Application of an Algorithmically Differentiated Turbomachinery Flow Solver to the Optimization of a Fan Stage

Jan Backhaus*, Andreas Schmitz*, Christian Frey†
Institute of Propulsion Technology, German Aerospace Center (DLR)
Sebastian Mann‡, Marc Nagel§
MTU Aero Engines AG
Max Sagebaum¶, Nicolas R. Gauger‖
Chair for Scientific Computing, TU Kaiserslautern

The adjoint method has already proven its potential to reduce the computational effort for optimizations of turbomachinery components based on flow simulations. However, the transfer of the adjoint-based optimization methods to industrial design problems turns out to pose specific requirements to both the adjoint solver as well as the optimization algorithms which utilize the gradient information. While the construction of the adjoint solver through algorithmic differentiation is described in a parallel publication, we focus here on the robust application of the gradient information in a high-dimensional multi-objective optimization with several constraints including non-differentiated mechanical constraints. We describe the optimization methods, which comprise the use of gradient-enhanced Kriging meta-models, and subsequently apply these to the design optimization of a contra-rotating fan stage. The results show that through the described combination of methods the adjoint method can be used in practical design optimizations of turbomachinery components.

I. Introduction

The aerodynamic design of turbomachinery components relies to a large extent on computational fluid dynamics. Due to the steadily increasing environmental and economical demands and strict safety requirements, design problems can become quite complex. Therefore, optimization techniques which help the designer to explore large design spaces, play an increasingly important role. Typically, optimizations are performed using gradient-free techniques, e.g. evolutionary algorithms, since the deployed flow solvers do not provide gradient information of their output quantities. However, the number of necessary design evaluations increases exponentially with the number of design variables, a phenomenon which is referred to as the curse of dimensionality.

The adjoint method can alleviate the limitation of the design space size by providing gradient information at costs which are independent of the number of design variables. The large number of publications about adjoint flow solvers provides evidence of the interest in this method.[1,2,3,4,5,6] Over the last two decades adjoint-based optimization methods have been successfully used for simple airfoil designs. However, as the limited number of publication shows, the transfer of the adjoint-based optimization methods to industrial design problems turns out to be less straightforward. To improve the applicability of the adjoint method for realistic designs, further efforts are required to improve both the adjoint solvers and the optimization methods.

*Phd. candidate, Institute of Propulsion Technology, German Aerospace Center (DLR), Cologne
†Research Associate, Institute of Propulsion Technology, German Aerospace Center (DLR), Cologne
‡Phd. candidate, MTU Aero Engines AG, Munich
§Research Associate, MTU Aero Engines AG, Munich
¶Phd. candidate, Chair for Scientific Computing, TU Kaiserslautern
‖Professor, Chair for Scientific Computing, TU Kaiserslautern

This publication describes an approach for gradient-enhanced meta-model assisted optimization of turbomachinery, demonstrated on a contra-rotating fan stage with special focus on requirements from test-rig design practice. A parallel paper[7] together with an earlier publication[8] show how reverse mode differentiation play together with elaborate performance optimizations to construct and maintain a consistent and efficient adjoint solver from a simulation code under frequent development. This paper mainly focuses on the efficient utilization of the gradient information. Here, we propose the use of gradient-enhanced meta-modeling and discuss the specific adaptation of this technique.

The paper is organized as follows: In section II we describe the specifics of the optimizations that influenced our choice of methods, followed by a general description of optimizations using gradient-enhanced meta-modeling and the specific implementation considerations. Subsequently we outline the choice of methods for the flow simulation and summarize the development techniques for the discrete adjoint. This is followed by the application of these methods to the design optimization of a contra-rotating fan stage in section III.

II. Method

A. Choice of Methods

For an optimization, two components are needed: A design evaluation process and an optimizer. Both components are typically loosely coupled: The optimizer provides a choice of values for the design variables $\mathbf{x}$ to the evaluation process, which calculates the objective and constraint values $\mathbf{f}$. The gradients $\frac{\partial f_i}{\partial x_j}$ are returned for all pairs i, j for which a derivative of f_i with respect to x_j is computed inside the evaluation process.

