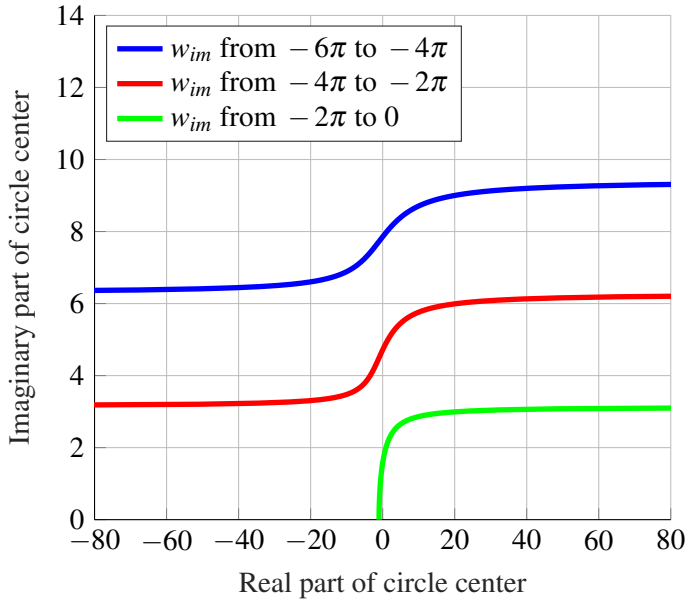


Figure 2.3.44: Plot of evolution of circle centers

Now we briefly consider the case that w is a real number. Hence, we consider $w := w_{re} \in \mathbb{R}$ and $w_{im} = 0$.

Lemma 2.3.45.
For $w_{im} = 0$, the stability circles have their center at $\left(\frac{w_{re}e^{w_{re}}}{1-e^{w_{re}}}, 0\right) \in \mathbb{C}$ and their radius is $\frac{|w_{re}|}{1-e^{w_{re}}}$.

Proof.
Using Theorem 2.3.29, setting $w_{im} = 0$ and using $\cos(w_{im}) = \cos(0) = 1$, $\sin(w_{im}) = \sin(0) = 0$, we obtain for the real coordinate

$$\frac{w_{re}(e^{w_{re}} - e^{2w_{re}})}{1 - 2e^{w_{re}} + e^{2w_{re}}} = \frac{w_{re}e^{w_{re}}(1 - e^{w_{re}})}{(1 - e^{w_{re}})^2} = \frac{w_{re}e^{w_{re}}}{1 - e^{w_{re}}},$$

whereas we obtain 0 for the imaginary coordinate. The radius is given by $\frac{1}{\beta} = \frac{|w_{re}|}{1-e^{w_{re}}}$. □

In this subsection, we saw that the stability regions of the exponential Euler method are circles. Their radius and center can be determined with respect to w. We determined that the stability circles completely move to the right side of the imaginary axis when $w_{im} < 0$ is close to $-2k\pi + \varepsilon$, $k \in \mathbb{N}$. Furthermore, the stability regions grow if w contains a negative real part.

2.3.4. Second order

Two-stage EERK methods are given by the Butcher-Tableau

$$\begin{array}{c|cc} 0 & & \\ c_2 & \mathbf{a}_{21} & \\ \hline & \mathbf{b}_1 & \mathbf{b}_2 \end{array},$$

and the according algorithm reads

Algorithm 2.3.46: Two-Stage Explicit Exponential Runge-Kutta Method

Input : $\mathbf{x}_0, T, N, S, \tilde{f}, \mathbf{a}_{21}(hS), \mathbf{b}_1(hS), \mathbf{b}_2(hS)$
Output: $\mathbf{x}$

$\mathbf{x} \leftarrow \mathbf{x}_0$
$h \leftarrow \frac{T}{N}$
for $\ell \leftarrow 1$ *to* $N+1$ **do**
    $\widetilde{F}_1 \leftarrow \tilde{f}(t, \mathbf{x})$;
    $\widetilde{F}_2 \leftarrow \tilde{f}\big(t + c_2 h, \mathbf{x} + h\mathbf{a}_{21}(hS) \cdot (\widetilde{F}_1 + S\mathbf{x})\big)$;
    $\mathbf{x} \leftarrow \mathbf{x} + h \cdot [\mathbf{b}_1(hS)(\widetilde{F}_1 + S\mathbf{x}) + \mathbf{b}_2(hS)(\widetilde{F}_2 + S\mathbf{x})]$;
    $t \leftarrow t + h$;
end

2.3.4.1. Consistency

Lemma 2.3.47 (cf. [HO10] (Table 2.2, Numbers 1–3))**.**
The two-stage explicit exponential method is consistent of second order if

1. $\psi_1(hS) = 0$,
2. $\psi_2(hS) = 0$,
3. $\psi_{1,i}(hS) = 0$.

Proof.
Using the definitions in (2.3.9), we obtain the conditions as

1.

$$\begin{aligned} \psi_1(hS) &= \varphi_1(hS) - \sum_{i=1}^{2} \mathbf{b}_i(hS) \frac{c_i^{1-1}}{(1-1)!} \\ &= \varphi_1(hS) - \mathbf{b}_1(hS) - \mathbf{b}_2(hS) \\ \Leftrightarrow\ \mathbf{b}_1(hS) + \mathbf{b}_2(hS) &= \varphi_1(hS). \end{aligned} \tag{2.3.48}$$

2.

$$\begin{aligned} \psi_2(hS) &= \varphi_2(hS) - \sum_{i=1}^{2} \mathbf{b}_i(hS) \frac{c_i^{2-1}}{(2-1)!} \\ &= \varphi_2(hS) - \mathbf{b}_1(hS) \cdot c_1 - \mathbf{b}_2(hS) \cdot c_2 \\ &\overset{c_1=0}{=} \varphi_2(hS) - \mathbf{b}_2(hS) \cdot c_2 \\ \Leftrightarrow \mathbf{b}_2(hS) &= \frac{1}{c_2} \varphi_2(hS). \end{aligned} \tag{2.3.49}$$

3. Since $\mathbf{a}_{21}(hS)$ is the only non-zero coefficient of type $\mathbf{a}_{i,j}(hS)$, we only need to consider $\psi_{1,2}(hS)$:

$$\begin{aligned} \psi_{1,2}(hS) &= \varphi_{1,2}(hS) c_2^1 - \sum_{j=1}^{2-1} \mathbf{a}_{2,j}(hS) \frac{c_j^{1-1}}{(1-1)!} \\ &= \varphi_1(c_2 hS) c_2 - \mathbf{a}_{21}(hS) \\ \mathbf{a}_{21}(hS) &= c_2 \varphi_1(c_2 hS). \end{aligned} \tag{2.3.50}$$

In order to observe the consistency error, we need to compare the Taylor expansion of the difference operator with the Taylor expansion of the increment function $\Phi(t, \mathbf{x})$. First, we calculate the

Taylor expansion of the difference operator up to second order:

$$\begin{aligned}
\mathbf{x}'(t) &= \frac{\mathbf{x}(t+h)-\mathbf{x}(t)}{h} \\
&= \mathbf{x}'(t)+\tfrac{h}{2}\mathbf{x}''(t)+O(h^2) \\
&= f(t,\mathbf{x})+\tfrac{h}{2}\left[\tfrac{\partial f}{\partial t}(t,\mathbf{x})+\tfrac{\partial f}{\partial \mathbf{x}}(t,\mathbf{x})\cdot f(t,\mathbf{x})\right]+O(h^2) \\
&= S\mathbf{x}+\tilde{f}(t,\mathbf{x})+h\left[\tfrac{1}{2}\tfrac{\partial \tilde{f}}{\partial t}(t,\mathbf{x})+\tfrac{1}{2}\left(S+\tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x})\right)\cdot\left(S\mathbf{x}+\tilde{f}(t,\mathbf{x})\right)\right]+O(h^2).
\end{aligned}$$

The increment function $\Phi(t,\mathbf{x})$ reads:

$$\begin{aligned}
\Phi(t,\mathbf{x}) &= \mathbf{b}_1(hS)(\widetilde{F}_1+S\mathbf{x})+\mathbf{b}_2(hS)(\widetilde{F}_2+S\mathbf{x}) \\
&\overset{(2.3.48)}{\underset{(2.3.49)}{=}} \left(\varphi_1(hS)-\tfrac{1}{c_2}\varphi_2(hS)\right)(\tilde{f}(t,\mathbf{x})+S\mathbf{x}) \\
&\qquad +\tfrac{1}{c_2}\varphi_2(hS)(\tilde{f}\big(t+c_2h,\mathbf{x}+ha_{21}(hS)\cdot(\widetilde{F}_1+S\mathbf{x})\big)+S\mathbf{x}) \\
&\overset{(2.3.50)}{=} \left(\varphi_1(hS)-\tfrac{1}{c_2}\varphi_2(hS)\right)(\tilde{f}(t,\mathbf{x})+S\mathbf{x}) \\
&\qquad +\tfrac{1}{c_2}\varphi_2(hS)(\tilde{f}\big(t+c_2h,\mathbf{x}+hc_2\varphi_1(c_2hS)\cdot(\tilde{f}(t,\mathbf{x})+S\mathbf{x})\big)+S\mathbf{x})
\end{aligned}$$

with

$$\begin{aligned}
\widetilde{F}_1 &= \tilde{f}(t,\mathbf{x}), \\
\widetilde{F}_2 &= \tilde{f}\big(t+c_2h,\mathbf{x}+h\mathbf{a}_{21}(hS)\cdot(\widetilde{F}_1+S\mathbf{x})\big)=\tilde{f}\big(t+c_2h,\mathbf{x}+h\mathbf{a}_{21}(hS)\cdot(\tilde{f}(t,\mathbf{x})+S\mathbf{x})\big).
\end{aligned}$$

In order to simplify the notation, we start by calculating second order approximations of the coefficients $\varphi_1(hS)$, $\varphi_2(hS)$ and $\varphi_1(c_2hS)$:

$$\begin{aligned}
\varphi_1(hS) &= \int_0^1 e^{h(1-\theta)S}d\theta=\int_0^1 I+h(1-\theta)S+O(h^2)d\theta \\
&\overset{\text{Lemma A.1.7}}{=} \left[\theta I+h\left(\theta-\tfrac{1}{2}\theta^2\right)S\right]_0^1+O(h^2)=I+\tfrac{1}{2}hS+O(h^2) \\
\varphi_2(hS) &= \int_0^1 e^{h(1-\theta)S}\cdot\theta d\theta=\int_0^1 (I+h(1-\theta)S+O(h^2))\cdot\theta d\theta \\
&\overset{\text{Lemma A.1.7}}{=} \left[\tfrac{1}{2}I\theta^2+h\left(\tfrac{1}{2}\theta^2-\tfrac{1}{3}\theta^3\right)S\right]_0^1+O(h^2)=\tfrac{1}{2}I+\tfrac{1}{6}hS+O(h^2) \\
\varphi_1(c_2hS) &= \int_0^1 e^{c_2h(1-\theta)S}d\theta=\int_0^1 I+c_2h(1-\theta)S+O(h^2)d\theta \\
&\overset{\text{Lemma A.1.7}}{=} \left[\theta I+c_2h\left(\theta-\tfrac{1}{2}\theta^2\right)S\right]_0^1+O(h^2)=I+\tfrac{1}{2}c_2hS+O(h^2).
\end{aligned}$$

Subsequently, the coefficient $\varphi_1(hS) - \frac{1}{c_2}\varphi_2(hS)$ in $\Phi(t,x)$ simplifies to

$$\begin{aligned}\varphi_1(hS) - \tfrac{1}{c_2}\varphi_2(hS) &= I + \tfrac{1}{2}hS + O(h^2) - \tfrac{1}{c_2}\left(\tfrac{1}{2}I + \tfrac{1}{6}hS + O(h^2)\right)\\ &= \left(1 - \tfrac{1}{2c_2}\right)I + \left(\tfrac{1}{2} - \tfrac{1}{6c_2}\right)hS + O(h^2).\end{aligned}$$

Also, we have

$$h \cdot \varphi_1(c_2 hS) = Ih + O(h^2).$$

The Taylor expansion of the increment function $\Phi(t,\mathbf{x})$ then reads

$$\begin{aligned}\Phi(t,\mathbf{x}) = &\left[\left(1 - \tfrac{1}{2c_2}\right)I + \left(\tfrac{1}{2} - \tfrac{1}{6c_2}\right)hS\right](\tilde{f}(t,\mathbf{x}) + S\mathbf{x})\\ &+ \tfrac{1}{c_2}\left(\tfrac{1}{2}I + \tfrac{1}{6}hS\right)\left[\tilde{f}(t,\mathbf{x}) + c_2 h\tfrac{\partial \tilde{f}}{\partial t} + \tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x}) \cdot hc_2\varphi_1(c_2hS)\cdot(\tilde{f}(t,\mathbf{x}) + S\mathbf{x}) + S\mathbf{x}\right] + O(h^2)\\
= &\left[\left(1 - \tfrac{1}{2c_2}\right)I + \tfrac{1}{2c_2}I\right](\tilde{f}(t,\mathbf{x}) + S\mathbf{x}) + \left[\left(\tfrac{1}{2} - \tfrac{1}{6c_2}\right)hS + \tfrac{1}{6c_2}hS\right](\tilde{f}(t,\mathbf{x}) + S\mathbf{x})\\
&+ \tfrac{1}{2c_2}I\left[c_2 h\tfrac{\partial \tilde{f}}{\partial t} + hc_2\tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x})\cdot(\tilde{f}(t,\mathbf{x}) + S\mathbf{x})\right] + O(h^2)\\
= &\tilde{f}(t,\mathbf{x}) + S\mathbf{x} + \tfrac{1}{2}hS(\tilde{f}(t,\mathbf{x}) + S\mathbf{x}) + \tfrac{1}{2}I\left[h\tfrac{\partial \tilde{f}}{\partial t} + h\tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x})\cdot(\tilde{f}(t,\mathbf{x}) + S\mathbf{x})\right] + O(h^2)\\
= &\tilde{f}(t,\mathbf{x}) + S\mathbf{x} + h\left[\tfrac{1}{2}\tfrac{\partial \tilde{f}}{\partial t} + \tfrac{1}{2}\left(S + \tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x})\right)\cdot(\tilde{f}(t,\mathbf{x}) + S\mathbf{x})\right] + O(h^2).\end{aligned}$$

Since $\Phi(t,\mathbf{x}) - \mathbf{x}'(t) = O(h^2)$, the method is convergent of second order. □

Abiding by the conditions 1.–3. of Lemma 2.3.47, we obtain a family of second order EERK methods (see [HO10] (Example 2.18)) which is given by the Butcher-Tableau

$$\begin{array}{c|cc} 0 & & \\ c_2 & c_2\varphi_{1,2} & \\ \hline & \varphi_1 - \frac{1}{c_2}\varphi_2 & \frac{1}{c_2}\varphi_2 \end{array}.$$

Corollary 2.3.51 (cf. [HO10] (Example 2.6))**.**
The easiest choice is $c_2 = 1$ which leads to the explicit exponential trapezoidal rule given by the Butcher-Tableau

$$\begin{array}{c|cc} 0 & & \\ 1 & \varphi_1 & \\ \hline & \varphi_1 - \varphi_2 & \varphi_2 \end{array}.$$

2.3.5. Combined Stability Plots

We already know the explicit exponential Runge-Kutta methods of orders 1 and 2. In this subsection we want to compare the stability regions of exponential Runge-Kutta methods of order 1–4 to the stability plot of ordinary Runge-Kutta methods in Figure 2.1.19. For a comparison, we still need to mention the EERK methods of orders 3 and 4. We provide the respective Butcher-Tableaus without further analysis here. Consistency results and derivations can be found in [HO05a, HO10].

There are various families of third order methods. An overview is given in [HO05a] (Section 5.2). We use the ETD3RK method ([HO05a], (Equation (5.13))) with the Butcher-Tableau

$$\begin{array}{c|ccc} 0 & & & \\ \frac{1}{2} & \frac{1}{2}\varphi_{1,2}(hS) & & \\ 1 & -\varphi_{1,3}(hS) & 2\varphi_{1,3}(hS) & \\ \hline & 4\varphi_3(hS)-2\varphi_2(hS)+\varphi_1(hS) & -8\varphi_3(hS)+4\varphi_2(hS) & 4\varphi_3(hS)-\varphi_2(hS) \end{array}.$$

There are also numerous fourth-order methods. We use the following five-stage fourth order method [HO10] (p.229):

$$\begin{array}{c|ccccc} 0 & & & & & \\ \frac{1}{2} & \frac{1}{2}\varphi_{1,2} & & & & \\ \frac{1}{2} & \frac{1}{2}\varphi_{1,3}-\varphi_{2,3} & \varphi_{2,3} & & & \\ 1 & \varphi_{1,4}-2\varphi_{2,4} & \varphi_{2,4} & \varphi_{2,4} & & \\ \frac{1}{2} & \frac{1}{2}\varphi_{1,5}-2\mathbf{a}_{5,2}-\mathbf{a}_{5,4} & \mathbf{a}_{5,2} & \mathbf{a}_{5,2} & \frac{1}{4}\varphi_{2,5}-\mathbf{a}_{5,2} & \\ \hline & \varphi_1-3\varphi_2+4\varphi_3 & 0 & 0 & -\varphi_2+4\varphi_3 & 4\varphi_2-8\varphi_3 \end{array}$$

with

$$\mathbf{a}_{5,2} := \frac{1}{2}\varphi_{2,5}-\varphi_{3,4}+\frac{1}{4}\varphi_{2,4}-\frac{1}{2}\varphi_{3,5}.$$

We start with the stability regions where w is equal to zero in Figure 2.3.52. The stability regions of the exponential Runge-Kutta methods are exactly the stability regions of the ordinary Runge-Kutta methods (see Figure 2.1.19) in this case.

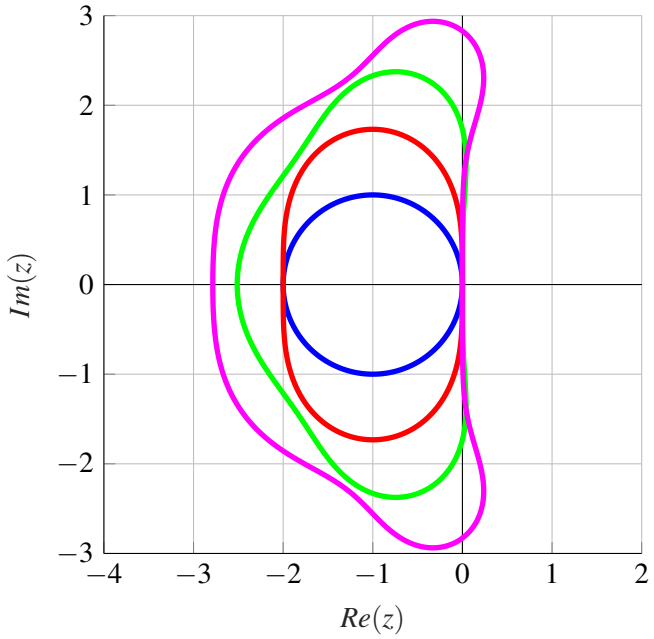


Figure 2.3.52: Plot of stability regions of 1^{st} (blue), 2^{nd} (red), 3^{rd} (green) and 4^{th} (pink) order exponential methods for $w = 0$ (author's plot)

2.3.5.1. Varying imaginary part of w

With varying w_{im} the appearance of the stability regions changes. For $w_{im} = -6$, we are already quite close to -2π, which is a singularity for the exponential Euler as we saw in Subsection 2.3.3.2. It also seems to be a critical value for the other methods as we can see in Figure 2.3.53d. For a purely imaginary w close to 2π (Figure 2.3.53d), we need very small step sizes for the methods of orders 1, 3 and 4. Especially, if we expect our system to have complex conjugate eigenvalues with high modulus, we can only count on the second order method whose stability region covers the same parts of the positive and the negative imaginary axis.

We also notice a part of the imaginary axis (circa from 1 to 2.5), where the methods of third and fourth order are unstable but both the methods of first and second order are stable in Figure 2.3.53b. Additionally, we see that the stability regions of the third and fourth order methods split into several disjoint regions with growing modulus of w_{im}. In Section 3.3 we look into these peculiarities.

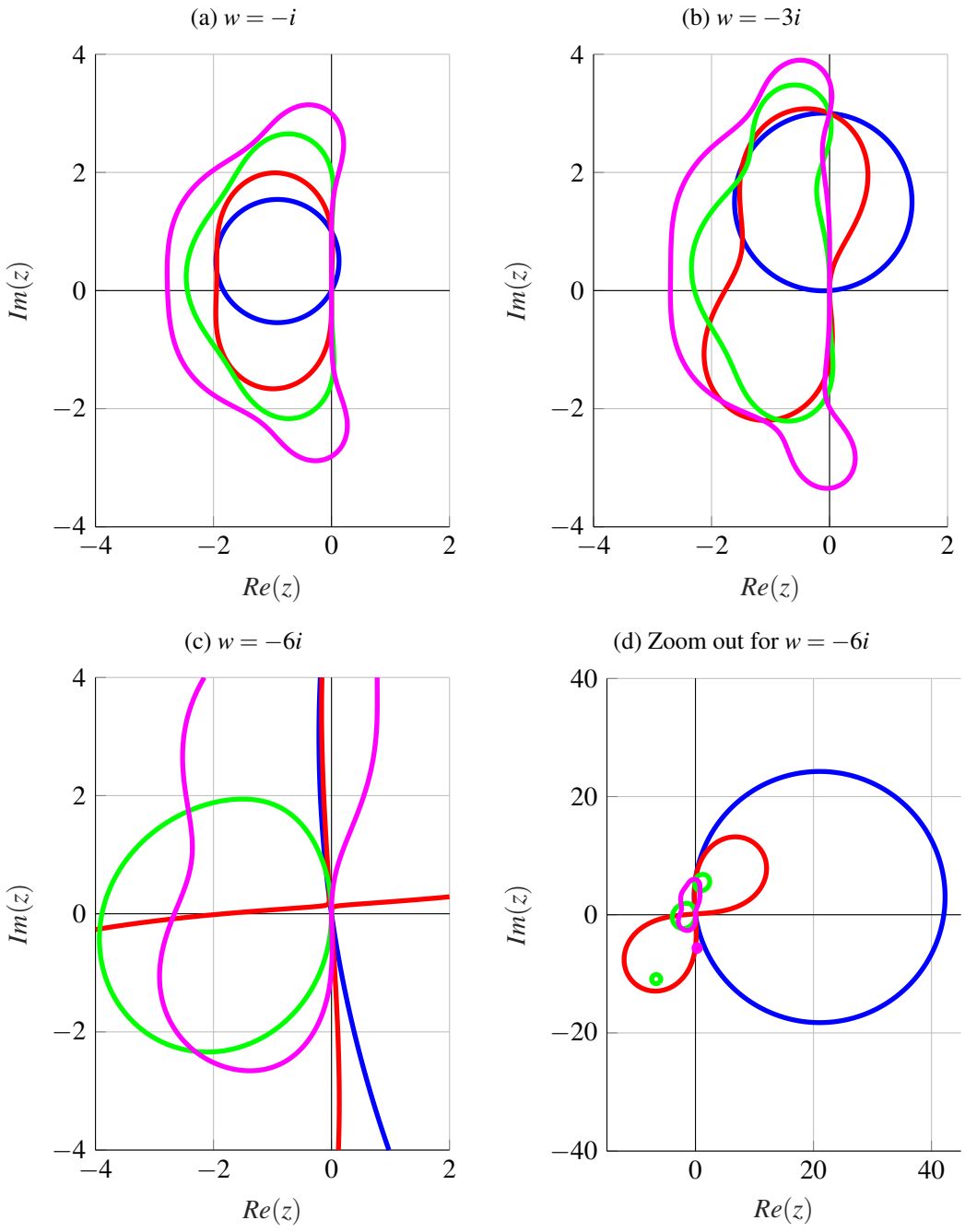


Figure 2.3.53: Plot of stability regions of 1^{st} (blue), 2^{nd} (red), 3^{rd} (green) and 4^{th} (pink) order exponential methods for $w_{re} = 0$ and different w_{im} (author's plots)

2.3.5.2. Varying real part of w

In the case of a purely real w, we notice that damping increases not only the stability region of the exponential Euler method (cf. Lemma 2.3.30), but also the stability regions of all methods. It is interesting to see that the stability region of the second order method grows most. However, the stability regions of the first and second order methods approach each other as we can see in Figure 2.3.54d. Section 3.3 provides test cases for this situation.

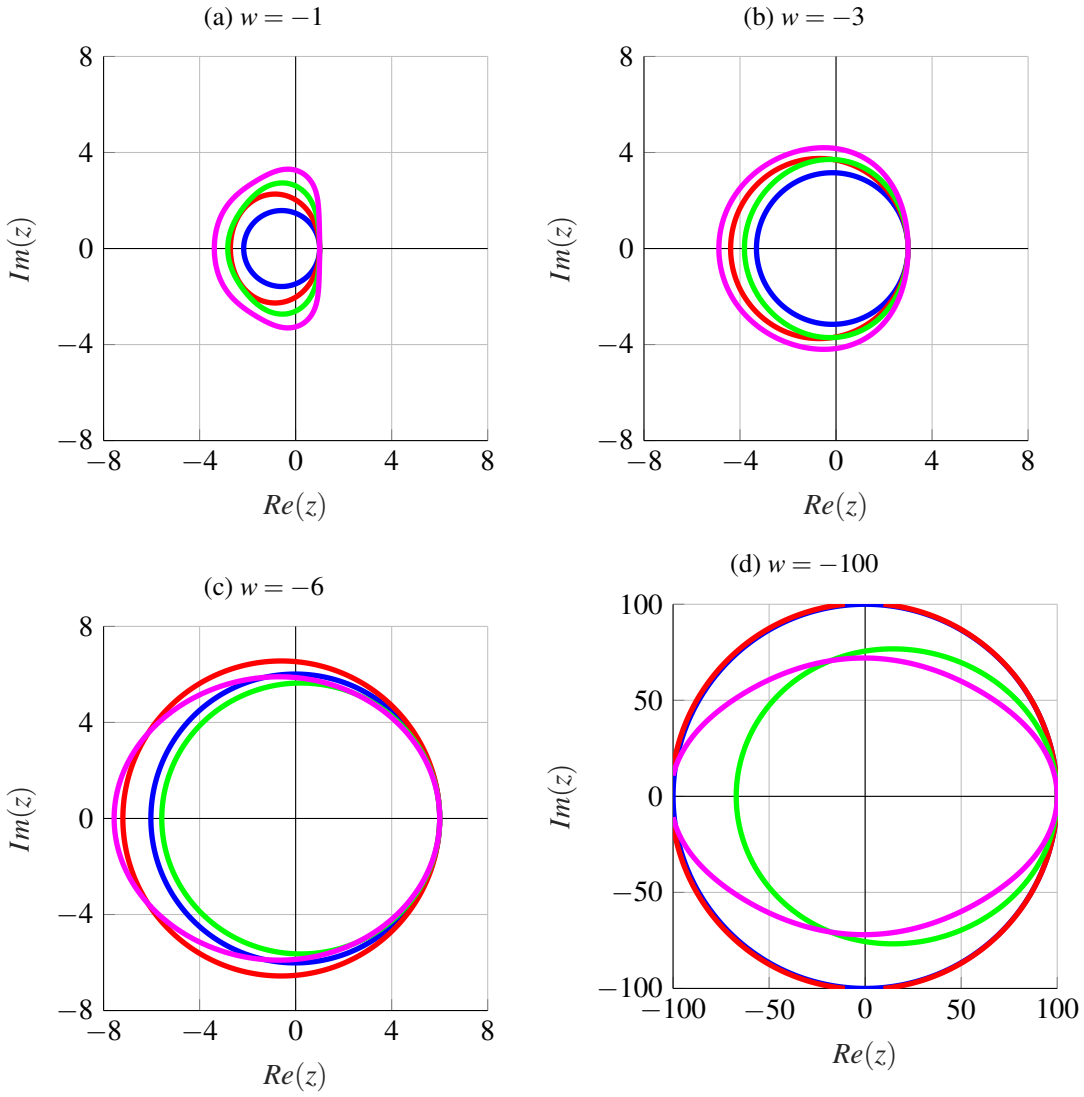


Figure 2.3.54: Plot of stability regions of 1^{st} (blue), 2^{nd} (red), 3^{rd} (green) and 4^{th} (pink) order exponential methods for $w_{im} = 0$ and different w_{re} (author's plots)

2.3.5.3. Varying w

We now vary both the real and the imaginary parts of w. Once more, the second order method yields the biggest stability regions, whereas the third and fourth order methods' stability regions are smaller. Furthermore, the stability regions of the first three methods tend towards a circle shape, whereas the fourth order method takes a more elliptic shape.

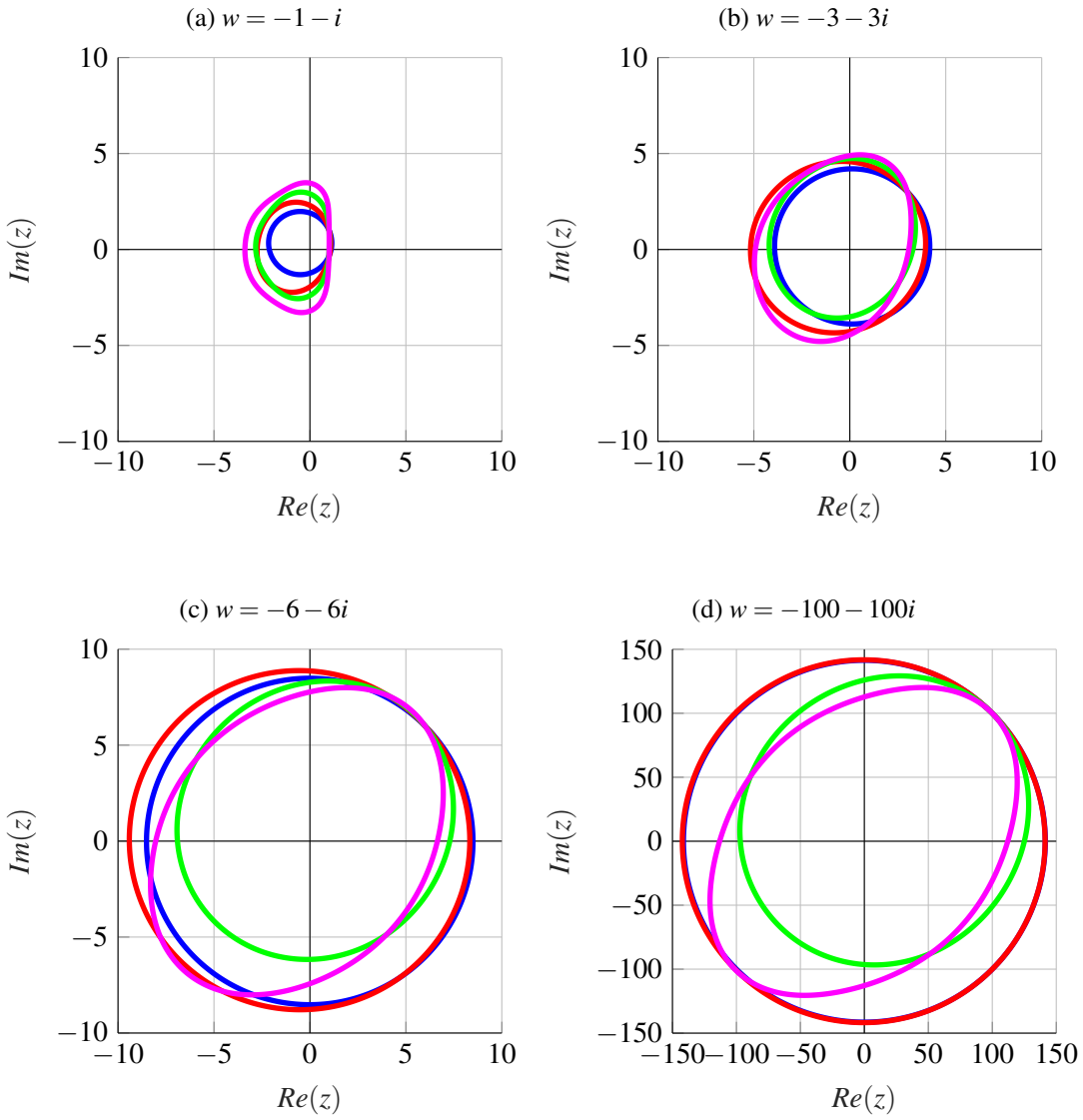


Figure 2.3.55: Plot of stability regions of 1^{st} (blue), 2^{nd} (red), 3^{rd} (green) and 4^{th} (pink) order exponential methods for different w (author's plots)

2.4. Half-Explicit Exponential Runge-Kutta (HEERK) methods

In analogy to HERK methods, we want to construct and analyze Half-Explicit Exponential Runge-Kutta (HEERK) methods. Similarly, we need to insert a Newton call in every internal stage of the Runge-Kutta method and obtain

$$\begin{aligned}
\mathbf{x}_{n+1} &= \mathbf{x}_n + h\sum_{i=1}^{s} \mathbf{b}_i(hS)(\widetilde{F}_{ni} + S\mathbf{x}_n),\\
X_{ni} &= \mathbf{x}_n + h\sum_{j=1}^{s} \mathbf{a}_{ij}(hs)(\widetilde{F}_{nj} + S\mathbf{x}_n),\\
Y_{ni} &= \text{Newton}(t_n + c_i h, X_{ni}, Y_{n,i-1}, g, \partial_{\mathbf{y}} g, O(\varepsilon)),\\
\widetilde{F}_{nj} &= \tilde{f}(t_n + c_j h, X_{nj}, Y_{nj}).
\end{aligned}$$

2.4.1. First order consistency

Inserting a Newton call into Algorithm 2.3.16, we obtain a one-stage half-explicit exponential Runge-Kutta method.

Algorithm 2.4.1: One-Stage Half-Explicit Exponential Runge-Kutta Method

Input : $\mathbf{x}_0, \mathbf{y}_0, T, N, S, \tilde{f}, g, \partial_{\mathbf{y}} g, \mathbf{b}_1(hS)$

Output: $\mathbf{x}, \mathbf{y}$

1 $\mathbf{x} \leftarrow \mathbf{x}_0$

2 $\mathbf{y} \leftarrow \mathbf{y}_0$

3 $h \leftarrow \frac{T}{N}$

4 **for** $j \leftarrow 1$ *to* $N+1$ **do**

5 $\quad \mathbf{y} \leftarrow \text{Newton}(t, \mathbf{x}, \mathbf{y}, g, \partial_{\mathbf{y}} g, \varepsilon(h))$;

6 $\quad \widetilde{F} \leftarrow \tilde{f}(t, \mathbf{x}, \mathbf{y})$;

7 $\quad \mathbf{x} \leftarrow \mathbf{x} + h \cdot \mathbf{b}_1(hS) \cdot (\widetilde{F} + S\mathbf{x})$;

8 $\quad t \leftarrow t + h$;

9 **end**

We now analyze the consistency of this algorithm.

Lemma 2.4.2 (Author's contribution)**.**
The half-explicit exponential Euler method is consistent of first order if the conditions of Lemma 2.3.17 are fulfilled ($\psi_1(hS) = 0$), and the Newton method is solved to an accuracy of $O(h)$ in line 5 when we apply Algorithm 2.4.1 to ($\star\star\star$).

Proof.
From Lemma 2.3.17, we know that the condition $\psi_1(hS) = 0$ simplifies to $\mathbf{b}_1(hS) = \varphi_1(hS)$ for the explicit exponential Euler method.

We consider the system

$$\begin{cases} \dot{\mathbf{x}}(t) &= S\mathbf{x} + \tilde{f}(t,\mathbf{x}(t),\mathbf{y}(t)) \\ \mathbf{y}(t) &= g(t,\mathbf{x}(t),\mathbf{y}(t)) \end{cases} . \qquad (\star\star\star)$$

Locally, we have a unique solution

$$\mathbf{y}(t) = \bar{g}^{-1}(0;t,\mathbf{x}(t))$$

as in (1.1.3).

We commit an error when applying the Newton method. Hence, we only get an inexact version $\tilde{g}^{-1}$ instead of $\bar{g}^{-1}$ for $\mathbf{y}$. Hence, we get a defective $\tilde{\mathbf{y}}$, which has an error $\triangle\mathbf{y}$ in comparison to the correct $\mathbf{y}$:

$$\tilde{\mathbf{y}} := \tilde{g}^{-1}(0;t,\mathbf{x}) = \bar{g}^{-1}(0;t,\mathbf{x}) + \triangle\mathbf{y} = \mathbf{y} + \triangle\mathbf{y}.$$

In order to observe the consistency error, we need to compare the Taylor expansion of the difference operator with the Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$. First, we calculate the Taylor expansion of the difference operator up to first order:

$$\begin{aligned} \mathbf{x}'(t) &= \frac{\mathbf{x}(t+h)-\mathbf{x}(t)}{h} \\ &= \mathbf{x}'(t) + O(h) \\ &= S\mathbf{x} + \tilde{f}(t,\mathbf{x},\mathbf{y}) + O(h). \end{aligned}$$

The increment function $\Phi(t,\mathbf{x},\mathbf{y})$ reads:

$$\Phi(t,\mathbf{x},\mathbf{y}) = \mathbf{b}_1(hS)(\widetilde{F}(t,\mathbf{x},\tilde{\mathbf{y}}) + S\mathbf{x}) \overset{(2.3.18)}{=} \varphi_1(hS)(\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}) + S\mathbf{x}).$$

We know that the exponential function in $\varphi_1(hS)$ (see Definition (2.3.9a)) has a power series representation (see Lemma A.1.6) as

$$\varphi_1(hS) = \int_0^1 e^{h(1-\theta)S} d\theta = \int_0^1 \sum_{k=0}^{\infty} \frac{(h(1-\theta)S)^k}{k!} d\theta.$$

By Theorem A.1.7 and Remark A.1.8, we can interchange summation and integration in the following Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$:

$$\begin{aligned} \Phi(t,\mathbf{x},\mathbf{y}) &= \varphi_1(hS)(\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}) + S\mathbf{x}) \\ &= \varphi_1(hS)(\tilde{f}(t,\mathbf{x},\mathbf{y}+\triangle\mathbf{y}) + S\mathbf{x}) \end{aligned}$$

$$
\begin{aligned}
&= \int_0^1 \sum_{k=0}^{\infty} \frac{(h(1-\theta)S)^k}{k!} d\theta \cdot \left(\tilde{f}(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y}) \cdot \triangle \mathbf{y} + O(h^2) + S\mathbf{x}\right) \\
&= \int_0^1 I + h \cdot \sum_{k=1}^{\infty} \frac{h^{k-1}((1-\theta)S)^k}{k!} d\theta \cdot \left(\tilde{f}(t,\mathbf{x},\mathbf{y}) + O(h) + S\mathbf{x}\right) \\
&= \left[\int_0^1 Id\theta + O(h)\right] \left(\tilde{f}(t,\mathbf{x},\mathbf{y}) + O(h) + S\mathbf{x}\right) \\
&= \tilde{f}(t,\mathbf{x},\mathbf{y}) + S\mathbf{x} + O(h).
\end{aligned}
$$

Since $\Phi(t,\mathbf{x},\mathbf{y}) - \mathbf{x}'(t) = O(h)$, the method is convergent of first order. □

2.4.2. Second order consistency

Inserting a Newton call before every stage in Algorithm 2.3.46, we obtain a two-stage half-explicit exponential Runge-Kutta method.

Algorithm 2.4.3: Two-Stage Half-Explicit Exponential Runge-Kutta Method

Input : $\mathbf{x}_0, \mathbf{y}_0, T, N, S, \tilde{f}, g, \partial_{\mathbf{y}} g, \mathbf{a}_{21}(hS), \mathbf{b}_1(hS), \mathbf{b}_2(hS)$

Output: $\mathbf{x}, \mathbf{y}$

1 $\mathbf{x} \leftarrow \mathbf{x}_0$

2 $\mathbf{y} \leftarrow \mathbf{y}_0$

3 $h \leftarrow \frac{T}{N}$

4 **for** $\ell \leftarrow 1$ *to* $N+1$ **do**

5 $\quad \mathbf{y} \leftarrow \text{Newton}(t,\mathbf{x},\mathbf{y},g,\partial_{\mathbf{y}} g,\varepsilon(h))$;

6 $\quad \widetilde{F}_1 \leftarrow \tilde{f}(t,\mathbf{x},\mathbf{y})$;

7 $\quad \mathbf{y} \leftarrow \text{Newton}\big(t + c_2 h, \mathbf{x} + h\mathbf{a}_{21}(hS) \cdot (\widetilde{F}_1 + S\mathbf{x}), \mathbf{y}, g, \partial_{\mathbf{y}} g, \varepsilon(h)\big)$;

8 $\quad \widetilde{F}_2 \leftarrow \tilde{f}\big(t + c_2 h, \mathbf{x} + h\mathbf{a}_{21}(hS) \cdot (\widetilde{F}_1 + S\mathbf{x}), \mathbf{y}\big)$;

9 $\quad \mathbf{x} \leftarrow \mathbf{x} + h \cdot \big[\mathbf{b}_1(hS)(\widetilde{F}_1 + S\mathbf{x}) + \mathbf{b}_2(hS)(\widetilde{F}_2 + S\mathbf{x})\big]$;

10 $\quad t \leftarrow t + h$;

11 **end**

Lemma 2.4.4 (Author's contribution)**.**
The two-stage half-explicit exponential method is consistent of second order if the conditions of Lemma 2.3.47 are fulfilled ($\psi_1(hS) = 0$, $\psi_2(hS) = 0$, $\psi_{1,i}(hS) = 0$), and the Newton method is solved to an accuracy of $O(h^2)$ in lines 5 and 7 when we apply Algorithm 2.4.3 to ($\star\star\star$).

Proof.
From the proof of Lemma 2.3.47, we know that the required conditions simplify to

1. $\mathbf{b}_1(hS)+\mathbf{b}_2(hS)=\varphi_1(hS)$,
2. $\mathbf{b}_2(hS)=\frac{1}{c_2}\varphi_2(hS)$,
3. $\mathbf{a}_{21}(hS)=c_2\varphi_1(c_2hS)$

for an explicit exponential two-stage method.

We consider the system

$$\left\{\begin{array}{lcl}\dot{\mathbf{x}}(t) &=& S\mathbf{x}+\tilde{f}(t,\mathbf{x}(t),\mathbf{y}(t))\\ \mathbf{y}(t) &=& g(t,\mathbf{x}(t),\mathbf{y}(t))\end{array}\right. . \qquad (\star\star\star)$$

Locally, we have a unique solution

$$\mathbf{y}(t)=\bar{g}^{-1}(0;t,\mathbf{x}(t))$$

as in (1.1.3).

However, we commit an error when applying the Newton method. Hence, we only get an inexact version $\tilde{g}^{-1}$ instead of $\bar{g}^{-1}$ for $\mathbf{y}$. In line 5, we obtain a defective $\tilde{\mathbf{y}}_1$, which has an error $\triangle\mathbf{y}_1$ in comparison to the correct $\mathbf{y}$:

$$\tilde{\mathbf{y}}_1:=\tilde{g}^{-1}(0;t,\mathbf{x})=\bar{g}^{-1}(0;t,\mathbf{x})+\triangle\mathbf{y}_1=\mathbf{y}+\triangle\mathbf{y}_1.$$

For the Newton call in line 7, we already use our defective $\tilde{\mathbf{y}}_1$ from the first Newton call. We need to compute $\mathbf{y}$ corresponding to $t=t+c_2h$ and $\mathbf{x}=\mathbf{x}+h\mathbf{a}_{21}(hS)f(t,\mathbf{x},\tilde{\mathbf{y}}_1)$. We denote the occurring error by $\triangle\mathbf{y}_2$ and obtain the defective $\tilde{\mathbf{y}}_2$ as:

$$\begin{aligned}\tilde{\mathbf{y}}_2 &:= \tilde{g}^{-1}(0;t+c_2h,\mathbf{x}+h\mathbf{a}_{21}(hS)f(t,\mathbf{x},\tilde{\mathbf{y}}_1))\\ &= \bar{g}^{-1}(0;t+c_2h,\mathbf{x}+h\mathbf{a}_{21}(hS)f(t,\mathbf{x},\tilde{\mathbf{y}}_1))+\triangle\mathbf{y}_2.\end{aligned}$$

In order to observe the consistency error, we need to compare the Taylor expansion of the difference operator with the Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$.