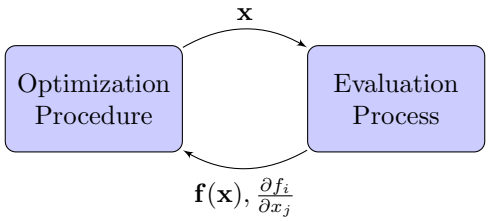


Figure 1: Optimization scheme

The choice of the optimization method depends on the specifics of optimization problem as well as the evaluation process. The most important topics are in this case:

1. Multiple objectives and constraints from different disciplines
2. Design parameterizations which often include third party software (CAD-Kernels etc.)
3. Objectives or constraints for which no derivative can be computed, either due to the lack of a differentiated tool or non-differentiability
4. Number of design variables of at least 80 per stage
5. Computationally expensive design evaluations (hours of computational time)
6. Demand for running multiple concurrent evaluations
7. An evaluation process that may fail to provide values or gradients for some of the objectives or constraints at any point of the design space.
8. Non-convex objective functions, with multiple local minima, e.g. caused by complex flow phenomena

The most popular algorithms to exploit gradient information stem from the class of gradient descent methods and are quite effective. However item 3, 7 and 8 are, from our experience, strong arguments against the use of gradient descent methods in this setting.

gradient-enhanced Kriging (GEK) allows building a meta-model even when incomplete information is present at sampling points and provides a very flexible model which can adapt to a large range of functions. This flexibility comes for the price of computational effort to build a GEK model. Therefore we apply acceleration techniques to the training procedure as described in a following section.

B. Gradient assisted meta-modeling

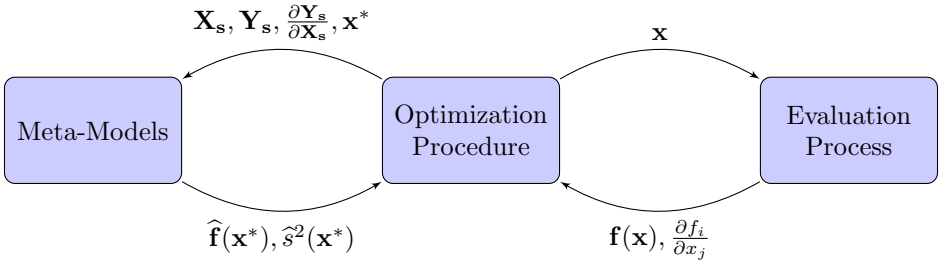


Figure 2: Optimization with a gradient-enhanced meta-model

The basic information flow of gradient-enhanced meta-modeling is depicted in Fig. 2. The optimization procedure triggers the evaluation process for one choice of design variables $\mathbf{x}$ and obtains function and gradient values in return. However, the choice of where to evaluate the model is now determined by predictions from the meta-models. These models are built from already sampled values $\mathbf{Y_s}$ at the locations $\mathbf{X_s}$, and the corresponding gradient information $\frac{\partial \mathbf{Y_s}}{\partial \mathbf{X_s}}$ where available. Once built, the meta-models can inexpensively be queried for function predictions $\widehat{\mathbf{f}}$ at unsampled points $\mathbf{x}^*$. Additionally, the meta-models considered here also return an estimated standard deviation of their prediction. This measure is useful for balancing the exploration of regions with missing samples and the exploitation of expected local minima. By optimizing on the meta-model, the optimization procedure obtains a new point which can then be fed into the evaluation process.

The following section will describe how gradient-enhanced Kriging models are constructed and evaluated in this work. For a more thorough description of the DGEK, the reader is referred to the cited publications.[9,10,11]

C. Gradient-enhanced Kriging

In this section, we will first describe the gradient-free Kriging approach and then describe the extension to direct gradient-enhanced Kriging. The Kriging method models an unknown function $f(\mathbf{x})$ as the realization of a random variable Y by fitting a correlation structure of this variable to given observations. The model is constructed based on a set of observations at sample points $(\mathbf{x_i}, f(\mathbf{x_i})), i \in \{1, ..., D\}$, where $\mathbf{x_i} \in \mathbb{R}^N$. The vector of observations $\mathbf{Y_s}$ is defined as $\mathbf{Y_s^T} = [f(\mathbf{x_1}), ..., f(\mathbf{x_D})] = [y_1, ..., y_D]$.