Using (2.2.12), we calculate the Taylor expansion of the difference operator up to second order:

$$\begin{aligned}\mathbf{x}'(t) &= \frac{\mathbf{x}(t+h)-\mathbf{x}(t)}{h}\\ &= \mathbf{x}'(t)+\tfrac{h}{2}\mathbf{x}''(t)+O(h^2)\\ &= f(t,\mathbf{x},\mathbf{y})+\tfrac{h}{2}\left[\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})\cdot f(t,\mathbf{x},\mathbf{y})+\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot\mathbf{y}'(t)\right]+O(h^2)\\ &= f(t,\mathbf{x},\mathbf{y})+h\left[\tfrac{1}{2}\tfrac{\partial f}{\partial t}(t,\mathbf{x},\mathbf{y})+\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{x}}(t,\mathbf{x},\mathbf{y})\cdot f(t,\mathbf{x},\mathbf{y})+\tfrac{1}{2}\tfrac{\partial f}{\partial\mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot\mathbf{y}'(t)\right]+O(h^2)\end{aligned}$$

$$
\begin{aligned}
&= S\mathbf{x} + \tilde{f}(t,\mathbf{x},\mathbf{y}) \\
&\quad + h\left[\tfrac{1}{2}\tfrac{\partial \tilde{f}}{\partial t}(t,\mathbf{x},\mathbf{y}) + \tfrac{1}{2}\left(S + \tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y})\right)\cdot\left(S\mathbf{x} + \tilde{f}(t,\mathbf{x},\mathbf{y})\right) + \tfrac{1}{2}\tfrac{\partial f}{\partial y}(t,\mathbf{x},\mathbf{y})\cdot\mathbf{y}'(t)\right] + O(h^2).
\end{aligned}
$$

The increment function $\Phi(t,\mathbf{x},\mathbf{y})$ reads:

$$
\begin{aligned}
\Phi(t,\mathbf{x},\mathbf{y}) &= \mathbf{b}_1(hS)(\widetilde{F}_1 + S\cdot\mathbf{x}) + \mathbf{b}_2(hS)(\widetilde{F}_2 + S\mathbf{x}) \\
&\overset{\substack{(2.3.48)\\(2.3.49)}}{=} \left(\varphi_1(hS) - \tfrac{1}{c_2}\varphi_2(hS)\right)\left(\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1) + S\mathbf{x}\right) \\
&\qquad + \tfrac{1}{c_2}\varphi_2(hS)\left(\tilde{f}\left(t + c_2 h, \mathbf{x} + h\mathbf{a}_{21}(hS)\cdot(\widetilde{F}_1 + S\mathbf{x}), \tilde{\mathbf{y}}_2\right) + S\mathbf{x}\right) \\
&\overset{(2.3.50)}{=} \left(\varphi_1(hS) - \tfrac{1}{c_2}\varphi_2(hS)\right)(\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1) + S\mathbf{x}) \\
&\qquad + \tfrac{1}{c_2}\varphi_2(hS)\left(\tilde{f}\left(t + c_2 h, \mathbf{x} + hc_2\varphi_1(c_2 hS)\cdot(\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1) + S\mathbf{x}), \tilde{\mathbf{y}}_2\right) + S\mathbf{x}\right)
\end{aligned}
$$

with

$$
\begin{aligned}
\widetilde{F}_1 &= \tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1), \\
\widetilde{F}_2 &= \tilde{f}\left(t + c_2 h, \mathbf{x} + h\mathbf{a}_{21}(hS)\cdot(\widetilde{F}_1 + S\mathbf{x}), \tilde{\mathbf{y}}_2\right) \\
&= \tilde{f}\left(t + c_2 h, \mathbf{x} + h\mathbf{a}_{21}(hS)\cdot(\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1) + S\mathbf{x}), \tilde{\mathbf{y}}_2\right).
\end{aligned}
$$

As in the proof of Lemma 2.3.47, second order approximations of the needed coefficients are given by

$$
\begin{aligned}
\varphi_1(hS) &= I + \tfrac{1}{2}hS + O(h^2), \\
\varphi_2(hS) &= \tfrac{1}{2}I + \tfrac{1}{6}hS + O(h^2), \\
\varphi_1(c_2 hS) &= I + \tfrac{1}{2}c_2 hS + O(h^2), \\
\varphi_1(hS) - \tfrac{1}{c_2}\varphi_2(hS) &= \left(1 - \tfrac{1}{2c_2}\right)I + \left(\tfrac{1}{2} - \tfrac{1}{6c_2}\right)hS + O(h^2), \\
h\cdot\varphi_1(c_2 hS) &= Ih + O(h^2).
\end{aligned} \tag{2.4.5}
$$

Furthermore,

$$
h\cdot\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1) = h\cdot\left(\tilde{f}(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot\triangle\mathbf{y}_1 + O((\triangle\mathbf{y}_1)^2)\right) = h\cdot\tilde{f}(t,\mathbf{x},\mathbf{y}) + O(h^2). \tag{2.4.6}
$$

In order to keep the calculations neat, we calculate

$$
\begin{aligned}
\tilde{\mathbf{y}}_2 - \mathbf{y} &= \bar{g}^{-1}(0; t + c_2 h, \mathbf{x} + \mathbf{a}_{21} h\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1)) + \triangle\mathbf{y}_2 - \mathbf{y} \\
&\overset{(2.4.6)}{=} \bar{g}^{-1}(0;t,\mathbf{x}) + c_2 h\cdot\tfrac{\partial \bar{g}^{-1}}{\partial t}(0;t,\mathbf{x}) + \tfrac{\partial \bar{g}^{-1}}{\partial \mathbf{x}}(0;t,\mathbf{x})\cdot\mathbf{a}_{21} h\tilde{f}(t,\mathbf{x},\mathbf{y}) + O(h^2) + \triangle\mathbf{y}_2 - \mathbf{y}
\end{aligned}
$$

$$\overset{\text{Cond. 3.}}{=} c_2 h \cdot \tfrac{\partial \bar{g}^{-1}}{\partial t}(0;t,\mathbf{x}) + \tfrac{\partial \bar{g}^{-1}}{\partial \mathbf{x}}(0;t,\mathbf{x}) \cdot c_2 h \varphi_1(c_2 hS)\tilde{f}(t,\mathbf{x},\mathbf{y}) + O(h^2) + \triangle \mathbf{y}_2$$
$$\underset{(2.2.12)}{\overset{(2.4.5)}{=}} c_2 h \mathbf{y}'(t) + \triangle \mathbf{y}_2 + O(h^2). \tag{2.4.7}$$

The Taylor expansion of the increment function $\Phi(t,\mathbf{x},\mathbf{y})$ then reads

$$\begin{aligned}
\Phi(t,\mathbf{x},\mathbf{y}) &= \left[\left(1-\tfrac{1}{2c_2}\right)I + \left(\tfrac{1}{2}-\tfrac{1}{6c_2}\right)hS\right](\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1)+S\mathbf{x}) + \tfrac{1}{c_2}\left(\tfrac{1}{2}I+\tfrac{1}{6}hS\right)\big[\tilde{f}(t,\mathbf{x},\mathbf{y}) \\
&\quad + c_2 h \tfrac{\partial \tilde{f}}{\partial t}(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y}) \cdot (hc_2\varphi_1(c_2hS)\cdot(\tilde{f}(t,\mathbf{x},\tilde{\mathbf{y}}_1)+S\mathbf{x})) \\
&\quad + \tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot(\tilde{\mathbf{y}}_2-\mathbf{y}) + S\mathbf{x}\Big] + O(h^2) \\
&\underset{(2.4.6)}{\overset{(2.4.7)}{=}} \left[\left(1-\tfrac{1}{2c_2}\right)I + \left(\tfrac{1}{2}-\tfrac{1}{6c_2}\right)hS\right]\left(\tilde{f}(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1 + S\mathbf{x}\right) \\
&\quad + \tfrac{1}{c_2}\left(\tfrac{1}{2}I+\tfrac{1}{6}hS\right)\Big[\tilde{f}(t,\mathbf{x},\mathbf{y}) + c_2 h \tfrac{\partial \tilde{f}}{\partial t}(t,x\mathbf{x},\mathbf{y}) + \tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y})\cdot(c_2 Ih\cdot(\tilde{f}(t,\mathbf{x},\mathbf{y})+S\mathbf{x})) \\
&\quad + \tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})(c_2 h\mathbf{y}'(t)+\triangle\mathbf{y}_2) + S\mathbf{x}\Big] + O(h^2) \\
&= \tilde{f}(t,\mathbf{x},\mathbf{y}) + S\mathbf{x} + \left(1-\tfrac{1}{2c_2}\right)\tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1 + \left(\tfrac{1}{2}-\tfrac{1}{6c_2}\right)hS\cdot(\tilde{f}(t,\mathbf{x},\mathbf{y})+S\mathbf{x}) \\
&\quad + \tfrac{1}{2c_2}\Big[c_2 h\tfrac{\partial \tilde{f}}{\partial t}(t,\mathbf{x},\mathbf{y}) + \tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y})(c_2 Ih\cdot(\tilde{f}(t,\mathbf{x},\mathbf{y})+S\mathbf{x})) \\
&\quad + \tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot(c_2 h\mathbf{y}'(t)+\triangle\mathbf{y}_2)\Big] + \tfrac{1}{6c_2}hS\cdot(\tilde{f}(t,\mathbf{x},\mathbf{y})+S\mathbf{x}) + O(h^2) \\
&= \tilde{f}(t,\mathbf{x},\mathbf{y}) + S\mathbf{x} + \tfrac{1}{2}hS\cdot(\tilde{f}(t,\mathbf{x},\mathbf{y})+S\mathbf{x}) \\
&\quad + \tfrac{1}{2}h\tfrac{\partial \tilde{f}}{\partial t}(t,\mathbf{x},\mathbf{y}) + \tfrac{1}{2}h\cdot\tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y})\cdot(\tilde{f}(t,\mathbf{x},\mathbf{y})+S\mathbf{x}) + \tfrac{1}{2}h\tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot\mathbf{y}'(t) \\
&\quad + \left(1-\tfrac{1}{2c_2}\right)\tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1 + \tfrac{1}{2c_2}\tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot\triangle\mathbf{y}_2 + O(h^2) \\
&= \tilde{f}(t,\mathbf{x},\mathbf{y}) + S\mathbf{x} + \tfrac{1}{2}h\tfrac{\partial \tilde{f}}{\partial t}(t,\mathbf{x},\mathbf{y}) \\
&\quad + \tfrac{1}{2}h\cdot\left(\tfrac{\partial \tilde{f}}{\partial \mathbf{x}}(t,\mathbf{x},\mathbf{y}) + S\right)\cdot(\tilde{f}(t,\mathbf{x},\mathbf{y})+S\mathbf{x}) + \tfrac{1}{2}h\tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\cdot\mathbf{y}'(t) \\
&\quad + \left(1-\tfrac{1}{2c_2}\right)\tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_1 + \tfrac{1}{2c_2}\tfrac{\partial \tilde{f}}{\partial \mathbf{y}}(t,\mathbf{x},\mathbf{y})\triangle\mathbf{y}_2 + O(h^2).
\end{aligned}$$

Now if $\triangle\mathbf{y}_1 = O(h^2)$ and $\triangle\mathbf{y}_2 = O(h^2)$, we obtain $\Phi(t,\mathbf{x},\mathbf{y}) - \mathbf{x}'(t) = O(h^2)$ which makes the method convergent of second order. □

3. Numerical results

In this section, we present numerical tests that help us understand the performance of HEERK methods better. Whenever we speak of an approximation error, we mean the root-mean-square (RMS) error; see Definition A.3.5.

To start with, we examine the exponential Euler method in Subsections 3.1 and 3.2. We investigate, how to choose a suitable time width for a certain setting, analyze possible pit falls and compare the performance of the exponential Euler to the ordinary Euler. In Subsection 3.3, we present some results concerning the stability plots from Subsection 2.3.5. Finally in Subsection 3.4, we use our model from Sections 1.2 and C in order to execute an in-depth analysis of the newly defined HEERK methods. Here, we focus on the HEERK4 method and compare its performance with the classical HERK4 algorithm.

3.1. Time width choice for exponential Euler

This numerical test shows how to choose an adequate step size h for given σ and λ. It also deals with the problem when w_{im} is close to $2k\pi$.

We choose $\sigma = -60i$ and $\lambda = -10 - 20i$. Then plotting ϕ with given σ and λ and leaving h as the only degree of freedom yields Figure 3.1.1.

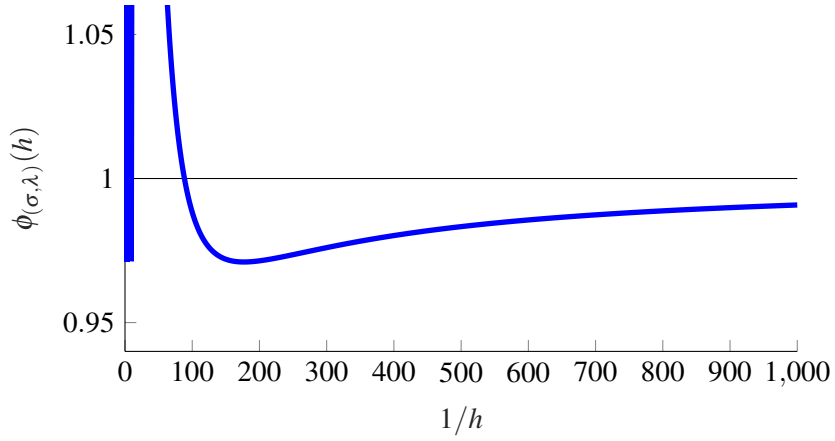


Figure 3.1.1: Plot of $\phi_{(\sigma,\lambda)}(h)$ with $\sigma = -60i$ and $\lambda = -10 - 20i$

We see that we need to choose approximately $h < \frac{1}{100}$ in order to obtain a stable method. To be exact, $\phi_{(\sigma,\lambda)}(h) = 1$ for $h_0 = 0.01131 \approx \frac{1}{88.42}$ and $\phi_{(\sigma,\lambda)}(h) < 1$ for all $h < h_0$ or $\frac{1}{h} > \frac{1}{h_0}$, respectively. Table 3.1.2 shows that the error is high for $h > h_0$ $\left(\frac{1}{h} < \frac{1}{h_0}\right)$ and decreases for $h < h_0$ $\left(\frac{1}{h} > \frac{1}{h_0}\right)$.

$1/h$	$\phi_{(\sigma,\lambda)}(h)$	Error for $T = 1000$
86	1.00326	$2.31130 \cdot 10^{122}$
88	1.00054	$6.14153 \cdot 10^{21}$
89	0.99927	$1.09677 \cdot 10^{1}$
100	0.98830	$2.22493 \cdot 10^{0}$
500	0.98330	$1.42392 \cdot 10^{-1}$

Table 3.1.2: Weighted Error for $h = \frac{1}{86}, \frac{1}{88}, \frac{1}{89}, \frac{1}{100}, \frac{1}{500}$

However, we should notice that the stability function oscillates strongly in the domain $\frac{1}{h} \in (1,10)$. Figure 3.1.3 shows a zoom of Figure 3.1.1, where the small intervals like e. g. $(4.57,5.73)$ and $(8.7,9.4)$ contain the values for which the stability function is less than 1.

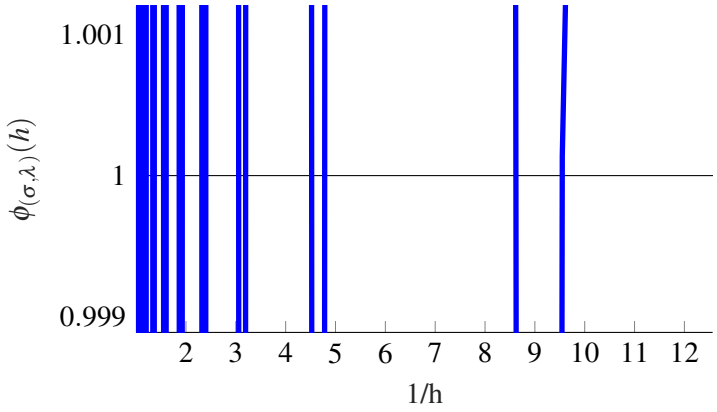


Figure 3.1.3: Zoom of Figure 3.1.1

Hence, there are also values for $h > \frac{1}{100}$, for which the method is stable, e. g. $h = \frac{1}{9}$. The stability circles for $h = \frac{1}{8}, \frac{1}{9}$ and $\frac{1}{10}$ are given in Figure 3.1.4. It is worthwhile noticing that for $h = \frac{1}{9}$ and $h = \frac{1}{10}$, the values w are both close to $-2\pi i$, whereas $\frac{1}{9}\sigma_{im} < -2\pi$ and $\frac{1}{10}\sigma_{im} > -2\pi$ $\left(\frac{1}{9}\sigma_{im} = \frac{1}{9} \cdot (-60) \approx -6.67 < -6.28 \approx -2\pi \text{ and } \frac{1}{10}\sigma_{im} = \frac{1}{10} \cdot (-60) = -6 > -6.28 \approx -2\pi\right)$.
We saw in Lemma 2.3.40 that for negative σ_{im} the circles move from right to left for every interval $[2k\pi, 2(k+1)\pi]$, $k \in \mathbb{Z}$. In this example, the circle for $h = \frac{1}{10}$ is far right, whereas the circle for $h = \frac{1}{9}$ is far left. Indeed, applying the explicit exponential Euler to the above problem, we notice that the error is bounded for the values for which the stability function is less than 1 (see Table 3.1.5) but the approximation is still very bad (see Figure 3.1.6).

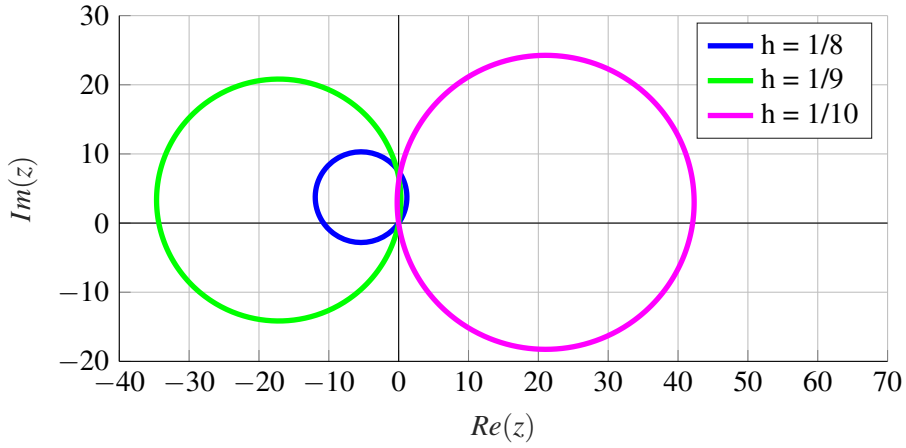
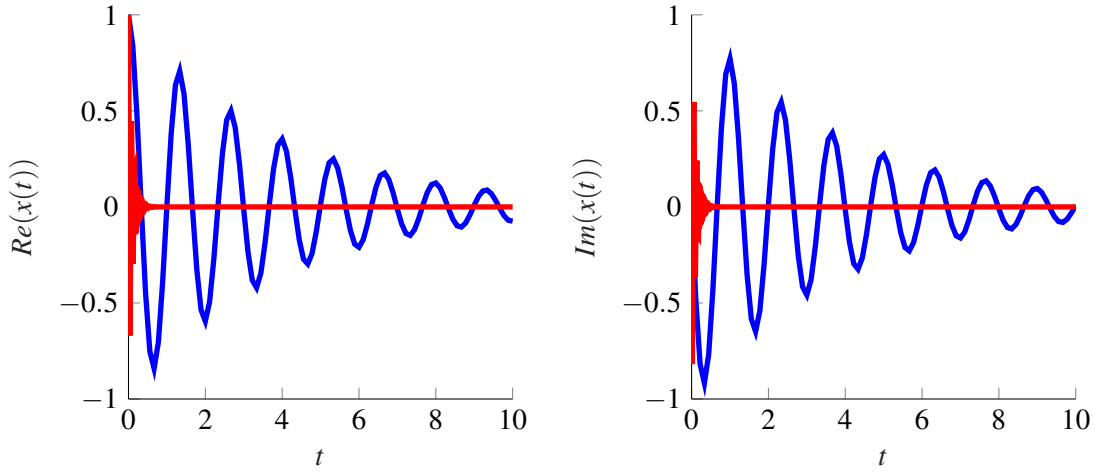


Figure 3.1.4: Stability regions for $h = 1/8, 1/9$ and $1/10$

$1/h$	$\phi_{(s,\lambda)}(h)$	Error for $T = 10$	Error for $T = 100$
8	1.14210	$5.58418 \cdot 10^4$	$2.16701 \cdot 10^{46}$
9	0.97153	$2.82653 \cdot 10^0$	$2.83384 \cdot 10^0$
10	1.06337	$8.80429 \cdot 10^2$	$1.06012 \cdot 10^{27}$

Table 3.1.5: Weighted Error for $h = \frac{1}{8}, \frac{1}{9}, \frac{1}{10}$



(a) Real parts

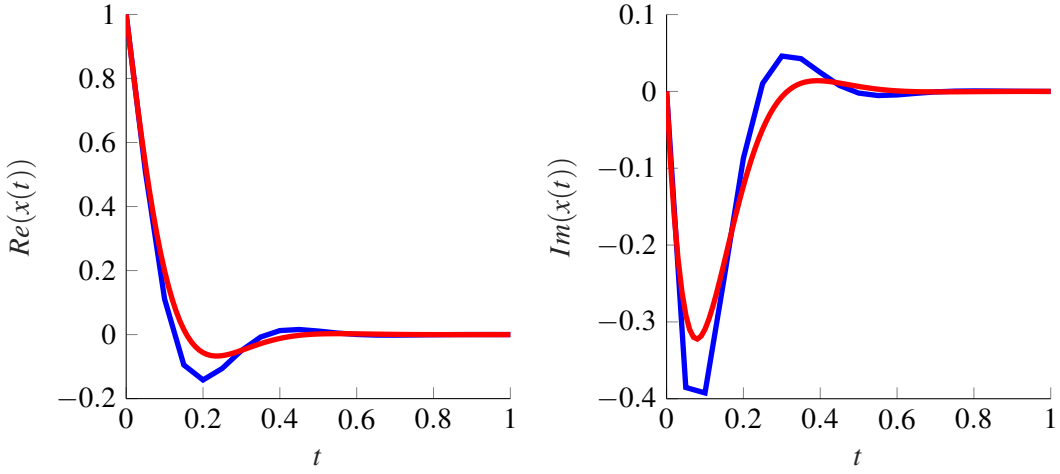
(b) Imaginary parts

Figure 3.1.6: Approximation (blue) vs. analytic solution (red) for $\sigma = -60i, \lambda = -10 - 20i$ and $h = \frac{1}{9}$

3.2. Comparison of exponential Euler method and ordinary Euler method

We compare the convergence properties and the run time of the exponential Euler and the ordinary Euler method. We consider a test equation, where $\sigma_{re} + \ell_{re} = -10$ and $\sigma_{im} + \ell_{im} = -10$. Hence, the equation for the ordinary Euler reads $\dot{x}(t) = (-10 - 10i)x(t)$. For the exponential Euler we can choose a distribution of the value -10 between σ and λ. Since the exponential Euler method integrates the ODE exactly if $\sigma = -10 - 10i$ and $\lambda = 0$, there is nothing to compare in that case. Also, if $\sigma = 0$ and $\lambda = -10 - 10i$, the exponential Euler method degenerates to the ordinary Euler method, in which case there is nothing to compare, either. Hence, we choose a medium with $\sigma = -5 - 5i$ and $\lambda = -5 - 5i$.

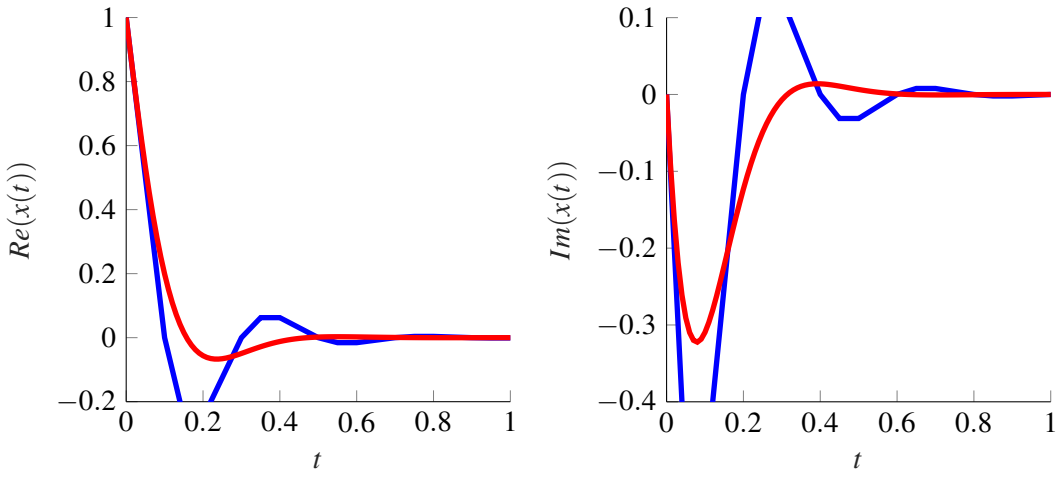
In order to meet the stability requirements of the ordinary Euler method, we need $h < \frac{1}{10}$ in order to obtain a stable method. We choose $h = \frac{1}{20}$. The exponential Euler method yields the results in Figure 3.2.1 and the ordinary Euler method yields the results in Figure 3.2.2



(a) Real parts

(b) Imaginary parts

Figure 3.2.1: Approximation of exponential Euler method (blue) vs. analytic solution (red) for $\sigma = -5 - 5i$, $\lambda = -5 - 5i$ and $h = 0.05$



(a) Real parts

(b) Imaginary parts

Figure 3.2.2: Approximation of ordinary Euler method (blue) vs. analytic solution (red) for $\sigma = -5-5i$, $\lambda = -5-5i$ and $h = 0.05$

We observe that the exponential Euler method (RMS error $1.5 \cdot 10^{-1}$) yields more accurate results than the ordinary Euler (RMS error $3.6 \cdot 10^{-1}$)for the same step width, when at least some linear parts of the right side are stored in σ.

As we can see in Figure 3.2.3, both methods have convergence order 1. However, the error of the exponential Euler approximation is less than the Error of the ordinary Euler method.

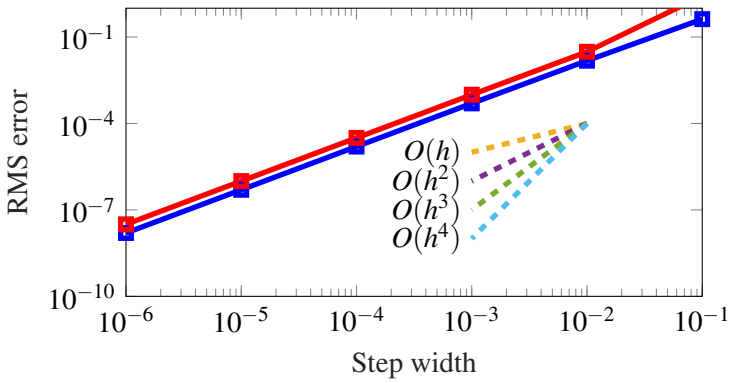


Figure 3.2.3: Comparison of the convergence orders of the exponential Euler method (blue) and the ordinary Euler method (red) for $\sigma = -5-5i$, $\lambda = -5-5i$

Figure 3.2.4 shows the run time comparison. Naturally, we need to input more effort into the exponential Euler method since it yields more accurate results than the ordinary Euler method whereas both have the same convergence order 1. We observe that there exists a difference between exponential Euler and ordinary Euler for a small amount of time steps but it diminishes for a

higher number of time steps. The run time difference stems from the more complex one-time computation of the Runge-Kutta coefficients for the exponential methods, which only have to be computed at the beginning of a call. However, this one-time computation is only dominant for a small computation time and is insignificant compared to the run time of a computation with a high number of time steps.

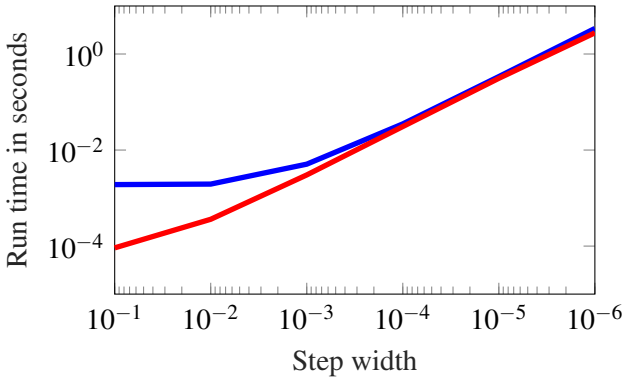


Figure 3.2.4: Comparison of the run times of the exponential Euler (blue) and the ordinary Euler (red) for $\sigma = -5-5i$, $\lambda = -5-5i$

Altogether, the exponential Euler method yields more accurate results than the ordinary Euler method, when a high linear part can be extracted from the function of the right hand side of the ODE. The more expensive computation of the coefficients of the Runge-Kutta method yields a significant run time prolongation for a small amount of time steps, but the overhead becomes insignificant for a higher amount of steps. This is valid for one-dimensional problems. For more-dimensional problems, there will always be an additional overhead caused by the matrix-valued coefficients of the exponential Runge-Kutta methods (cf. Table 3.4.6). However, with a good separation of the underlying system in stiff linear and nonstiff nonlinear, the additional effort will be awarded with a more accurate solution with fewer time steps.

3.3. Comparison of EERK2 and EERK4 for special cases

This section focuses on the comparison of exponential Runge-Kutta methods of second and fourth order for special cases which seem to be interesting when considering the plots in Subsection 2.3.5.

We choose some examples that catch our eye when looking at these plots. Values, for which EERK2 is a stable algorithm and EERK4 is not (see Table 3.3.1) and the other way around are interesting (see Table 3.3.2).

We observe that the value of the stability function has a relevant impact on the approximation error, as expected. The differences in the approximation error can be substantial as Tables 3.3.1

and 3.3.2 show. For problems with a high damping factor going into $h\sigma$, there are no regions that are covered by the stability region of the second order method and uncovered by the fourth order method (cf. Figures 2.3.54d and 2.3.55d).

Figure	$h\sigma$	$h\lambda$	$\phi(h\sigma, h\lambda)$		RMS error	
			EERK2	EERK4	EERK2	EERK4
2.3.53b	$-3i$	$2.2i$	0.3	1.1	$9 \cdot 10^{1}$	$4 \cdot 10^{2}$
2.3.53c	$-6i$	$-8-8i$	0.1	64	$5 \cdot 10^{4}$	$2 \cdot 10^{16}$
2.3.54d	-100	$-25-75i$	0.6	1.3	$5 \cdot 10^{-1}$	$1 \cdot 10^{1}$
2.3.55d	$-100-100i$	$-125i$	0.8	1.4	$3 \cdot 10^{0}$	$3 \cdot 10^{4}$

Table 3.3.1: Example values such that EERK2 is stable and EERK4 is not

Figure	$h\sigma$	$h\lambda$	$\phi(h\sigma, h\lambda)$		RMS error	
			EERK2	EERK4	EERK2	EERK4
2.3.53b	$-3i$	-2	1.1	0.5	$6 \cdot 10^{5}$	$3 \cdot 10^{3}$
2.3.53c	$-6i$	-2.5	1.0	0.9	$6 \cdot 10^{0}$	$9 \cdot 10^{-1}$

Table 3.3.2: Example values such that EERK4 is stable and EERK2 is not

3.4. Comparison of HEERK4 and HERK4

In this section, we compare the classical half-explicit 4^{th} order Runge-Kutta method to our newly defined half-explicit exponential 4^{th} order Runge-Kutta method. As a test example, we use the model defined in Sections 1.2 and C. Additionally to the constants given in Table C.1, we use $\Omega = 7$, our end time is $T = 10$, the initial time is $t_0 = 0$.

In Subsection 1.2, we state that the variable

$$\omega_s := \sqrt{\frac{K_s}{4(a+b)^2 m_b + 4I_{zb}}}$$

indicates the stiffness of the system and hence plays an important role in our simulation. Accordingly, we set

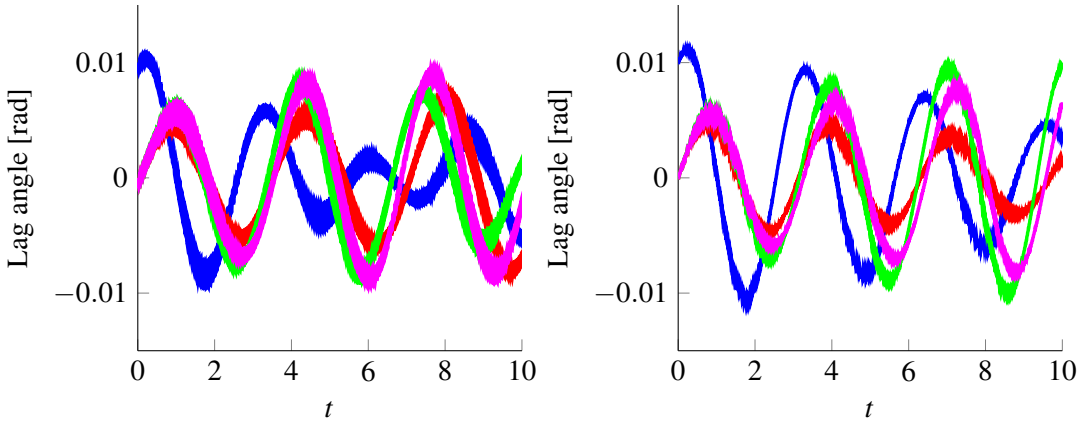
$$S_2 := \begin{pmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{pmatrix}$$

in equation (1.2.19). Since $S_1 = \mathbf{O}_{12\times 12}$ is the 12×12-zero matrix (cf. (1.2.14)), the eigenvalues of

$$S := \begin{pmatrix} S_1 & \mathbf{O}_{12\times 2} \\ \mathbf{O}_{2\times 12} & S_2 \end{pmatrix},$$

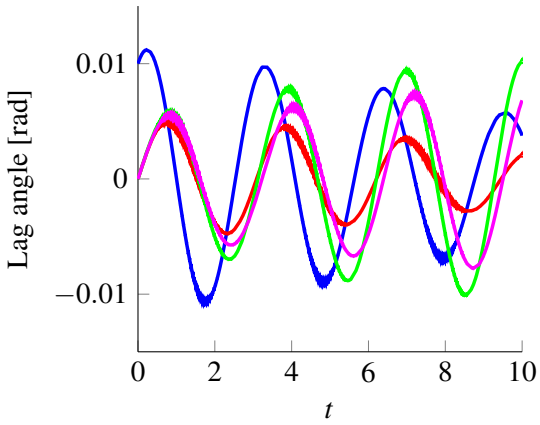
are $\{0, \pm\omega_s i\}$. The higher ω_s is the stiffer the system gets. For our simulation, we will try different values of K_s in order to influence the value of ω_s.

We obtain a reference solution by solving the monolithic system (given in Section C.2) with an explicit 4^{th} order Runge-Kutta method with small time steps ($h = 10^{-5}$). For our analysis of the convergence error, we focus on the variables $\phi_i(t), i = 1, \ldots, 4$ as they exhibit the most interesting behavior. $\phi_1(t)$ indicates the local angle of the first blade. If we start with a small displacement (0.01 rad) for one blade – say the first – leaving the other blades at rest, then the first blade will vibrate when rotating around the mast. Since the mast has also some play, this vibration will set the other blades in a vibrating motion as well. The frequency of the vibrations and the transmission rate depend on the particular degree of stiffness of the mast. Figure 3.4.1 shows the motion of the four blades for $t \in [0, 10]$ and $\omega_s \in \{3.91, 12.35, 39.06\}$ ($K_s \in \{30,000; 300,000; 3,000,000\}$).



(a) $\omega_s = 3.91$

(b) $\omega_s = 12.35$

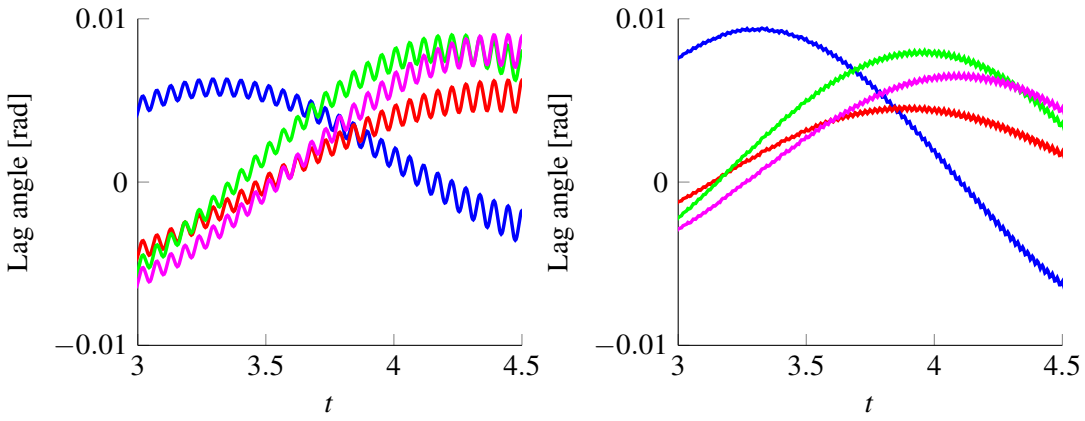


(c) $\omega_s = 39.06$

Figure 3.4.1: Motion of the four blades (blade 1: blue, blade 2: red, blade 3: green, blade 4: pink) of the model for different values of ω_s (author's plot)

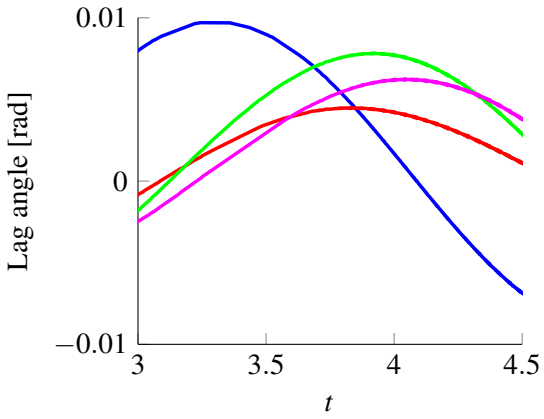
We see that the initial displacement of the first blade is 0.01rad, whereas the other three start

with no lag angle (0 rad). The blades then vibrate back and forth, whereas the overall motion of the first blade decreases and the motion of the other blades builds up due to the play of the mast. Additionally to the low-frequency motions, there is also a high-frequency motion due to the overall vibration of the system. For a less stiff mast ($\omega_s \approx 4$, Figure 3.4.2a), the vibrations have low frequencies and the movement of the blades due to the initial displacement of the first blade is quickly transmitted to the other blades. The stiffer the mast gets the higher the frequencies of the vibration are and the slower the transmission rate becomes (Figures 3.4.2c and 3.4.2b).



(a) $\omega_s = 3.91$

(b) $\omega_s = 12.35$



(c) $\omega_s = 39.06$

Figure 3.4.2: Zoom of the motion of the four blades of the model (blade 1: blue, blade 2: red, blade 3: green, blade 4: pink) (author's plot)

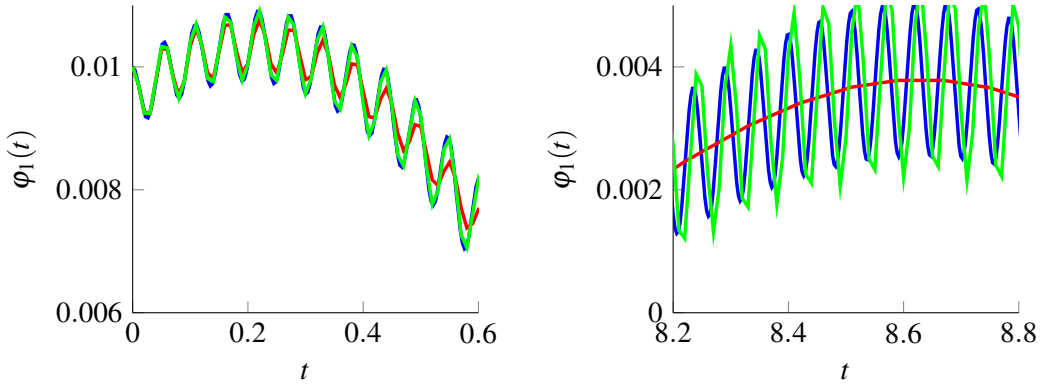
We want to see if the HEERK4 method needs less time steps than the HERK4 method in order to obtain a suitable approximation. Furthermore, we show that it can handle stiffer systems.

We start with a stiffness value of $\omega_s = 3.91$ ($K_s = 30{,}000$). For $h \in \{10^{-2}, 10^{-3}, 10^{-4}\}$, both methods run through. For $h \in \{10^{-3}, 10^{-4}\}$, Table 3.4.3 shows that the approximation of HEERK4 is more accurate than the approximation of HERK4. However, for $h = 10^{-2}$, the significance of

the approximation error is low since the approximation of HERK4 starts off with accounting for the high frequency vibrations in the beginning and finally ends up smoothing them out, whereas the approximation of HEERK4 accounts for the vibrations but obtains an offset with advancing time (see Figure 3.4.4). Both approximations yield a similar RMS error while approximating the solution in very different ways.

h	HERK4	HEERK4
10^{-2}	$1.5 \cdot 10^{-1}$	$1.2 \cdot 10^{-1}$
10^{-3}	$1.4 \cdot 10^{-4}$	$7.3 \cdot 10^{-6}$
10^{-4}	$1.4 \cdot 10^{-8}$	$7.2 \cdot 10^{-10}$

Table 3.4.3: RMS error for $\omega_s = 3.91$



(a) $t \in [0, 0.6]$ (b) $t \in [8.2, 8.8]$

Figure 3.4.4: Plot of reference solution (blue), approximation HERK4 (red) and approximation HEERK4 (green) for $\varphi_1(t)$ (author's plot)

The simulation also shows the convergence orders of HERK4 and HEERK4. As expected, both are fourth order methods (cf. Table 3.4.3 and Figure 3.4.5).