Kriging predictions assume a spatial correlation between the observed samples in order to interpolate the sampled values. This assumption is expressed through the choice of a correlation function which only depends on the weighted distance between two points. Gaussian or cubic spline functions are the most commonly used forms of the spatial correlation function. In this work we use the Gaussian correlation function which is defined as

$$\mathrm{Corr}(\mathbf{x_i}, \mathbf{x_j}) = \prod_{k=1}^{N} \mathrm{Corr}_k(\theta_k, \mathbf{x_i}, \mathbf{x_j}) = \exp\left(-\sum_{k=1}^{N} \theta_k (x_{i,k} - x_{j,k})^2\right). \tag{1}$$

The parameters of the correlation function $\theta_k, k \in \{1, ..., N\}$, the hyper-parameters, are used to adjust the model to the observed data. This procedure is called training and is described later. If we assume the parameters are already determined, the Kriging model can be evaluated as follows. The correlations between all samples are arranged in a covariance matrix $\mathrm{Cov}_{i,j} = \sigma^2 \mathrm{Corr}(\mathbf{x_i}, \mathbf{x_j}) \in \mathbb{R}^{D \times D}$, where σ denotes the standard deviation of the Kriging process. With this matrix the ordinary Kriging predictor $\widehat{y}$ and the

variance of the prediction of any point $\mathbf{x}$ in the design space can be obtained by

$$\widehat{y}(\mathbf{x}) = \beta + \mathbf{c}^T(\mathbf{x})\mathbf{Cov}^{-1}(\mathbf{Y_s} - \mathbf{F}), \tag{2}$$

$$\widehat{s}^2(\mathbf{x}) = \left(\sigma^2 - \mathbf{c}^T(\mathbf{x})\mathbf{Cov}^{-1}\mathbf{c}(\mathbf{x}) + \frac{(\mathbf{c}^T(\mathbf{x})\mathbf{Cov}^{-1}\mathbf{F} - \beta)^2}{\mathbf{F}^T\mathbf{Cov}^{-1}\mathbf{F}}\right), \tag{3}$$

with

$$\begin{aligned}
\mathbf{c}_i(\mathbf{x}) &= \mathrm{Cov}(\mathbf{x}, \mathbf{x_i}),\\
\mathbf{F}^T &= [\beta, ..., \beta] \in \mathbb{R}^D,\\
\beta &= \frac{\mathbf{G}^T\mathbf{Cov}^{-1}\mathbf{Y_s}}{\mathbf{G^T}\mathbf{Cov}^{-1}\mathbf{G}},\\
\mathbf{G}^T &= [1, ..., 1] \in \mathbb{R}^D.
\end{aligned}$$

Gradients can already be incorporated in these models, without any change to the formulation, by transforming gradients into artificial new samples and including these in the covariance matrix. This approach is called indirect gradient-enhanced Kriging. We follow a different approach, where the gradient information is directly included in the correlation matrix by introducing new correlation functions. These correlate function values with gradients and gradients with gradients for any pair of sampling locations. These additional correlations are obtained by differentiating the covariance function. The extended covariance matrix then reads

$$\widetilde{\mathbf{Cov}} = \left(\begin{array}{cc}
\mathrm{Cov}(x_i, x_j) & \dfrac{\partial \mathrm{Cov}(\mathbf{x}_i, \mathbf{x}_l)}{\partial \mathbf{x}_l^n}\\
\underbrace{\dfrac{\partial \mathrm{Cov}(\mathbf{x}_l, \mathbf{x}_i)}{\partial \mathbf{x}_l^n}}_{\text{D entries}} & \underbrace{\dfrac{\partial^2 \mathrm{Cov}(\mathbf{x}_j, \mathbf{x}_l)}{\partial \mathbf{x}_l^m \partial \mathbf{x}_l^n}}_{\text{P entries}}
\end{array}\right) \begin{array}{l}\Big\} \text{D entries}\\ \Big\} \text{P entries}\end{array} \tag{4}$$