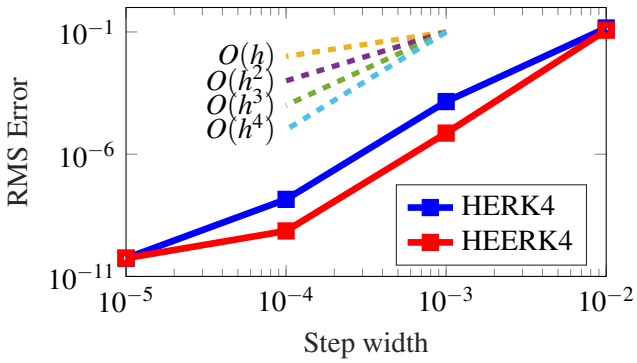


Figure 3.4.5: Visualization of convergence rates of HERK4 and HEERK4

Let us now consider $\omega_s = 12.35$ ($K_s = 300,000$). For this stiffer system, Figure 3.4.2b shows that the frequency of the high frequency vibrations becomes higher. For a step size of $h = 0.01$, the HERK4 method does not converge any more, whereas the HEERK4 method yields an approximation in under one second.

Table 3.4.6 contains the run times of HERK4 and HEERK4 for the different step widths and values for ω_s. The run times include the calculation time needed for the main loop of the algorithms. The calculation of the coefficients for HEERK4 is constant for any stiffness and any step width and needs around 0.3 seconds. We observe that HEERK4 needs about 30% more time than the respective HERK4 method with the same step width. This overhead is due to the matrix-valued coefficients in the exponential Runge-Kutta methods that cause a higher run time in every iteration than the scalar coefficients of ordinary Runge-Kutta methods. However, the lower limit of step widths, for which the methods deliver approximations is about 10 times higher for the HEERK4 method (i. e. for $\omega_s = 12.35$, HERK4 does not converge anymore for $h = 10^{-2}$, and for $\omega_s = 39.06$, both methods do not converge for $h = 10^{-2}$).

h	$\omega_s = 3.91$		$\omega_s = 12.35$		$\omega_s = 39.06$	
	HERK4	HEERK4	HERK4	HEERK4	HERK4	HEERK4
10^{-1}	n.a.	n.a.	n.a.	n.a.	n.a.	n.a.
10^{-2}	0.7	1.0	n.a.	0.9	n.a.	n.a.
10^{-3}	7.4	9.8	7.5	9.7	7.4	10.8
10^{-4}	70.0	93.7	67.1	93.0	74.0	104.0

Table 3.4.6: Run time comparison in seconds

For different values of ω_s, the run times can differ because the inner Newton iteration may need different numbers of steps in order to reach the desired accuracy.

It is interesting to observe how both methods deliver smoothed approximations (as shown in Figure 3.4.4) for the smallest amount of time steps, for which they deliver an approximation at all. This applies to the HERK4 with a step width of $h = 10^{-2}$ and $\omega_s = 3.91$, HEERK4 with a step width of $h = 10^{-2}$ and $\omega_s = 12.35$ and HERK4 with a step width of $h = 10^{-3}$ and $\omega_s = 39.06$.

We observe that we cannot go much higher with our ω_s or K_s. With every zero that we add to K_s one more method delivers divergent results. However, we assumed that the HEERK4 method would perform even better than seen here. If we consider the eigenvalues of the linear part of the monolithic system for $t = 0$ and respective initial conditions, we find that the eigenvalues with the highest modulus are much higher than the eigenvalues of S (see Table 3.4.7).

max. eigenvalues of S	max. eigenvalues of mon. system
$\pm 3.91i$	$\pm 114i$
$\pm 12.35i$	$\pm 326i$
$\pm 39.06i$	$\pm 1,021i$

Table 3.4.7: Highest modulus eigenvalues of S vs. highest modulus eigenvalues of the monolithic system for $t = 0$

Apparently, this means that we did not take too much of the system's stiffness into our linear part represented by S. Most of the stiffness stays in the nonlinear part $\tilde{f}$. If we consider the monolithic model and its governing matrix

$$\begin{pmatrix} \mathbf{O}_{7\times 7} & \mathbf{I}_{7\times 7} \\ -M_{mon}^{-1}K_{mon} & -M_{mon}^{-1}G_{mon} \end{pmatrix}$$

from equation (1.2.11) at $t = t_0, \mathbf{x} = \mathbf{x}_0$, we observe that $-M_{mon}^{-1}K_{mon}$ is a 7×7-matrix, where all entries of the seventh column linearly depend on ω_s^2. These are also the highest entries in this matrix. If we divide the whole column by ω_s^2, we roughly obtain the vector $(0,0,705,705,705,705,-680)^T$.

If we now set

$$S_{mon} := \omega_s^2 \cdot \begin{pmatrix} & & 0 & \\ & & 0 & \\ & & 0 & \\ & & 0 & \\ & & 0 & \\ & & 0 & \\ & & 0 & \\ \mathbf{O}_{14\times 6} & & 0 & \mathbf{O}_{14\times 7} \\ & & 0 & \\ & & 705 & \\ & & 705 & \\ & & 705 & \\ & & 705 & \\ & & -680 & \end{pmatrix}, \tag{3.4.8}$$

$$\tilde{f}_{mon}(t,\mathbf{x}(t)) := f_{mon}(t,\mathbf{x}(t)) - S_{mon}\cdot\mathbf{x}(t) \tag{3.4.9}$$

and apply the EERK4 to $\dot{x}(t) = S_{mon}\cdot x(t) + \tilde{f}_{mon}(t,x(t))$, then $\tilde{f}_{mon}$ is less stiff than f_{mon} and we can go much higher with the value of ω_s without loosing the stability of the exponential Runge-Kutta method (see Table 3.4.10). Only for a value of 390.61 for ω_s ($K_s = 300,000,000$), the EERK4

method stops being convergent for a step width of $h = 10^{-1}$. This is quite impressing considering that the system is very stiff since the highest modulus eigenvalues of the monolithic system at time $t = 0$ are $\pm 10,199i$ for $K_s = 300,000,000$.

h	$\omega_s = 3.91$	$\omega_s = 12.35$	$\omega_s = 39.06$	$\omega_s = 123.52$	$\omega_s = 390.61$
10^{-1}	✓	✓	✓	✓	✗
10^{-2}	✓	✓	✓	✓	✓
10^{-3}	✓	✓	✓	✓	✓
10^{-4}	✓	✓	✓	✓	✓

Table 3.4.10: Converging step widths for EERK4

The seventh column of the system matrix refers to the seventh entry of x which is $\bar{\varphi}_{RH}(t)$. This explains the above phenomenon. We chose $\bar{\varphi}_{RH}(t)$ – a variable that itself exhibits stiff behavior – as an in-/output variable. Hence, the respective parts in the DAE-system are stored in the function g_{DAE} and not in f_{DAE} and hence cannot be captured in S_{DAE}. So, when defining the submodels of a model, we probably should avoid using oscillatory variables as in-/output variables in order to be able to put most of the stiffness of the system into the constant linear part S.

Apparently, we cannot explain the fact that a nilpotent matrix S can have such any effect on the performance of a HEERK method by a linear model. Table 3.4.10 shows that although all eigenvalues of our matrix S_{mon} are 0, it still has a huge impact on the performance of the method. In the linear case, $\sigma = 0$ would not yield any optimizations to the ordinary methods.

From Table 3.4.10, we also observe that a step width of $h = 10^{-1}$ is still enough for $\omega_s = 123.52$ ($K_s = 30,000,000$). Using the HERK4 method for such a setting, we require a step width of $h = 10^{-4}$ in order to obtain a converging approximation. That means the required time steps are a thousand times smaller for a comparable quality approximation.

Another possibility of obtaining a better S in our DAE model would be a linearization. However that way, we loose the block diagonal structure of S which makes the computation of $exp(hS)$ in the calculation of the coefficients of exponential integrators much more complex.

We conclude that the HEERK4 method is better suited for stiff problems as soon as it is possible to put some of the system's stiffness in the linear constant part. Then, HEERK4 delivers better results than HERK4 for equal step widths and HEERK4 even yields converging results where HERK4 does not anymore. However, it is important to not choose variables that exhibit stiff behavior as in-/output variables when defining the submodels of the system.

4. Conclusion and recommendations for further work

With the definition of half-explicit exponential Runge-Kutta (HEERK) methods, we did a first step towards a more efficient helicopter simulation. We started this thesis with an overview of already existing numerical approaches and software. After a discussion of the drawbacks of the current state of the art, we presented our approach: a general coupled state-space model or mathematically speaking: an index-1 differential algebraic equations (DAEs) system.

Since we were looking for a method that would simulate the helicopter's behavior over longer periods in an acceptable simulation time, we focused on explicit methods. A family of existing explicit methods for index-1 DAEs are half-explicit Runge-Kutta (HERK) methods that are derived from explicit Runge-Kutta (ERK) methods for ODEs paired with a Newton method. These methods are well suited for solving index-1 DAEs. However, they cannot handle the stiff helicopter system. For such stiff ODE systems, there exist explicit exponential Runge-Kutta (EERK) methods. For these methods, a formulation of the ODE is necessary where the stiffness of the ODE is separately given in form of a linear part. In order to make exponential integrators applicable to index-1 systems, we needed to define half-explicit exponential Runge-Kutta (HEERK) methods analogously to HERK methods.

With this aim, we investigated consistency and stability properties of ERK and their translation to HERK methods. Stability and consistency results for ERK methods are well known. While the stability results translate to HERK methods, the consistency results need to be shown seperately. Although, there are proofs of consistency in the literature (e. g. [ASW93, HW96]), we chose to conduct a straight forward and more comprehensible proof for methods of orders one and two which we could then translate more easily to HEERK methods.

In an analogous way, we analyzed EERK methods focusing on consistency and stability. Here, consistency results were available in the literature [HO10]. However, we could not find a linear stability analysis of EERK methods. For the first order explicit exponential method (exponential Euler method) we considered the stability function and its behavior. For methods of higher orders, we plotted the stability regions in analogy to the well-known stability plots of ordinary Runge-Kutta methods. Since the stability functions additionally depend on the linear stiff part of the state function, we obtained interesting plots, where the stability regions deformed and moved with respect to different inputs.

Combining our findings for HERK methods and EERK methods, we developped HEERK methods. To the best of our knowledge, they have not been used in the literature before. For the methods of first and second order we used our comprehensible proofs to show consistency.

Our numerical testing showed the strengths of HEERK methods compared to HERK methods. We used a mechanical model, the stiffness of which we could adapt according to our needs. We

saw that HEERK methods need far less time steps than HERK methods in order to achieve a comparable approximation quality on stiff systems. However, we need to be careful when defining the submodels. We chose the rotational angle of the rotor head as an in-/output variable in our example. However, this variable exhibits stiff behavior itself. As a result, our HEERK method already becomes instable with comparably small step sizes and hence is not that much better than the ordinary integrators. Though if we place the oscillatory variables solely in the state function and if we are able to seperate them into the linear part, then we can dramatically enhance the approximation quality of HEERK methods which makes them a very efficient tool for solving stiff index-1 DAE systems.

This thesis already contains a lot of promising results. However, there are still many aspects that could be analyzed for an even better framework.

Concerning stability aspects of exponential methods, we covered the behavior of the stability regions of methods up to order 4 for various settings of linear problems. We also thoroughly investigated the analytical stability function of the exponential Euler method (first order exponential Runge-Kutta method). An in-depth analysis of the analytical functions describing the stability regions of higher order methods is just as interesting but has not yet been conducted.

For the consistency of HERK and HEERK methods, we have shown that the Newton methods need to be executed to the accuracy that we want the underlying Runge-Kutta method to have. In [ARW93], Arnold et al. give a stronger statement for HERK methods. They indicate specific numbers of Newton steps that are necessary for the underlying Runge-Kutta method to be convergent of its ordinary order. It would be interesting to apply their results to half-explicit exponential Runge-Kutta methods. This would further enhance their run time.

Future research will show how our HEERK methods will perform on more complex helicopter models. Our mechanical model showed that the submodels need to be chosen cautiously. The in-/output variables should not exhibit any oscillatory behavior and the linear part should contain most of the stiffness of the system. In our experiment we learned that this does not have to mean that the matrix S needs to have high modulus eigenvalues. It will be enlightening to analyze the criteria which S has to fulfil for HEERK methods to work well.

Altogether, half-explicit exponential methods bring us considerably closer to a more efficient and effective simulation of large stiff systems in a general coupled state-space representation. Their elaborate application in comprehensive helicopter simulation has the ability to prevent accidents and support the simulation driven design of helicopters in the future.

List of abbreviations

cf.	confer	3
DAE	Differential-Algebraic Equation	3
DLR	German Aerospace Center	1
EERK	Explicit Exponential Runge-Kutta	29
e. g.	Exempli gratia (for example)	5
ERK	Explicit Runge-Kutta	14
FT	Institute of Flight Systems	1
HEERK	Half-Explicit Exponential Runge-Kutta	62
HERK	Half-Explicit Runge-Kutta	20
HPC	High Performance Computing	1
i. e.	Id est (that is to say)	7
IMEX	Implicit-Explicit	28
ODE	Ordinary Differential Equation	5
p.	page	13
PDE	Partial Differential Equation	2
pp.	pages	13
RK	Runge-Kutta	14
SPP	Singularly Perturbed Problem	13
VAST	Versatile Aeromechanics Simulation Tool	1

Notation directory

Section 1

Notation	Description	Def. on
$t \in \mathbb{R}_+$	Real time variable	page 5
$n_i \in \mathbb{N}$	Degrees of freedom of the state vector of subsystem i	page 5
$m_i \in \mathbb{N}$	Degrees of freedom of the output vector of subsystem i	page 5
$q_i \in \mathbb{N}$	Degrees of freedom of the input vector of subsystem i	page 5
$\mathbf{x}_i(t) \in \mathbb{R}^{n_i}$	Local vector of state variables of subsystem i	page 5
$\dot{\mathbf{x}}_i(t) \in \mathbb{R}^{n_i}$	Also $\mathbf{x}_i'(t)$, time derivative of $\mathbf{x}_i(t)$	page 5
$\mathbf{y}_i(t) \in \mathbb{R}^{m_i}$	Local vector of output variables of subsystem i	page 5
$\mathbf{u}_i(t) \in \mathbb{R}^{q_i}$	Local vector of input variables of subsystem i	page 5
$f_i : \mathbb{R}_+ \times \mathbb{R}^{n_i} \times \mathbb{R}^{q_i} \to \mathbb{R}^{n_i}$	State function of subsystem i	page 5
$g_i : \mathbb{R}_+ \times \mathbb{R}^{n_i} \times \mathbb{R}^{q_i} \to \mathbb{R}^{m_i}$	Output function of subsystem i	page 5
$n = \sum_i n_i$	Degrees of freedom of the global state vector	page 5
$m = \sum_i m_i$	Degrees of freedom of the global output vector	page 5
$\mathbf{x}(t) \in \mathbb{R}^n$	Vector of global state variables	page 5
$\mathbf{y}(t) \in \mathbb{R}^m$	Vector of global output variables	page 5
$f : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$	Global state function	page 5
$g : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m$	Global output function	page 5
$\bar{g} : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^m$	Modified output function	page 5
$\bar{g}^{-1} : 0 \times \mathbb{R}_+ \times \mathbb{R}^n$	Implicit function to $\bar{g}(t, x(t), y(t)) = 0$	page 6
$\frac{\partial g}{\partial \mathbf{y}}(t, \mathbf{x}(t), \mathbf{y}(t))$	Also $\partial_{\mathbf{y}} g(t, \mathbf{x}(t), \mathbf{y}(t))$, partial derivative of g with respect to y	page 6
$t_0 \in \mathbb{R}_+$	Initial time	page 6
$\mathbf{x}_0 \in \mathbb{R}^n$	Initial vector for the state variables	page 6
$\mathbf{y}_0 \in \mathbb{R}^m$	Initial vector for the output variables	page 6
$S_i \in \mathbb{R}^{n_i \times n_i}$	Constant matrix containing the stiff part of f_i	page 7
$\tilde{f}_i : \mathbb{R}_+ \times \mathbb{R}^{n_i} \times \mathbb{R}^{q_i} \to \mathbb{R}^{n_i}$	In general nonlinear function containing the nonstiff part of f_i	page 7
$S \in \mathbb{R}^{n \times n}$	Constant block diagonal matrix containing the stiff part of f	page 7
$\tilde{f} : \mathbb{R}_+ \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$	In general nonlinear function containing the nonstiff part of f	page 7
$M \in \mathbb{R}^{6 \times 6}$	Mass matrix of the rotor model in [SMBA11]	page 8
$\mathbf{u} : \mathbb{R}_+ \to \mathbb{R}^6$	Vector of variables of the rotor model in [SMBA11]	page 8
$G \in \mathbb{R}^{6 \times 6}$	Damping matrix of the rotor model in [SMBA11]	page 8

$K \in \mathbb{R}^{6\times 6}$	Stiffness matrix of the rotor model in [SMBA11]	page 8
$F \in \mathbb{R}^6$	External force vector of the rotor model in [SMBA11]	page 8
$x_{Fus} : \mathbb{R}_+ \to \mathbb{R}$	Longitudinal displacement of the fuselage in meters	page 8
$y_{Fus} : \mathbb{R}_+ \to \mathbb{R}$	Transversal displacement of the fuselage in meters	page 8
$\varphi_i : \mathbb{R}_+ \to \mathbb{R}$	Lead-lag angle of i^{th} blade in rad	page 8
$\mathbf{v} : \mathbb{R}_+ \to \mathbb{R}^6$	Vector of derivatives of variables of the rotor model in [SMBA11]	page 8
$M^{-1} \in \mathbb{R}^{6\times 6}$	Inverse of the matrix M	page 8
$K_s \in \mathbb{R}$	Stiffness coefficient of the mast	page 8
$\varphi_{RH} : \mathbb{R}_+ \to \mathbb{R}$	Rotational angle of the rotor head in rad	page 8
$\omega_i : \mathbb{R}_+ \to \mathbb{R}$	Time derivative of $\varphi_i(t)$	page 8
$\omega_{RH} : \mathbb{R}_+ \to \mathbb{R}$	Rotor head's resonance frequency in Hz (time derivative of $\varphi_{RH}(t)$)	page 8
$\alpha_{RH} : \mathbb{R}_+ \to \mathbb{R}$	Time derivative of $\omega_{RH}(t)$	page 8
$a \in \mathbb{R}$	Rotor eccentricity in meters	page 8
$b \in \mathbb{R}$	Blade length in meters	page 8
$m_b \in \mathbb{R}$	Mass of a blade in kilogramm	page 8
$I_{zb} \in \mathbb{R}$	Lag rotational inertia of a blade around its center of gravity in kg m^2	page 8
$\omega_s \in \mathbb{R}$	Variable in the advanced monolithic rotor model	page 8
$r_{sb} \in \mathbb{R}$	Variable in the advanced monolithic rotor model	page 9
$M_{mon} \in \mathbb{R}^{7\times 7}$	Mass matrix of the advanced monolithic rotor model	page 9
$G_{mon} \in \mathbb{R}^{7\times 7}$	Damping matrix of the advanced monolithic rotor model	page 9
$K_{mon} \in \mathbb{R}^{7\times 7}$	Stiffness matrix of the advanced monolithic rotor model	page 9
$F_{mon} \in \mathbb{R}^7$	External force vector of the advanced monolithic rotor model	page 9
$\mathbf{u}_{mon} : \mathbb{R}_+ \to \mathbb{R}^7$	Vector of variables of the advanced monolithic rotor model	page 9
$\mathbf{v}_{mon}(t)$	Vector of derivatives of variables of the advanced monolithic rotor model	page 9
$\mathbf{x}_{mon}(t)$	State vector of variables of the advanced monolithic rotor model	page 9
$\mathbf{O}_{n\times n}$	$n \times n$-zero matrix	page 9
$\mathbf{I}_{n\times n}$	$n \times n$-identity matrix	page 9
$\mathbf{0}_n$	n-dimensional zero vector	page 9
$M_{DAE} \in \mathbb{R}^{6\times 6}$	Mass matrix of the advanced rotor model in DAE formulation	page 9

$G_{DAE} \in \mathbb{R}^{6\times6}$	Damping matrix of the advanced rotor model in DAE formulation	page 9
$K_{DAE} \in \mathbb{R}^{6\times6}$	Stiffness matrix of the advanced rotor model in DAE formulation	page 9
$F_{DAE} \in \mathbb{R}^{6}$	External force vector of the advanced rotor model in DAE formulation	page 9

Section 2

Notation	Description	Def. on
$\varepsilon \in \mathbb{R}$	Small value greater than zero	page 13
s	Number of stages of a method	page 14
$A = (a_{ij})$	Coefficient matrix in Butcher-Tableau for Runge-Kutta methods	page 14
$\mathbf{b} = (b_i)$	Coefficient vector in Butcher-Tableau for Runge-Kutta methods	page 14
$\mathbf{c} = (c_j)$	Coefficient vector in Butcher-Tableau for Runge-Kutta methods	page 14
T	End time of a numerical simulation	page 15
N	Total number of steps between 0 and T	page 15
h	Constant step width for numerical algorithm	page 15
k_i or $\mathbf{k}_i$	Internal stages of Runge-Kutta methods	page 15
t_n	Time after n time steps ($t_n := t_0 + n \cdot h$)	page 15
$\mathbf{x}_n$	Approximation of $\mathbf{x}(t_n)$ supplied by the algorithm	page 15
$\Phi(t, \mathbf{x}(t))$	Increment function of Runge-Kutta methods	page 15
$\tau(h)$	Local truncation error	page 15
$O(\cdot)$	Landau symbol / Big O notation	page 15
$\mathbf{x}_h(t)$	Numerical approximation of $\mathbf{x}(t)$	page 16
$\varepsilon(t,h)$	Global truncation error	page 16
$\lambda \in \mathbb{C}$	Parameter in the Dahlquist test equation	page 16
z	Variable for $h\lambda$	page 16
$\phi(z)$	Stability function of explicit Runge-Kutta methods	page 16
$\mathbf{k} \in \mathbb{R}^s$	Vector containing all internal stages k_i	page 18
$\mathbf{e} \in \mathbb{R}^s$	One vector in $\mathbb{R}^s$	page 18
$L \in \mathbb{R}^{n\times n}$	Constant matrix in n-dimensional Dahlquist test equation	page 18
$U \in \mathbb{R}^{n\times n}$	Matrix containing the eigenvectors of L	page 18
$\lambda_i \in \mathbb{C}$	i^{th} eigenvalue of L	page 18
$\Lambda \in \mathbb{R}^{n\times n}$	Matrix containing the eigenvalues λ_i of L on its diagonal	page 18

$\hat{\mathbf{x}}_n := U^{-1}x_n$	Substitution in the proof of Lemma 2.1.13	page 18
$\hat{\mathbf{k}}_i := U^{-1}k_i,\ i = 1,\dots s$	Substitution in the proof of Lemma 2.1.13	page 18
$\hat{\mathbf{k}}_i^{(\ell)}$	ℓ^{th} entry of the vector $\hat{\mathbf{k}}_i$	page 19
$\tilde{g}^{-1}$	Approximation of $\bar{g}^{-1}$ when applying the Newton method	page 23
$\triangle\mathbf{y}$	Error in $\mathbf{y}$ when evaluating $\tilde{g}^{-1}(0,t,\mathbf{x})$	page 23
$\tilde{\mathbf{y}} = \mathbf{y} + \triangle\mathbf{y}$	Approximation of $\mathbf{y}$ when applying the Newton method	page 23
$\mathbf{C}(t) \in \mathbb{R}^n$	Integration constant in derivation of EERK methods	page 29
$\tau \in \mathbb{R}$	Auxiliary integration variable	page 29
$\mathbf{b}_i(hS)$	$n \times n$-coefficient matrix for EERK methods	page 30
$\theta \in \mathbb{R}$	Auxiliary integration variable	page 30
$\ell_i(\theta)$	Lagrange interpolation polynomials	page 30
$\mathbf{a}_{ij}(hS)$	$n \times n$-coefficient matrix for EERK methods	page 30
X_{ni}	Approximation of $\mathbf{x}(t_n + c_i h)$ in EERK methods	page 30
$\widetilde{F}_{nj}$	Internal stages in EERK methods	page 31
$\chi(hS)$	Coefficients for the linear part in EERK methods	page 31
$\chi_i(hS)$	Coefficients for the linear part in EERK methods	page 31
$\varphi_k(hS)$	Auxiliary function for order conditions for EERK	page 33
$\varphi_{k,\ell}(hS)$	Auxiliary function for order conditions for EERK	page 33
$\psi_k(hS)$	Auxiliary function for order conditions for EERK	page 33
$\psi_{k,\ell}(hS)$	Auxiliary function for order conditions for EERK	page 33
J_1, J_2	Bounded operators for order conditions for EERK	page 34
$\sigma \in \mathbb{C}$	One-dimensional stiff part of state equation	page 34
$\widetilde{\mathbf{F}} \in \mathbb{R}^s$	Vector of internal stages of EERK	page 35
$A(h\sigma) \in \mathbb{R}^{s\times s}$	Coefficient matrix for one-dimensional EERK method	page 35
$\mathbf{b}(h\sigma) \in \mathbb{R}^s$	Coefficient vector for one-dimensional EERK method	page 35
w	Variable for $h\sigma$ in analogy to z	page 36
$\phi(w,z)$	Stability function of explicit exponential Runge-Kutta methods	page 36
$\ell_{re} \in \mathbb{R}$	Real part of λ	page 39
$\ell_{im} \in \mathbb{R}$	Imaginary part of λ	page 39
$\sigma_{re} \in \mathbb{R}$	Real part of σ	page 39
$\sigma_{im} \in \mathbb{R}$	Imaginary part of σ	page 39
$z_{re} = h\ell_{re} \in \mathbb{R}$	Real part of z	page 39
$z_{im} = h\ell_{im} \in \mathbb{R}$	Imaginary part of z	page 39
$w_{re} = h\sigma_{re} \in \mathbb{R}$	Real part of w	page 39
$w_{im} = h\sigma_{im} \in \mathbb{R}$	Imaginary part of w	page 39
a, b	Auxiliary variables in the proof of Theorem 2.3.20	page 40
$\alpha \in \mathbb{C}, \beta, \gamma \in \mathbb{R}$	Auxiliary variables in Lemma 2.3.24	page 43

$o \in \mathbb{C}$	Circle center of a circle in $\mathbb{C}$	page 43
$r \in \mathbb{R}$	Radius of a circle	page 43
$\tilde{\alpha} \in \mathbb{C},\ \tilde{\beta}, \tilde{\gamma} \in \mathbb{R}$	Auxiliary variables in the proof of Theorem 2.3.29	page 44
$\tilde{\eta}(w_{re}, w_{im})$	Auxiliary function in the proof of Theorem 2.3.29	page 45
η	Auxiliary variable in the proof of Theorem 2.3.29	page 45
$r(w_{re}, w_{im})$	Auxiliary function in the proof of Lemma 2.3.30	page 47
$h(w_{im})$	Auxiliary function in the proof of Lemma 2.3.40	page 50

Section 3

Notation	Description	Def. on
$\phi_{(\sigma,\lambda)}(h)$	Stability function for fixed σ and λ	page 68

A. Basic definitions and theorems

A.1. Analysis

Theorem A.1.1 (Implicit function theorem ([Fo13] (p.93, Theorem 2)))**.**
Let $U_1 \subset \mathbb{R}^k$ and $U_2 \subset \mathbb{R}^m$ be open subsets and let

$$F : U_1 \times U_2 \to \mathbb{R}^m, \quad (x,y) \mapsto F(x,y)$$

be a continuous differentiable map. Let $(a,b) \in U_1 \times U_2$ with $F(a,b) = 0$. Let the $m \times m$-matrix

$$\frac{\partial F}{\partial y} := \begin{pmatrix} \frac{\partial F_1}{\partial y_1} & \cdots & \frac{\partial F_1}{\partial y_m} \\ \vdots & & \vdots \\ \frac{\partial F_m}{\partial y_1} & \cdots & \frac{\partial F_m}{\partial y_m} \end{pmatrix}$$

be regular in (a,b). Then there exist an open neighborhood $V_1 \subset U_1$ of a, an open neighborhood $V_2 \subset U_2$ of b and a continuous differentiable map $g : V_1 \to V_2 \subset \mathbb{R}^m$ with $g(a) = b$, such that

$$F(x, g(x)) = 0 \text{ for all } x \in V_1.$$

If $(x,y) \in V_1 \times V_2$ with $F(x,y) = 0$, then $y = g(x)$.

Theorem A.1.2 (L'Hôspital's rule ([Fo16] (p.190, Theorem 10)))**.**
Let $f, g : I \to \mathbb{R}$ be two differentiable functions on the interval $I = (a,b)$ with $-\infty \leq a < b \leq \infty$. Let $g'(x) \neq 0$ for all $x \in I$ and let the limit

$$\lim_{x \uparrow b} \frac{f'(x)}{g'(x)} =: c \in \mathbb{R}$$

exist. If $\lim\limits_{x \uparrow b} g(x) = \lim\limits_{x \uparrow b} f(x) = 0$, then $g(x) \neq 0$ for all $x \in I$ and

$$\lim_{x \uparrow b} \frac{f(x)}{g(x)} = c.$$

The same holds for the limit $x \downarrow a$.

Definition A.1.3 (Integrability ([Fo17] (p. 48, Theorem 8)))**.**
Let $(\Omega, \mathcal{A}, \mu)$ be a measure space and let $f : \Omega \to \mathbb{R} \cup \{\pm\infty\}$ be an $\mathcal{A}$-measurable function. Then,

$$f \text{ is integrable } \Leftrightarrow \int_\Omega |f| d\mu < \infty.$$

Definition A.1.4 (Uniform convergence ([Fo13] (p.24)))**.**
Let X be a set, let Y be a metric space and let

$$f_k : X \to Y, k \in \mathbb{N}, \text{ and } f : X \to Y$$

be maps. The series $(f_k)_{k\in\mathbb{N}}$ converges uniformly towards f, if for every $\varepsilon > 0$, there exists an $N \in \mathbb{N}$ such that

$$\|f_k(x), f(x)\| < \varepsilon \text{ for all } x \in X \text{ and all } k \geq N.$$

Definition A.1.5 (Absolute convergence ([Fo16] (p.74)))**.**
A series $\sum_{k=0}^{\infty} a_k$ is called absolutely convergent, if the series of the moduli $\sum_{k=0}^{\infty} |a_k|$ converges.

Lemma A.1.6. ([Fo16] (p. 83, Theorem 1))
The power series $exp(x) := \sum_{k=0}^{\infty} \frac{x^k}{k!}$ converges absolutely towards $exp(x)$ for every $x \in \mathbb{R}$.

Theorem A.1.7 (Interchange of summation and integration ([Re08] (p. 85, Theorem 8)))**.**
Let $f_k : [a,b] \to \mathbb{K}$ be functions such that the limit

$$F(x) := \sum_{k=0}^{\infty} f_k(x) \text{ for all } x \in [a,b]$$

exists. If all f_k are integrable and if the sequence $\left\{ \sum_{k=0}^{n} f_k(x) \right\}$ converges uniformly towards $F(x)$, then

$$\int_a^b F(x)dx = \int_a^b \sum_{k=0}^{\infty} f_k(x)dx = \sum_{k=0}^{\infty} \int_a^b f_k(x)dx \text{ for all } x \in [a,b].$$

Remark A.1.8.
Let $M \in \mathbb{R}^{n\times n}, n \in \mathbb{N}$ and $f_k := \frac{(M\theta)^k}{k!}, k \in \mathbb{N}$. Since the exponential power series $\sum_{k=0}^{\infty} f_k$ is absolutely convergent towards the exponential function (Lemma A.1.6), it is particularly locally uniformly convergent in $\Omega := [0,1]$. Furthermore the partial sums $s_n := \sum_{k=0}^{n} f_k$ are locally integrable in Ω. Since all requirements of Theorem A.1.7 are met, we obtain

$$\int_0^1 e^{M\theta} d\theta = \int_0^1 \sum_{k=0}^{\infty} \frac{(M\theta)^k}{k!} d\theta = \sum_{k=0}^{\infty} \int_0^1 \frac{(M\theta)^k}{k!} d\theta.$$

A.2. Linear Algebra

Definition A.2.1 (Jordan matrix ([Fi03] (p. 264, 4.6.5)))**.**
A matrix

$$J_k := \begin{pmatrix} 0 & 1 & & 0 \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ 0 & & & 0 \end{pmatrix} \in \mathbb{R}^{k\times k}$$

is called Jordan matrix.

Lemma A.2.2 (Jordan normal form ([Fi03] (p.268)))**.**
Let $A \in \mathbb{R}^{n\times n}$ such that the characteristic polynomial decomposes in linear factors

$$P_A = \pm(\lambda-\lambda_1)^{r_1}\cdot\dots\cdot(\lambda-\lambda_k)^{r_k}.$$

Then there exists a basis $\mathcal{B}$ of $\mathbb{R}^{n\times n}$, such that

$$M_{\mathcal{B}}(A) = \begin{pmatrix} \boxed{\lambda_1 E_{r_1}+J_1} & & 0 \\ & \ddots & \\ 0 & & \boxed{\lambda_k E_{r_k}+J_k} \end{pmatrix},$$

where $J_i, i = 1,\dots,k$ denotes a Jordan matrix.

Lemma A.2.3 (Matrix exponential ([Fi03] (pp. 272/3, Exercise 5)))**.**
Let $A \in \mathbb{R}^{n\times n}$.

$$exp(A) := \lim_{m\to\infty}\sum_{k=0}^{m}\frac{1}{k!}A^k.$$

A.3. Numerical mathematics

Definition A.3.1 (Newton method ([DR06] (Theorem 5.22)))**.**
Let $f:\mathbb{R}\to\mathbb{R}$ be twice continuously differentiable in a neighborhood $U := (a,b)$ of a root x^* of f ($f(x^*)=0, f'(x^*)\neq 0$). Then, the classical Newton iteration is given by

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0,1,2,\dots\ .$$

Definition A.3.2 (Simplified Newton method ([DR06] (Section 5.6.2)))**.**
The simplified Newton iteration is given by

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_0)}, \quad k = 0,1,2,\dots\ .$$

Remark A.3.3.
Particularly for systems, the evaluation of the Jacobian matrix $f'(x_k)$ in every iteration is expensive. Hence for these systems, the simplified Newton method is preferred.

Definition A.3.4 (Classical 4^{th} order Runge-Kutta method ([DR06] (Algorithm 11.28)))**.**
The classical 4^{th} order Runge-Kutta method is given by the Butcher-Tableau

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}.$$

Definition A.3.5 (RMS error [GSJJ13])**.**
Let N be the number of time steps executed by a numerical algorithm for the solution of ODEs. Let $\mathbf{x} \in \mathbb{R}^{N+1}$ denote the approximation vector and let $x : \mathbb{R}_+ \to \mathbb{R}$ denote the analytical solution. The root-mean-square (RMS) error is defined as

$$\varepsilon(\mathbf{x}) := \sqrt{\frac{\sum_{k=0}^{N+1} [x_k - x(t_k)]^2}{\sum_{k=0}^{N+1} [x(t_k)]^2}}.$$

B. Minimal working example of a coupled index-1 system

This example shows how submodels can be chosen in a coupled dynamic system. Both the monolithic and the coupled index-1 DAE system are provided. Let us consider two water containers (see Figure B.1) which are interconnected by two pipes. The containers have water levels x_1, x_2 and an amount y_1 – which is a percentage α of x_1 – continuously flows from container 1 to container 2, whereas a pump pumps an amount y_2 of water – which is a percentage β of the inflow y_1 – back to container 1.

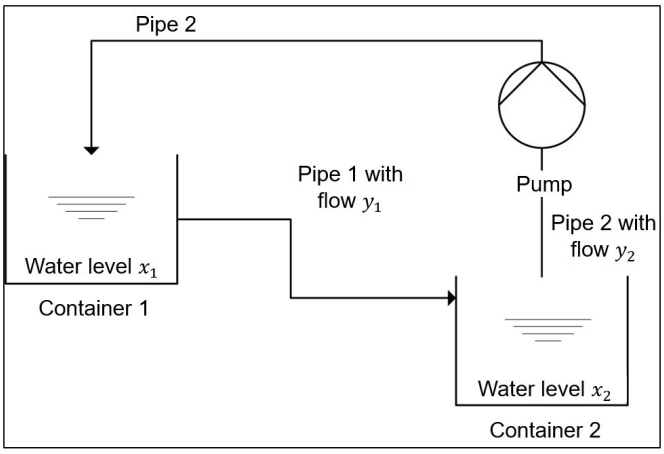


Figure B.1: Example of two interconnected water containers (sketch by Melven Röhrig-Zöllner)

Certainly this system is easily written as an ordinary differential equation:

$$\begin{cases} \dot{x}_1 &= (\beta\alpha - \alpha)x_1 \\ \dot{x}_2 &= (\alpha - \beta\alpha)x_1 \end{cases}. \tag{B.2}$$

Prescribing initial conditions $x_1(0) = x_1^0,\ x_2(0) = x_2^0$, we can also easily compute the analytic solution of system (B.2):

$$\begin{cases} x_1(t) &= x_1^0 e^{(\beta\alpha-\alpha)t} \\ x_2(t) &= -x_1^0 e^{(\beta\alpha-\alpha)t} + x_1^0 + x_2^0 \end{cases}. \tag{B.3}$$

However for our purposes, we can also formulate it as an index-1 differential algebraic equation, where each water container forms a submodel with a state, an input and an output:

$$\begin{cases} \dot{x}_1 &= y_2 - \alpha x_1 \\ \dot{x}_2 &= y_1 - \beta y_1 \\ y_1 &= \alpha x_1 (= u_2) \\ y_2 &= \beta y_1 (= u_1) \end{cases}. \tag{B.4}$$

C. The Helicopter model

We adapt the mechanical model in [SMBA11]. In contrary to the model in [SMBA11], we do not consider anisotropic blades, hence the masses and inertias of every blade are equal in our model. It contains the following variables:

Var.	Explanation	Value [unit]
a	Rotor eccentricity	0.2 [m]
b	Blade length	2.5 [m]
F	External force vector of the original rotor model	
F_{DAE}	External force vector of the advanced rotor model in DAE formulation	
F_{mon}	External force vector of the advanced monolithic rotor model	
G	Damping matrix of the original rotor model	
G_{DAE}	Damping matrix of the advanced rotor model in DAE formulation	
G_{mon}	Damping matrix of the advanced monolithic rotor model	
I_{zb}	Lag rotational inertia of a blade around its center of gravity	259 [kg m^2]
K	Stiffness matrix of the original rotor model	
K_{DAE}	Stiffness matrix of the advanced rotor model in DAE formulation	
K_{mon}	Stiffness matrix of the advanced monolithic rotor model	
K_s	Stiffness coefficient of the mast	
M	Mass matrix of the original rotor model	
M_{DAE}	Mass matrix of the advanced rotor model in DAE formulation	
M_{mon}	Mass matrix of the advanced monolithic rotor model	
m_b	Mass of a blade	31.9 [kg]
m_f	Fuselage mass	2902.9 [kg]
r_a	$\sqrt{ar_b}$	
r_m	Ratio between the static moment of a blade over the total mass of the helicopter	[m]
r_b	Ratio between the static moment over the total lead-lag rotational inertia of a blade	[m^{-1}]
$\mathbf{u}(t)$	Vector of general variables of the original rotor model	
$\mathbf{u}_{mon}(t)$	Vector of general variables of the advanced monolithic rotor model	
$x_{Fus}(t)$	Longitudinal displacement of the fuselage	[m]
$y_{Fus}(t)$	Transversal displacement of the fuselage	[m]
$\varphi_i(t)$	Lead-lag angle of i^{th} blade	[rad]
$\varphi_{RH}(t)$	Rotational angle of the rotor head	[rad]

ω_b	Lag resonance frequency of the a blade at $\Omega = 0$	1.5 [Hz]
$\omega_{RH}(t)$	Rotor head's resonance frequency	[Hz]
ω_x	Fuselage's resonance frequency in x direction	3 [Hz]
ω_y	Fuselage's resonance frequency in y direction	3 [Hz]
Ω	Rotor speed	[Hz]

Table C.1: Variables for the rotor model given in [SMBA11], Section 1.1 and Table 1

This section contains three parts. We first state the original model from [SMBA11] in Subsection C.1. In Subsection C.2, we present the advanced model in a monolithic form and subsequently we deal with the advanced model in a DAE formulation in Subsection C.3.

C.1. Original model

The matrices of the original model are

$$M = \begin{pmatrix} 1 & 0 & -r_m\sin(\Omega t) & -r_m\cos(\Omega t) & r_m\sin(\Omega t) & r_m\cos(\Omega t) \\ 0 & 1 & r_m\cos(\Omega t) & -r_m\sin(\Omega t) & -r_m\cos(\Omega t) & r_m\sin(\Omega t) \\ -r_b\sin(\Omega t) & r_b\cos(\Omega t) & 1 & 0 & 0 & 0 \\ -r_b\cos(\Omega t) & -r_b\sin(\Omega t) & 0 & 1 & 0 & 0 \\ r_b\sin(\Omega t) & -r_b\cos(\Omega t) & 0 & 0 & 1 & 0 \\ r_b\cos(\Omega t) & r_b\sin(\Omega t) & 0 & 0 & 0 & 1 \end{pmatrix},$$

$$G = \begin{pmatrix} 0 & 0 & -2r_m\Omega\cos(\Omega t) & 2r_m\Omega\sin(\Omega t) & 2r_m\Omega\cos(\Omega t) & -2r_m\Omega\sin(\Omega t) & \\ 0 & 0 & -2r_m\Omega\sin(\Omega t) & -2r_m\Omega\cos(\Omega t) & 2r_m\Omega\sin(\Omega t) & 2r_m\Omega\cos(\Omega t) & \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix},$$

$$K = \begin{pmatrix} \omega_x^2 & 0 & \Omega^2 r_m\sin(\Omega t) & \Omega^2 r_m\cos(\Omega t) & -\Omega^2 r_m\sin(\Omega t) & -\Omega^2 r_m\cos(\Omega t) \\ 0 & \omega_y^2 & -\Omega^2 r_m\cos(\Omega t) & \Omega^2 r_m\sin(\Omega t) & \Omega^2 r_m\cos(\Omega t) & -\Omega^2 r_m\sin(\Omega t) \\ 0 & 0 & \Omega^2 a r_b + \omega_b^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \Omega^2 a r_b + \omega_b^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \Omega^2 a r_b + \omega_b^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Omega^2 a r_b + \omega_b^2 \end{pmatrix}$$

and

$$F = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}.$$

The model with

$$\mathbf{u}(t) = (x_{Fus}(t), y_{Fus}(t), \varphi_1(t), \varphi_2(t), \varphi_3(t), \varphi_4(t))^T$$

reads

$$M\ddot{\mathbf{u}}(t) + G\dot{\mathbf{u}}(t) + K\mathbf{u}(t) = F. \tag{C.1.1}$$

With

$$\mathbf{v}(t) := (\dot{x}_{Fus}(t), \dot{y}_{Fus}(t), \dot{\varphi}_1(t), \dot{\varphi}_2(t), \dot{\varphi}_3(t), \dot{\varphi}_4(t))^T,$$

we transform system (C.1.1) into a first order system in the variables $\mathbf{u}(t)$ and $\mathbf{v}(t)$

$$\begin{cases} \dot{\mathbf{u}}(t) = & \mathbf{v}(t) \\ \dot{\mathbf{v}}(t) = & -M^{-1}G\mathbf{v}(t) - M^{-1}K\mathbf{u}(t) + M^{-1}F. \end{cases}$$

C.2. Advanced model in monolithic form

In the advanced model, we have additional springs at both ends of the mast that connects the fuselage with the rotor. We need additional constants and variables to model the occurring behavior. Let

- K_s denote the stiffness coefficient of the mast
- $\varphi_{RH}(t)$ denote the rotational angle of the rotor head.