where $k, l \in \{1, ..., D\}$ and $n, m \in \{1, ..., N\}$ and P denotes the total number of observed derivatives. If partial derivatives for each design parameter at all samples are provided then $P = D \cdot N$ and the matrix has the dimension of $D(1+N) \times D(1+N)$. This extended correlation matrix can be used in place of the point-point correlation matrix defined above. Therefore, we do not distinguish between both matrices. With this extended correlation matrix the extended vectors

$$\begin{aligned}
\mathbf{Y_s}^T &= [y_1, ..., y_D, \frac{\partial y_1}{\partial x_1}, ..., \frac{\partial y_D}{\partial x_n}],\\
\mathbf{G}^T &= [\underbrace{1, ..., 1}_{\text{D entries}}, \underbrace{0, ...0}_{\text{P entries}}],\\
\mathbf{F}^T &= [\underbrace{\beta, ..., \beta}_{\text{D entries}}, \underbrace{0, ...0}_{\text{P entries}}],\\
\mathbf{c}_i(\mathbf{x})^T &= [\mathrm{Cov}(\mathbf{x_1}, \mathbf{x}), ..., \mathrm{Cov}(\mathbf{x_D}, \mathbf{x}),\\
&\quad \frac{\partial \mathrm{Cov}(\mathbf{x_1}, \mathbf{x})}{\partial x_1^1}, ..., \frac{\partial \mathrm{Cov}(\mathbf{x_D}, \mathbf{x})}{\partial x_D^n}],
\end{aligned}$$

the GEK predictor $\widehat{y}$ and the GEK uncertainty $\widehat{s}$ of the prediction for any point can still be calculated by equation (2) and (3). This direct gradient-enhanced Kriging (DGEK) approach is superior to the indirect approach in terms of stability and accuracy of the model.[12]

D. Training

For the training of the hyper-parameters θ_k, the reduced maximum-likelihood approach is used, which aims at finding the most likely model which produces the observed samples values and derivatives. This is achieved by minimizing the negative log-likelihood function with respect to the hyper-parameters θ_k:

$$\ell = -\ln\left(\det\left(\mathbf{Cov}\right)\right) - (\mathbf{Y_s} - \mathbf{F})^T\mathbf{Cov}^{-1}(\mathbf{Y_s} - \mathbf{F}). \tag{5}$$

Usually the variation of the data, σ, is computed analytically. However, as this is subject to a large sampling error, we include σ as a hyper-parameter in the likelihood minimization iteration, which has proven to lead to better approximations.

To minimize the reduced likelihood, we employ the gradient descent method from the resilient back propagation algorithm,[13] which is a common algorithm in the training of artificial neural networks. The necessary partial derivatives of (5) with respect to θ_k are calculated as

$$\frac{\partial \ell}{\partial \theta_k} = -\mathrm{Tr}\left(\mathbf{Cov}^{-1}\frac{\partial \mathbf{Cov}}{\partial \theta_\mathrm{k}}\right) + (\mathbf{Y_s} - \mathbf{F})^T \mathbf{Cov}^{-1}\frac{\partial \mathbf{Cov}}{\partial \theta_k}\mathbf{Cov}^{-1}(\mathbf{Y_s} - \mathbf{F})$$

To avoid the constraint $\theta_k > 0$ in the maximum likelihood optimization and obtain a better scaling of values, we define $\theta_k := e^{\widehat{\theta}_k}$ and use unconstrained minimization to find an optimal $\widehat{\theta}_k$.

E. Optimization on the Meta-Model

The next point to be evaluated is obtained by optimizing on the meta-model. Obtaining predictions from a trained meta-model is comparatively cheap. While efficiency is not the main concern for the choice of optimization method, other aspects like simplicity and robustness are more important. Stochastic methods have the advantage of producing different predictions, when repeatedly run on the same model, which is advantageous when the evaluation chain fails to deliver function values for a prediction and therefore no model update takes place. Therefore we chose to use evolutionary algorithms to optimize on the meta-model. This optimization has to fulfill two requirements:

- Search regions of optimality for the best solution (exploitation)
- Reduce uncertainty about unsampled regions since they might also contain regions of optimality (exploration)

It can be shown that an optimization, based only on the predicted values $\widehat{y}$ would mainly focus on exploitation and might converge to non-optimal solutions (cf. section 3.2.1 in Forrester et al.[14]). This has to be considered by formulating a dedicated objective function for the optimization on the meta-model. For single-objective optimizations Jones et al.[15] suggest the expected improvement criterion: The objective is to maximize the expected value of the improvement over the currently best value.