Then we define

$$\begin{aligned} \omega_i(t) &:= \dot{\varphi}_i(t),\ i = 1, \ldots, 4, \\ \omega_{RH}(t) &:= \dot{\varphi}_{RH}(t), \\ \alpha_{RH}(t) &:= \dot{\omega}_{RH}(t), \\ \omega_s &:= \sqrt{\frac{K_s}{4(a+b)^2 m_b + 4I_{zb}}}, \qquad \text{(C.2.1)} \\ r_{sb} &:= \frac{b(a+b)m_b + I_{zb}}{4(a+b)^2 m_b + 4I_{zb}}. \qquad \text{(C.2.2)} \end{aligned}$$

The model matrices become

$$M_{mon} = \begin{pmatrix} 1 & 0 & -r_m\sin(\varphi_{RH}(t)) & -r_m\cos(\varphi_{RH}(t)) & r_m\sin(\varphi_{RH}(t)) \\ 0 & 1 & r_m\cos(\varphi_{RH}(t)) & -r_m\sin(\varphi_{RH}(t)) & -r_m\cos(\varphi_{RH}(t)) \\ -r_b\sin(\varphi_{RH}(t)) & r_b\cos(\varphi_{RH}(t)) & 1 & 0 & 0 \\ -r_b\cos(\varphi_{RH}(t)) & -r_b\sin(\varphi_{RH}(t)) & 0 & 1 & 0 \\ r_b\sin(\varphi_{RH}(t)) & -r_b\cos(\varphi_{RH}(t)) & 0 & 0 & 1 \\ r_b\cos(\varphi_{RH}(t)) & r_b\sin(\varphi_{RH}(t)) & 0 & 0 & 0 \\ 0 & 0 & r_{sb} & r_{sb} & r_{sb} \end{pmatrix} \tag{C.2.3}$$

$$\begin{pmatrix} r_m\cos(\varphi_{RH}(t)) & -r_m((\varphi_1(t)-\varphi_3(t))\cos(\varphi_{RH}(t))-\sin(\varphi_{RH}(t))(\varphi_2(t)-\varphi_4(t))) \\ r_m\sin(\varphi_{RH}(t)) & -r_m((\varphi_2(t)-\varphi_4(t))\cos(\varphi_{RH}(t))+\sin(\varphi_{RH}(t))(\varphi_1(t)-\varphi_3(t))) \\ 0 & r_a^2+1 \\ 0 & r_a^2+1 \\ 0 & r_a^2+1 \\ 1 & r_a^2+1 \\ r_{sb} & 1 \end{pmatrix},$$

$$G_{mon} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{C.2.4}$$

and

$$K_{mon} = \begin{pmatrix} \omega_x^2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \omega_y^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & r_a^2\omega_{RH}^2(t)+\omega_b^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & r_a^2\omega_{RH}^2(t)+\omega_b^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & r_a^2\omega_{RH}^2(t)+\omega_b^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & r_a^2\omega_{RH}^2(t)+\omega_b^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \omega_s^2 \end{pmatrix}. \tag{C.2.5}$$

With $\varphi_{2,4} = \varphi_2(t) - \varphi_4(t)$, $\varphi_{3,1} = \varphi_3(t) - \varphi_1(t)$, $\varphi_{RH} := \varphi_{RH}(t)$ $\omega_{RH} := \omega_{RH}(t)$ and $\omega_i := \omega_i(t)$,

the right side of the monolithic model F_{mon} takes the form

$$F_{mon} = \begin{pmatrix} -r_m((\varphi_{2,4}\omega_{RH} - 2\omega_1 + 2\omega_3)\cos(\varphi_{RH}) - (\varphi_{3,1}\omega_{RH} - 2\omega_2 + 2\omega_4)\sin(\varphi_{RH}))\omega_{RH} \\ -r_m((\varphi_{3,1}\omega_{RH} - 2\omega_2 + 2\omega_4)\cos(\varphi_{RH}) + (\varphi_{2,4}\omega_{RH} - 2\omega_1 + 2\omega_3)\sin(\varphi_{RH}))\omega_{RH} \\ 0 \\ 0 \\ 0 \\ 0 \\ \omega_s^2 \Omega t \end{pmatrix}. \tag{C.2.6}$$

The model with

$$\mathbf{u}_{mon}(t) = (x_{Fus}(t), y_{Fus}(t), \varphi_1(t), \varphi_2(t), \varphi_3(t), \varphi_4(t), \varphi_{RH}(t))^T \tag{C.2.7}$$

reads

$$M_{mon}\ddot{\mathbf{u}}_{mon}(t) + G_{mon}\dot{\mathbf{u}}_{mon}(t) + K_{mon}\mathbf{u}_{mon}(t) = F_{mon}. \tag{C.2.8}$$

With

$$\mathbf{v}_{mon}(t) = (\dot{x}_{Fus}(t), \dot{y}_{Fus}(t), \dot{\varphi}_1(t), \dot{\varphi}_2(t), \dot{\varphi}_3(t), \dot{\varphi}_4(t), \dot{\varphi}_{RH}(t))^T, \tag{C.2.9}$$

we transform system (C.2.8) into a first order system in the variables $\mathbf{u}_{mon}(t)$ and $\mathbf{v}_{mon}(t)$

$$\begin{cases} \dot{\mathbf{u}}_{mon}(t) = & \mathbf{v}_{mon}(t) \\ \dot{\mathbf{v}}_{mon}(t) = & -M_{mon}^{-1}G_{mon}\mathbf{v}_{mon}(t) - M_{mon}^{-1}K_{mon}\mathbf{u}_{mon}(t) + M_{mon}^{-1}F_{mon}. \end{cases}$$

Defining

$$\mathbf{x}_{mon} := \begin{pmatrix} \mathbf{u}_{mon}(t) \\ \mathbf{v}_{mon}(t) \end{pmatrix}, \tag{C.2.10}$$

we obtain a 14-dimensional linear monolithic system

$$\dot{\mathbf{x}}_{mon} = \begin{pmatrix} \mathbf{O}_{7\times 7} & \mathbf{I}_{7\times 7} \\ -M_{mon}^{-1}K_{mon} & -M_{mon}^{-1}G_{mon} \end{pmatrix} \cdot \mathbf{x}_{mon} + \begin{pmatrix} \mathbf{0}_7 \\ M_{mon}^{-1}F_{mon} \end{pmatrix} =: f_{mon}(t, \mathbf{x}_{mon}(t)), \tag{C.2.11}$$

where $\mathbf{O}_{7\times 7}$ denotes the 7×7-zero matrix, $\mathbf{I}_{7\times 7}$ denotes the 7×7-identity matrix and $\mathbf{0}_7$ denotes the 7-dimensional zero vector.

Remark C.2.12.
For the implementation, we use the variables $\bar{\varphi}_{RH}(t) := \varphi_{RH}(t) - \Omega t$ and $\bar{\omega}_{RH}(t) := \dot{\bar{\varphi}}_{RH}(t) = \omega_{RH}(t) - \Omega$. Accordingly, all entries of M_{mon}, K_{mon} and F_{mon} containing $\varphi_{RH}(t)$ or $\omega_{RH}(t)$ need to be adapted. Additionally, the last entry of F_{mon} becomes zero.

C.3. Advanced model in DAE form

The model matrices for the advanced model in DAE formulation read

$$M_{DAE} = \begin{pmatrix} 1 & 0 & -r_m\sin(\varphi_{RH}(t)) & -r_m\cos(\varphi_{RH}(t)) & r_m\sin(\varphi_{RH}(t)) & r_m\cos(\varphi_{RH}(t)) \\ 0 & 1 & r_m\cos(\varphi_{RH}(t)) & -r_m\sin(\varphi_{RH}(t)) & -r_m\cos(\varphi_{RH}(t)) & r_m\sin(\varphi_{RH}(t)) \\ -r_b\sin(\varphi_{RH}(t)) & r_b\cos(\varphi_{RH}(t)) & 1 & 0 & 0 & 0 \\ -r_b\cos(\varphi_{RH}(t)) & -r_b\sin(\varphi_{RH}(t)) & 0 & 1 & 0 & 0 \\ r_b\sin(\varphi_{RH}(t)) & -r_b\cos(\varphi_{RH}(t)) & 0 & 0 & 1 & 0 \\ r_b\cos(\varphi_{RH}(t)) & r_b\sin(\varphi_{RH}(t)) & 0 & 0 & 0 & 1 \end{pmatrix}, \tag{C.3.1}$$

$$G_{DAE} = \begin{pmatrix} 0 & 0 & -2r_m\omega_{RH}(t)\cos(\varphi_{RH}(t)) & 2r_m\omega_{RH}(t)\sin(\varphi_{RH}(t)) & 2r_m\omega_{RH}(t)\cos(\varphi_{RH}(t)) & -2r_m\omega_{RH}(t)\sin(\varphi_{RH}(t)) \\ 0 & 0 & -2r_m\omega_{RH}(t)\sin(\varphi_{RH}(t)) & -2r_m\omega_{RH}(t)\cos(\varphi_{RH}(t)) & 2r_m\omega_{RH}(t)\sin(\varphi_{RH}(t)) & 2r_m\omega_{RH}(t)\cos(\varphi_{RH}(t)) \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}, \tag{C.3.2}$$

and

$$K_{DAE} = \begin{pmatrix} \omega_x^2 & 0 & -r_m(-\omega_{RH}^2(t)\sin(\varphi_{RH}(t))+\alpha_{RH}(t)\cos(\varphi_{RH}(t))) & r_m(\omega_{RH}^2(t)\cos(\varphi_{RH}(t))+\alpha_{RH}(t)\sin(\varphi_{RH}(t))) \\ 0 & \omega_y^2 & -r_m(\omega_{RH}^2(t)\cos(\varphi_{RH}(t))+\alpha_{RH}(t)\sin(\varphi_{RH}(t))) & -r_m(-\omega_{RH}^2(t)\sin(\varphi_{RH}(t))+\alpha_{RH}(t)\cos(\varphi_{RH}(t))) \\ 0 & 0 & r_a^2\omega_{RH}^2(t)+\omega_b^2 & 0 \\ 0 & 0 & 0 & r_a^2\omega_{RH}^2(t)+\omega_b^2 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \tag{C.3.3}$$

$$\begin{pmatrix} r_m(-\omega_{RH}^2(t)\sin(\varphi_{RH}(t))+\alpha_{RH}(t)\cos(\varphi_{RH}(t))) & -r_m(\omega_{RH}^2(t)\cos(\varphi_{RH}(t))+\alpha_{RH}(t)\sin(\varphi_{RH}(t))) \\ r_m(\omega_{RH}^2(t)\cos(\varphi_{RH}(t))+\alpha_{RH}(t)\sin(\varphi_{RH}(t))) & r_m(-\omega_{RH}^2(t)\sin(\varphi_{RH}(t))+\alpha_{RH}(t)\cos(\varphi_{RH}(t))) \\ 0 & 0 \\ 0 & 0 \\ r_a^2\omega_{RH}^2(t)+\omega_b^2 & 0 \\ 0 & r_a^2\omega_{RH}^2(t)+\omega_b^2 \end{pmatrix}.$$

The right side becomes

$$F_{DAE} = \begin{pmatrix} 0 \\ 0 \\ (-r_a^2 - 1)\alpha_{RH}(t) \\ (-r_a^2 - 1)\alpha_{RH}(t) \\ (-r_a^2 - 1)\alpha_{RH}(t) \\ (-r_a^2 - 1)\alpha_{RH}(t) \end{pmatrix}. \tag{C.3.4}$$

For the two models we then obtain:

Model 1:
Inputs: $\varphi_{RH}(t)$, $\omega_{RH}(t)$ and $\alpha_{RH}(t)$
Outputs: $\dot{\omega}_1(t), \dot{\omega}_2(t), \dot{\omega}_3(t), \dot{\omega}_4(t)$

The local state vector $\mathbf{x}_1(t)$ is given by

$$\mathbf{x}_1(t) = \begin{pmatrix} \mathbf{u}(t) \\ \mathbf{v}(t) \end{pmatrix} = \begin{pmatrix} x_{Fus}(t) \\ y_{Fus}(t) \\ \varphi_1(t) \\ \varphi_2(t) \\ \varphi_3(t) \\ \varphi_4(t) \\ \dot{x}_{Fus}(t) \\ \dot{y}_{Fus}(t) \\ \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \\ \omega_4(t) \end{pmatrix} \in \mathbb{R}^{12}.$$

This yields

$$\dot{\mathbf{x}}_1(t) = f_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)) = \begin{pmatrix} \mathbf{O}_{6\times 6} & \mathbf{I}_{6\times 6} \\ -M_{DAE}^{-1} K_{DAE} & -M_{DAE}^{-1} G_{DAE} \end{pmatrix} \mathbf{x}_1(t) + \begin{pmatrix} \mathbf{0}_6 \\ M_{DAE}^{-1} F_{DAE} \end{pmatrix},$$

where $\mathbf{O}_{6\times 6}$ denotes the 6×6-zero matrix, $\mathbf{I}_{6\times 6}$ denotes the 6×6-identity matrix and $\mathbf{0}_6$ denotes the 6-dimensional zero vector. The dependence on $\mathbf{y}_2(t) = (\varphi_{RH}(t), \omega_{RH}(t), \alpha_{RH}(t))^T$ is manifested in the definitions of M_{DAE}, K_{DAE}, G_{DAE} and F_{DAE}.

We see that the outputs of the first model ($\dot{\omega}_1(t), \dot{\omega}_2(t), \dot{\omega}_3(t), \dot{\omega}_4(t)$) are the time derivatives of the 9^{th} to 12^{th} entries of $\mathbf{x}_1(t)$. Hence, in order to state the algebraic function $g_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t))$ explicitly, we need to calculate $-M_{DAE}^{-1} K_{DAE}$ and $-M_{DAE}^{-1} G_{DAE}$:

$$\mathbf{y}_1(t) = \begin{pmatrix} \dot{\omega}_1(t) \\ \dot{\omega}_2(t) \\ \dot{\omega}_3(t) \\ \dot{\omega}_4(t) \end{pmatrix} = \begin{pmatrix} \dot{\mathbf{x}}_1^{(9)} \\ \dot{\mathbf{x}}_1^{(10)} \\ \dot{\mathbf{x}}_1^{(11)} \\ \dot{\mathbf{x}}_1^{(12)} \end{pmatrix} = \frac{1}{2r_b r_m - 1} \begin{pmatrix} \sin(\varphi_{RH}(t))r_b\omega_x^2 & -\cos(\varphi_{RH}(t))r_b\omega_y^2 \\ \cos(\varphi_{RH}(t))r_b\omega_x^2 & \sin(\varphi_{RH}(t))r_b\omega_y^2 \\ -\sin(\varphi_{RH}(t))r_b\omega_x^2 & \cos(\varphi_{RH}(t))r_b\omega_y^2 \\ -\cos(\varphi_{RH}(t))r_b\omega_x^2 & -\sin(\varphi_{RH}(t))r_b\omega_y^2 \end{pmatrix}$$

$$\begin{pmatrix} ((-r_a^2+1)r_m r_b + r_a^2)\omega_{RH}(t)^2 - \omega_b^2(r_b r_m - 1) & r_b r_m \alpha_{RH}(t) \\ -r_b r_m \alpha_{RH}(t) & ((-r_a^2+1)r_m r_b + r_a^2)\omega_{RH}(t)^2 - \omega_b^2(r_b r_m - 1) \\ -r_b r_m(\omega_{RH}(t)^2 + r_a^2\omega_{RH}(t)^2 + \omega_b^2) & -r_b r_m \alpha_{RH}(t) \\ r_b r_m \alpha_{RH}(t) & -r_b r_m(\omega_{RH}(t)^2 + r_a^2\omega_{RH}(t)^2 + \omega_b^2) \end{pmatrix}$$

$$\begin{pmatrix} -r_b r_m(\omega_{RH}(t)^2 + r_a^2\omega_{RH}(t)^2 + \omega_b^2) & -r_b r_m \alpha_{RH}(t) \\ r_b r_m \alpha_{RH}(t) & -r_b r_m(\omega_{RH}(t)^2 + r_a^2\omega_{RH}(t)^2 + \omega_b^2) \\ ((-r_a^2+1)r_m r_b + r_a^2)\omega_{RH}(t)^2 - \omega_b^2(r_b r_m - 1) & r_b r_m \alpha_{RH}(t) \\ -r_b r_m \alpha_{RH}(t) & ((-r_a^2+1)r_m r_b + r_a^2)\omega_{RH}(t)^2 - \omega_b^2(r_b r_m - 1) \end{pmatrix}$$

$$\cdot \begin{pmatrix} x_{Fus}(t) \\ y_{Fus}(t) \\ \varphi_1(t) \\ \varphi_2(t) \\ \varphi_3(t) \\ \varphi_4(t) \end{pmatrix} + \frac{2\omega_{RH}(t) r_b r_m}{2r_b r_m - 1} \begin{pmatrix} 0 & 1 & 0 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 1 & 0 & -1 & 0 \end{pmatrix} \cdot \begin{pmatrix} \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \\ \omega_4(t) \end{pmatrix} - (r_a^2+1)\alpha_{RH}(t) \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$=: g_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)). \tag{C.3.5}$$

From the first model we do not extract a stiff linear part. Hence,

$$S_1 := \mathbf{O}_{12\times 12} \tag{C.3.6}$$

$$\tilde{f}_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)) := f_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)). \tag{C.3.7}$$

Model 2:
Inputs: $\dot{\omega}_1(t), \dot{\omega}_2(t), \dot{\omega}_3(t), \dot{\omega}_4(t)$
Outputs: $\varphi_{RH}(t)$, $\omega_{RH}(t)$ and $\alpha_{RH}(t)$

The governing equation of the second model reads

$$\dot{\omega}_{RH}(t) = \omega_s^2(\Omega t - \varphi_{RH}(t)) - r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t).$$

With the local state vector

$$\mathbf{x}_2(t) := \begin{pmatrix} \varphi_{RH}(t) \\ \omega_{RH}(t) \end{pmatrix} \in \mathbb{R}^2,$$

this yields

$$\dot{\mathbf{x}}_2(t) = \begin{pmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{pmatrix} \mathbf{x}_2(t) + \begin{pmatrix} 0 \\ \omega_s^2 \Omega t - r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t) \end{pmatrix}.$$

Since ω_s^2 is the variable which is responsible for the stiffness of the system, we want it to be solely part of the first linear summand of the differential equations system. Hence we define

$$\begin{aligned} \mathbf{x}_2^{(1)} &:= \bar{\varphi}_{RH}(t) := \varphi_{RH}(t) - \Omega t, \\ \mathbf{x}_2^{(2)} &:= \bar{\omega}_{RH}(t) := \dot{\bar{\varphi}}_{RH}(t) = \bar{\varphi}_{RH}(t) - \Omega, \end{aligned}$$

which yields

$$\dot{\mathbf{x}}_2(t) = \begin{pmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{pmatrix} \mathbf{x}_2(t) + \begin{pmatrix} 0 \\ -r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t) \end{pmatrix}.$$

Accordingly, we define

$$S_2 := \begin{pmatrix} 0 & 1 \\ -\omega_s^2 & 0 \end{pmatrix} \tag{C.3.8}$$

and

$$\tilde{f}_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t)) := \begin{pmatrix} 0 \\ -r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t) \end{pmatrix}. \tag{C.3.9}$$

The local output vector and hence the local output function $g_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t))$ is given by

$$\begin{aligned} \mathbf{y}_2(t) &= \begin{pmatrix} \varphi_{RH}(t) \\ \omega_{RH}(t) \\ \alpha_{RH}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{x}_2^{(1)}(t) + \Omega t \\ \mathbf{x}_2^{(2)}(t) + \Omega \\ \dot{\mathbf{x}}_2^{(2)}(t) \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{x}_2^{(1)}(t) + \Omega t \\ \mathbf{x}_2^{(2)}(t) + \Omega \\ -\omega_s^2 \mathbf{x}_2^{(1)}(t) - r_{sb}\dot{\omega}_1(t) - r_{sb}\dot{\omega}_2(t) - r_{sb}\dot{\omega}_3(t) - r_{sb}\dot{\omega}_4(t) \end{pmatrix} =: g_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t)). \end{aligned}$$

Global model:

For the global model we then obtain the global state vector

$$\mathbf{x}(t) = \begin{pmatrix} x_{Fus}(t) \\ y_{Fus}(t) \\ \varphi_1(t) \\ \varphi_2(t) \\ \varphi_3(t) \\ \varphi_4(t) \\ \dot{x}_{Fus}(t) \\ \dot{y}_{Fus}(t) \\ \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \\ \omega_4(t) \\ \bar{\varphi}_{RH}(t) \\ \bar{\omega}_{RH}(t) \end{pmatrix} \tag{C.3.10}$$

and our global output vector reads

$$\mathbf{y}(t) = \begin{pmatrix} \dot{\omega}_1(t) \\ \dot{\omega}_2(t) \\ \dot{\omega}_3(t) \\ \dot{\omega}_4(t) \\ \varphi_{RH}(t) \\ \omega_{RH}(t) \\ \alpha_{RH}(t) \end{pmatrix}. \tag{C.3.11}$$

The global state and output functions read

$$f_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t)) = S\mathbf{x} + \tilde{f}_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t))$$

with

$$S := \begin{pmatrix} S_1 & \mathbf{O}_{12\times 2} \\ \mathbf{O}_{2\times 12} & S_2 \end{pmatrix}, \tag{C.3.12}$$

$$\tilde{f}_{DAE}(t, \mathbf{x}(t), \mathbf{y}(t)) := \begin{pmatrix} \tilde{f}_1(t, \mathbf{x}_1(t), \mathbf{y}_2(t)) \\ \tilde{f}_2(t, \mathbf{x}_2(t), \mathbf{y}_1(t)) \end{pmatrix} \tag{C.3.13}$$

and

$$g_{DAE}(t,\mathbf{x}(t),\mathbf{y}(t)) := \begin{pmatrix} g_1(t,\mathbf{x}_1(t),\mathbf{y}_2(t)) \\ g_2(t,\mathbf{x}_2(t),\mathbf{y}_1(t)) \end{pmatrix}. \tag{C.3.14}$$