In the case of multi-objective optimizations, this criterion must be extended, since there is no single best solution but a set of equally optimal solutions called Pareto front. A member, represented by its objective values $\mathbf{a} = (a^{(1)}, ..., a^{(M)})$ dominates another member $\mathbf{b} = (b^{(1)}, ..., b^{(M)})$ if $a^{(i)} \geq b^{(i)}$ for all $i = 1, 2, ..., M$, and $a^{(i)} > b^{(i)}$ for some i. Furthermore, members that violate at least one constraint are dominated by members that fulfill all constraints. A solution which is not dominated by any other solution is called Pareto-optimal and the set of Pareto-optimal solutions is called Pareto front.

The gain from a new sample can be calculated as the volume gain by calculating the volume which is enclosed between the Pareto front Y and the new sample x:

$$\prod_i \left(x^{(i)} - \max\{y^{(i)}, y \in Y | y^{(i)} < x^{(i)}\}\right). \tag{6}$$

If no sample in the Pareto front satisfies this criterion, the difference to a lower limit for the objective is used instead. In order to turn this criterion into a statistical prediction, the expected volume gain, we use mean and variance from a location point in the meta-model to describe a multi-dimensional normal distribution. By Monte-Carlo-Sampling of this distribution we obtain samples around the selected location to calculate statistics of the above formula[a].

This objective can be minimized in two ways: Either by a prediction $\widehat{y}$ with a lower mean than, or one where the expected value is worse than the current optimum, but with a high predicted variance $\widehat{s}^2$. In the second case the high variance means, that the real observation can deviate from the predicted mean and there is a chance that it could undercut the current optimum. The first mechanism will prefer exploitation. The second accounts for exploration of the model, since it will prefer samples in regions with high variance, which is caused by uncertainty over the real values at this point. Imposing a constraint on the variation of a prediction during the optimization on the meta-model allows shifting the balance from exploration to exploitation.

[a] In order to predict multiple new members from one meta-model, the joint distribution of N models is built

F. Kriging regularization

A main difficulty, especially in gradient-enhanced Kriging, is that the covariance matrix easily becomes ill-conditioned. The condition number can even reach magnitudes where the Cholesky algorithm fails to provide a factorization of the matrix. In gradient.free Kriging this is often caused by highly correlated samples, e.g. sampling points becoming too close. Therefore a strategy that avoids placing sampling points close to already known points would be optimal. However this cannot be assured during the optimization, since sampling points will naturally become clustered in regions of optimality. A well-known solution to this problem is to introduce a Tikhonov regularization - usually by adding a small constant to the main diagonal of the covariance matrix. In gradient-free Kriging an addition to the main diagonal corresponds to the assumption of random noise in the sampled values. This regularization eliminates the interpolation property of the Kriging method: already known points are no longer reproduced, but only approximated. Through larger regularization the model increasingly becomes a regression model. A very small regularization is often beneficial, as simulation results are prone to a small random error since convergence to machine accuracy is never achieved in practice.

It can be observed that gradient-enhanced Kriging tends to produce much larger condition numbers than gradient-free Kriging. The usual regularization with a constant addition to the main diagonal could be applied to gradient-enhanced Kriging but would impair the prediction quality of the function values. For this reason and, since it is the additional gradient information that causes the ill-conditioning, we employ a slightly modified regularization

$$\widetilde{\mathbf{Cov}} + \Gamma$$

where

$$\Gamma = \mathrm{diag}(\underbrace{\delta \quad \delta \quad \cdots \quad \delta}_{D \text{ times}} \quad \underbrace{\gamma \quad \gamma \quad \cdots \quad \gamma}_{P \text{ times}}).$$

Two constants are used in this regularization scheme: $\delta > 0$ is the regularization constant for the point-point correlations and $\gamma > 0$ for the gradient-gradient part of the matrix. Since both $\widetilde{\mathbf{Cov}}$ and Γ are positive definite matrices this corresponds to a regularization of $\widetilde{\mathbf{Cov}}$. Note that the representation accuracy of value information can be adjusted independently (e.g. using experience from gradient-free Kriging) from the gradient representation accuracy. This is a desirable property since we do not use predictions of gradients but only use the gradients to improve the prediction of values between samples.