Since we use the Newton method on the function $\bar{g}_{DAE}(t,\mathbf{x}(t),\mathbf{y}(t)) := \mathbf{y}(t) - g_{DAE}(t,\mathbf{x}(t),\mathbf{y}(t))$ when applying half-explicit methods, we need to calculate the gradient $\nabla_{\mathbf{y}}\bar{g}_{DAE}$:

$$\nabla_{\mathbf{y}}\bar{g}_{DAE} = \begin{pmatrix} 1 & 0 & 0 & 0 & -\frac{\partial g^{(1)}_{DAE}}{\partial \varphi_{RH}(t)} & -\frac{\partial g^{(1)}_{DAE}}{\partial \omega_{RH}(t)} & -\frac{\partial g^{(1)}_{DAE}}{\partial \alpha_{RH}(t)} \\ 0 & 1 & 0 & 0 & -\frac{\partial g^{(2)}_{DAE}}{\partial \varphi_{RH}(t)} & -\frac{\partial g^{(2)}_{DAE}}{\partial \omega_{RH}(t)} & -\frac{\partial g^{(2)}_{DAE}}{\partial \alpha_{RH}(t)} \\ 0 & 0 & 1 & 0 & -\frac{\partial g^{(3)}_{DAE}}{\partial \varphi_{RH}(t)} & -\frac{\partial g^{(3)}_{DAE}}{\partial \omega_{RH}(t)} & -\frac{\partial g^{(3)}_{DAE}}{\partial \alpha_{RH}(t)} \\ 0 & 0 & 0 & 1 & -\frac{\partial g^{(4)}_{DAE}}{\partial \varphi_{RH}(t)} & -\frac{\partial g^{(4)}_{DAE}}{\partial \omega_{RH}(t)} & -\frac{\partial g^{(4)}_{DAE}}{\partial \alpha_{RH}(t)} \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ r_{sb} & r_{sb} & r_{sb} & r_{sb} & 0 & 0 & 1 \end{pmatrix} \tag{C.3.15}$$

with

$$\begin{aligned}
\frac{\partial g^{(1)}_{DAE}}{\partial \varphi_{RH}(t)} &= \frac{r_b\omega_x^2}{2r_br_m-1}\cos(\varphi_{RH}(t))x_{Fus}(t) + \frac{r_b\omega_y^2}{2r_br_m-1}\sin(\varphi_{RH}(t))y_{Fus}(t), \\
\frac{\partial g^{(2)}_{DAE}}{\partial \varphi_{RH}(t)} &= -\frac{r_b\omega_x^2}{2r_br_m-1}\sin(\varphi_{RH}(t))x_{Fus}(t) + \frac{r_b\omega_y^2}{2r_br_m-1}\cos(\varphi_{RH}(t))y_{Fus}(t), \\
\frac{\partial g^{(3)}_{DAE}}{\partial \varphi_{RH}(t)} &= -\frac{r_b\omega_x^2}{2r_br_m-1}\cos(\varphi_{RH}(t))x_{Fus}(t) - \frac{r_b\omega_y^2}{2r_br_m-1}\sin(\varphi_{RH}(t))y_{Fus}(t), \\
\frac{\partial g^{(4)}_{DAE}}{\partial \varphi_{RH}(t)} &= \frac{r_b\omega_x^2}{2r_br_m-1}\sin(\varphi_{RH}(t))x_{Fus}(t) - \frac{r_b\omega_y^2}{2r_br_m-1}\cos(\varphi_{RH}(t))y_{Fus}(t), \\
\frac{\partial g^{(1)}_{DAE}}{\partial \omega_{RH}(t)} &= \frac{2((-r_a^2+1)r_mr_b+r_a^2)\varphi_1(t) - 2r_br_m(1+r_a^2)\varphi_3(t)}{2r_br_m-1}\omega_{RH}(t) + \frac{2r_br_m}{2r_br_m-1}(\omega_2(t)-\omega_4(t)), \\
\frac{\partial g^{(2)}_{DAE}}{\partial \omega_{RH}(t)} &= \frac{2((-r_a^2+1)r_mr_b+r_a^2)\varphi_2(t) - 2r_br_m(1+r_a^2)\varphi_4(t)}{2r_br_m-1}\omega_{RH}(t) + \frac{2r_br_m}{2r_br_m-1}(\omega_3(t)-\omega_1(t)), \\
\frac{\partial g^{(3)}_{DAE}}{\partial \omega_{RH}(t)} &= \frac{2((-r_a^2+1)r_mr_b+r_a^2)\varphi_3(t) - 2r_br_m(1+r_a^2)\varphi_1(t)}{2r_br_m-1}\omega_{RH}(t) + \frac{2r_br_m}{2r_br_m-1}(\omega_4(t)-\omega_2(t)), \\
\frac{\partial g^{(4)}_{DAE}}{\partial \omega_{RH}(t)} &= \frac{2((-r_a^2+1)r_mr_b+r_a^2)\varphi_4(t) - 2r_br_m(1+r_a^2)\varphi_2(t)}{2r_br_m-1}\omega_{RH}(t) + \frac{2r_br_m}{2r_br_m-1}(\omega_1(t)-\omega_3(t)),
\end{aligned}$$

$$\frac{\partial g_{DAE}^{(1)}}{\partial \alpha_{RH}(t)} = \frac{r_b r_m}{2r_b r_m - 1}(\varphi_2(t) - \varphi_4(t)) - (r_a^2 + 1),$$
$$\frac{\partial g_{DAE}^{(2)}}{\partial \alpha_{RH}(t)} = \frac{r_b r_m}{2r_b r_m - 1}(\varphi_3(t) - \varphi_1(t)) - (r_a^2 + 1),$$
$$\frac{\partial g_{DAE}^{(3)}}{\partial \alpha_{RH}(t)} = \frac{r_b r_m}{2r_b r_m - 1}(\varphi_4(t) - \varphi_2(t)) - (r_a^2 + 1),$$
$$\frac{\partial g_{DAE}^{(4)}}{\partial \alpha_{RH}(t)} = \frac{r_b r_m}{2r_b r_m - 1}(\varphi_1(t) - \varphi_3(t)) - (r_a^2 + 1).$$

D. Matlab Code remarks

In order to simplify the Matlab code for the stability plots in Subsection 2.3.5, we derive the general form of a stability function of explicit s-stage exponential Runge-Kutta methods. We consider one-dimensional problems for the plots. Since we only consider explicit methods up to order 4 (5 stages), we can simply insert the explicit equations for the $\widetilde{F}_{ni}$ (as in (2.3.14)) directly ($a_{ij} := a_{ij}(h\sigma) \in \mathbb{R}$):

$$
\begin{aligned}
\widetilde{F}_{n1} &= \underbrace{\lambda}_{=:\xi_1} x_n, \\
\widetilde{F}_{n2} &= \lambda(x_n + ha_{21}(\widetilde{F}_{n1} + \sigma x_n)) \\
&= \lambda(x_n + ha_{21}(\xi_1 x_n + \sigma x_n)) \\
&= \underbrace{\{\lambda + \lambda h a_{21}(\xi_1 + \sigma)\}}_{=:\xi_2} x_n, \\
\widetilde{F}_{n3} &= \lambda(x_n + h[a_{31}(\widetilde{F}_{n1} + \sigma x_n) + a_{32}(\widetilde{F}_{n2} + \sigma x_n))]) \\
&= \lambda(x_n + h[a_{31}(\xi_1 x_n + \sigma x_n) + a_{32}(\xi_2 x_n + \sigma x_n)]) \\
&= \underbrace{\{\lambda + \lambda h a_{31}(\xi_1 + \sigma) + \lambda h a_{32}(\xi_2 + \sigma)\}}_{=:\xi_3} x_n, \\
\widetilde{F}_{n4} &= \lambda(x_n + h[a_{41}(\widetilde{F}_{n1} + \sigma x_n) + a_{42}(\widetilde{F}_{n2} + \sigma x_n) + a_{43}(\widetilde{F}_{n3} + \sigma x_n)]) \\
&= \lambda(x_n + h[a_{41}(\xi_1 x_n + \sigma x_n) + a_{42}(\xi_2 x_n + \sigma x_n) + a_{43}(\xi_3 x_n + \sigma x_n)]) \\
&= \underbrace{\{\lambda + \lambda h a_{41}(\xi_1 + \sigma) + \lambda h a_{42}(\xi_2 + \sigma) + \lambda h a_{43}(\xi_3 + \sigma)\}}_{=:\xi_4} x_n, \\
\widetilde{F}_{n5} &= \lambda(x_n + h[a_{51}(\widetilde{F}_{n1} + \sigma x_n) + a_{52}(\widetilde{F}_{n2} + \sigma x_n) + a_{53}(\widetilde{F}_{n3} + \sigma x_n) + a_{54}(\widetilde{F}_{n4} + \sigma x_n)]) \\
&= \lambda(x_n + h[a_{51}(\xi_1 x_n + \sigma x_n) + a_{52}(\xi_2 x_n + \sigma x_n) + a_{53}(\xi_3 x_n + \sigma x_n) \\
&\quad + a_{54}(\xi_4 x_n + \sigma x_n)]) \\
&= \underbrace{\{\lambda + \lambda h a_{51}(\xi_1 + \sigma) + \lambda h a_{52}(\xi_2 + \sigma) + \lambda h a_{53}(\xi_3 + \sigma) + \lambda h a_{54}(\xi_4 + \sigma)\}}_{=:\xi_5} x_n.
\end{aligned}
$$

In general, for explicit methods, we can compute ξ_i as

$$
\xi_i = \lambda + \sum_{j=1}^{i-1} \lambda h a_{ij}(\xi_j + \sigma). \tag{D.1}
$$

So, short with the above definitions:

$$
\widetilde{F}_{ni} = \xi_i x_n. \tag{D.2}
$$

Hence, inserting (D.2) into (2.3.7a), we obtain

$$\begin{aligned} x_{n+1} &= x_n + h\sum_{i=1}^{s} b_i(h\sigma)(\widetilde{F}_{ni} + \sigma x_n) \\ &= x_n + h\sum_{i=1}^{s} b_i(h\sigma)(\xi_i x_n + \sigma x_n) \\ &= \left[1 + h\sum_{i=1}^{s} b_i(h\sigma)(\xi_i + \sigma)\right] x_n. \end{aligned}$$

So, the stability function for an explicit s-stage exponential Runge-Kutta method reads

$$\phi = 1 + h\sum_{i=1}^{s} b_i(h\sigma)(\xi_i + \sigma). \tag{D.3}$$

References

[AHMEC] American Helicopter Museum & Education Center, History of helicopters, available from `http://americanhelicopter.museum/exhibits-and-resources/history-helicopters` (29.03.2017).

[ARW93] Ascher, U. M., Ruuth, S. J., Wetton, B. T. R., Implicit-explicit methods for time-dependent PDE's, University of British Columbia, Department of Computer Science, 1993.

[ASW93] Arnold, M., Strehmel, K., Weiner, R., Half-explicit Runge-Kutta methods for semi-explicit differential-algebraic equations of index 1, Numerische Mathematik, Volume 64, Issue 1, pp. 409–431, 1993.

[BFR16] Boscarino, S., Filbet, F., Russo, G., High order semi-implicit schemes for time dependent partial differential equations, Journal of Scientific Computing, Volume 68, Issue 3, pp. 1–27, 2016.

[Bi09] Bittner, W., Flugmechanik der Hubschrauber: Technologie, das flugdynamische System Hubschrauber, Flugstabilitäten, Steuerbarkeit, Springer Science & Business Media, 2009.

[BK93] Bauchau, O. A., Kang, N. K., A multibody formulation for helicopter structural dynamic analysis, Journal of the American Helicopter Society, Volume 38, Issue 2, pp. 3–14, 1993.

[BKV98] Beylkin, G., Keiser, J. M., Vozovoi, L., A new class of time discretization schemes for the solution of nonlinear PDEs, Journal of Computational Physics, Volume 147, Issue 2, pp. 362–387, 1998.

[BKMS17] Bab, S., Khadem, S. E., Mahdiabadi, M. K., Shahgholi, M., Vibration mitigation of a rotating beam under external periodic force using a nonlinear energy sink (NES), Journal of Vibration and Control, Volume 23, Issue 6, pp. 1001–1025, 2017.

[BN01] Bachau, O. A., Nikishkov, Y. G., An implicit Floquet analysis for rotorcraft stability evaluation, Journal of the American Helicopter Society, Volume 46, Issue 3, pp 200–209, 2001.

[Bo07] Boscarino, S., Error analysis of IMEX Runge-Kutta methods derived from differential-algebraic systems, SIAM Journal on Numerical Analysis, Volume 45, Issue 4, pp. 1600–1621, 2007.

[Br01] Bramwell, A. R. S., Done, G., Balmford, D., Bramwell's Helicopter Dynamics, Butterworth-Heinemann, 2nd edition, 2001.

[BR09] Boscarino, S., Russo, G., On a class of uniformily accurate IMEX Runge-Kutta schemes and applications to hyperbolic systems with relaxation, SIAM Journal on Scientific Computing, Volume 31, Issue 3, pp. 1926–1945, 2009.

[BT99] Bendtsen, C., Thomsen, P. G., Numerical solution of differential algebraic equations, IMM, Department of Mathematical Modeling, Technical University of Denmark, 1999.

[Bu96] Butcher, J. C., A history of Runge-Kutta methods, Applied Numerical Mathematics, Volume 20, Issue 3, pp. 247–260, 1996.

[BW96] Butcher, J. C., Wanner, G., Runge-Kutta methods: some historical notes, Applied Numerical Mathematics, Volume 22, Issues 1–3, pp. 113–151, 1996.

[BW10] Bachau, O. A., Wang, J., Efficient and robust approaches for rotorcraft stability analysis, Journal of the American Helicopter Society, Volume 55, Issue 3, pp. 32006-1–32006-9, 2010.

[CFN01] Calvo, M. P., De Frutos, J., Novo, J., Linearly implicit Runge-Kutta methods for advection-reaction-diffusion equations, Applied Numerical Mathematics, Volume 37, Issue 4, pp. 535–549, 2001.

[CH52] Curtiss, C. F., Hirschfelder, J. O., Integration of stiff equations, Proceedings of the National Academy of Sciences, Volume 38, Issue 3, pp. 235–243, 1952.

[CM02] Cox, S. M., Matthews, P. C., Exponential time differencing for stiff systems, Journal of Computational Physics, Volume 176, Issue 2, pp. 430–455, 2002.

[DB08] Deuflhard, P., Bornemann, F., Numerische Mathematik 2 - Gewöhnliche Differentialgleichungen, 3. Auflage, De Gruyter, 2008.

[DR06] Dahmen, W., Reusken, A., Numerik für Ingenieure und Naturwissenschaftler, Springer-Verlag, 2006.

[FD99] Fornberg, B., Driscoll, T. A., A fast spectral algorithm for nonlinear wave equations with linear dispersion, Journal of Computational Physics, Volume 155, Issue 2, pp. 456–467, 1999.

[Fi03] Fischer, G., Lineare Algebra, Friedr. Vieweg & Sohn Verlag, 14. Auflage, 2003.

[Fo13] Forster, O., Analysis 2, Springer, 10. Auflage, 2013.

[Fo16] Forster, O., Analysis 1, Springer, 12. Auflage, 2016.

[Fo17] Forster, O., Analysis 3, Springer, 8. Auflage, 2017.

[Fr86] Friedmann, P. P., Numerical methods for determining the stability and response of periodic systems with applications to helicopter rotor dynamics and aeroelasticity,

Computers and Mathematics with Applications, Volume 12, Issue 1, pp. 131–148, 1986.

[Fr08] Frank, J., Numerical modelling of dynamical systems, Centrum Wiskunde & Informatica (CWI), Lecture notes, Chapter 10 (Stability), available from `http://www.staff.science.uu.nl/~frank011/Classes/numwisk/` (23.11.2016), 2008.

[GSJJ13] Gasmi, A., Sprague, M., Jonkman, J., Jones, W., Numerical stability and accuracy of temporally coupled multi-physics modules in wind turbine CAE tools, 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, 2013.

[GW08] Griewank, A., Walther, A., Evaluating derivatives: principles and techniques of algorithmic differentiation, Society for Industrial and Applied Mathematics, 2nd edition, 2008.

[HLR89] Hairer, E., Lubich, C., Roche, M., The numerical solution of differential-algebraic systems by Runge-Kutta methods, Springer-Verlag, 1989.

[HO05a] Hochbruck, M., Ostermann, A., Explicit exponential Runge-Kutta methods for semilinear parabolic problems, SIAM Journal on Numerical Analysis, Volume 43, Issue 3, pp. 1069–1090, 2005.

[HO05b] Hochbruck, M., Ostermann, A., Exponential Runge-Kutta methods for parabolic problems, Applied Numerical Mathematics, Volume 53, Issues 2-4, pp. 323–339, 2005.

[HO10] Hochbruck, M., Ostermann, A., Exponential integrators, Acta Numerica, Volume 19, pp. 209–286, 2010.

[HR07] Hundsdorfer, W., Ruuth, S. J., IMEX extensions of linear multistep methods with general monotonicity and boundedness properties, Journal of Computational Physics, Volume 225, Issue 2, pp. 2016–2042, 2007.

[HSAT16] Helicopter Safety Analysis Team of IHSTI-CIS, Helicopter accidents: statistics, trends and causes, Presentation slides for IHST Regional Partners Panel, Louisville, Kentucky, USA, 2016.

[HW96] Hairer, E., Wanner, G., Solving ordinary differential equations II. Stiff and differential-algebraic problems, Springer-Verlag, second revised edition, 1996.

[Jo12] Johnson, W., Helicopter theory, Courier Corporation, 2012.

[Jo13] Johnson, W., A history of rotorcraft comprehensive analyses, NASA Ames Research Center, 2013.

[Ko08] Koto, T., Stability of IMEX Runge-Kutta methods for delay differential equations, Journal of Computational and Applied Mathematics, Volume 211, Issue 2, pp. 201–212, 2008.

[Ku13] Kunoth, A., Numerik II - Wissenschaftliches Rechnen I, Vorlesungsskript, Universität zu Köln, Wintersemester 2013/14.

[La73] Lambert, J. D., Computational methods in ordinary differential equations, John Wiley & Sons, 1973.

[La91] Lambert, J. D., Numerical methods for ordinary differential systems: the initial value problem, John Wiley & Sons, 1991.

[MZ09] Maset, S., Zennaro, M., Unconditional stability of explicit exponential Runge-Kutta methods for semilinear ordinary differential equations, Mathematics of Computation, Volume 78, Issue 266, pp. 957–967, 2009.

[MZ13] Maset, S., Zennaro, M., Stability properties of explicit exponential Runge-Kutta methods, IMA Journal of Numerical Analysis, Volume 33, Issue 1, pp. 111–135, 2013.

[OP16] Owolabi, K. M., Patidar, K. C., Numerical simulations of multicomponent ecological models with adaptive methods, Theoretical Biology & Medical Modelling, Volume 13, Issue 1, doi: 10.1186/s12976-016-0027-4, 2016.

[Pa07] Padfield, G. D., Helicopter flight dynamics, Blackwell Publishing, 2007.

[Pe94] Peters, D. A., Fast Floquet Theory and Trim for Multi-Bladed Rotorcraft, Journal of the American Helicopter Society, Volume 39, Issue 4, pp. 82–89, 1994.

[Re08] Reinhardt, H.-J., Arbeitsmaterialien Analysis I + II, Lecture notes, University Siegen, 2007&2008.

[RKM] No author, Runge-Kutta Methods, available from `http://web.mit.edu/course/16/16.90/OldFiles/BackUp/www/pdfs/Chapter10.pdf` (16.02.2017), no date.

[Ru08] Rump, W., Lecture HM III for aer, autip, verf, wewi, (folien Jan13) available from `http://www.mathematik.uni-stuttgart.de/studium/infomat/HM-Rump-WS0708/Folien/` (16.02.2017), winter term 2007/8.

[SN11] Seddon, J. M., Newman, S., Basic helicopter aerodynamics, John Wiley & Sons, Volume 40, 2011.

[SMBA11] Sanches L., Michon G., Berlioz A., Alazard D., Instability zones for isotropic and anisotropic multibladed rotor configurations, Mechanism and Machine Theory, Volume 46, Issue 8, pp. 1054–1065, 2011.

[Sp96] Spijker, M. N., Stiffness in numerical initial-value problems, Journal of Computational and Applied Mathematics, Volume 72, Issue 2, pp. 393–406, 1996.

[Sp98] Spijker, M. N., Numerical stability - stability estimates and resolvent sonditions in the numerical solution of initial value problems, Lecture Notes, University of Leider, 1998.

[Tr11] Trefethen, N., Stability regions of ODE formulas, File Exchange, MathWorks, available from `https://de.mathworks.com/matlabcentral/fileexchange/23972-chebfun/content/chebfun/examples/ode/html/Regions.html` (16.02.2017), 2011.

[Wa05] Wagner, S., Flow-structure interactions on helicopter rotors in forward flight, GAMM-Mitteilungen, Volume 28, Issue 1, pp. 7–36, 2005.

[ZSB16] Zhang, H., Sandu, A., Blaise, S., High order implicit-explicit general linear methods with optimized stability regions, SIAM Journal on Scientific Computing, Volume 38, Issue 3, pp. A1430–A1453, 2016.

[Zh17a] Zhang, R. et al., Steady-state-preserving simulation of genetic regulatory systems, Computational and Mathematical Methods in Medicine, doi:10.1155/2017/2729683, 2017.

[Zh17b] Zhu, L., Efficient and stable exponential Runge-Kutta methods for parabolic equations, Advances in Applied Mathematics and Mechanics, Volume 9, Issue 1, pp. 157–172, 2017.

Eigenständigkeitserklärung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel angefertig habe. Alle Stellen der Arbeit, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht.

.....	.....
Datum	Unterschrift