Other authors propose the selective removal of correlations from the matrix which contribute the least additional information (e.g. pivoted Cholesky decomposition is used by Dalbey[16]) to improve the Kriging condition number. Since the Kriging meta-model does not require full information on each sampling point, we also propose not to calculate the complete gradient information at all sampling points. In the application section we demonstrate a simple criterion based on the number of evaluated points.

G. Flow simulations

Aerodynamic objectives are evaluated using DLR's turbomachinery flow solver TRACE (for an overview cf.[17,18]). The methods used in this work are exemplary for optimization simulations performed with this solver. We calculate the steady state solution to the compressible Reynolds-averaged Navier-Stokes equations in a rotating frame of reference using the ideal gas assumption. Turbulence is modelled by the Wilcox k-ω two equation turbulence model[19] with modifications to correct the stagnation point anomaly[20] and to account for rotational effects.[21] The spatial discretization is based on a hybrid structured/unstructured finite volume approach; although completely structured grids have been used in this work. Convective fluxes are discretized using the MUSCL upwind approach with second-order accuracy and Harten's entropy fix. A van Albada-type limiter is used to prevent oscillations in the vicinity of shocks (for the effect of flux limiters on differentiability cf.[22]). The wall boundary layer is modelled using wall functions. The solution is obtained by implicit pseudo-time marching. Adjacent blade rows are coupled by Denton's mixing plane[23] approach. Non-reflecting boundary conditions[24] are used to avoid unphysical reflections of waves at inlets, outlets and blade row interfaces. For a validation of the solver with experimental data for a contra-rotating fan cf. Lengyel-Kampmann et al.[25] .

H. Gradient Calculation

We calculate gradients of the aerodynamic objective functions through the discrete adjoint approach.[26] For the implementation of the discrete adjoint solver to the existing CFD code we use the reverse mode of algorithmic differentiation (AD).[27] However, an adjoint solver which is built by simply applying reverse-mode AD to the complete primal code leads to infeasibly high resource demands. For this reason, algorithmic differentiation is frequently combined with the more efficient manual differentiation. Many authors use manual differentiation on the top level, e.g. for the time marching scheme and the spatial stencil, and use algorithmic differentiation underneath for flux routines etc.[28, 29, 3] A recent study describes how this technique can be applied to adjoin the flow solver under consideration.[30] The third author originally developed an approximate discrete adjoint through a combination of manual differentiation and finite differencing.[31]

However, experience shows that an approach which is driven by manual implementation becomes difficult to maintain. The continuing development on the primal code requires constant effort for updating or adding adjoint models. Since manual adjoining of involved CFD modules is a sophisticated task, there is at least a lag between the availability of a primal simulation model and its adjoint - or primal models which are completely omitted. In case primal models with an existing adjoint are modified, the development lag can lead to inconsistent implementations and therefore wrong derivatives.[32]

Consequently we decided to use algorithmic differentiation to differentiate the whole code in a black box fashion, which yields a complete and consistent, yet inefficient, adjoint solver and subsequently apply manual performance optimizations through annotations in the primal code[8].[7] To describe these optimizations we regard the primal flow solver (including post processing) as a function of the mesh coordinates $\mathbf{x} \in \mathbb{R}^n$ which delivers the vector of desired objective functions $\mathbf{I} \in \mathbb{R}^m$:

$$\mathbf{I} = \mathbf{f}(\mathbf{x}).$$

Applying reverse mode differentiation to such a function allows to compute products of the function's Jacobi matrix with a constant vector,

$$\bar{\mathbf{I}} \frac{\partial \mathbf{f}}{\partial \mathbf{x}}.$$

Here, the adjoint seed $\bar{\mathbf{I}} \in \mathbb{R}^m$ is usually chosen to be one output quantity

$$\bar{\mathbf{I}}_j = \frac{\partial \mathbf{I}}{\partial \mathbf{I}_j} = \mathbf{e}_j$$

with $\mathbf{e}_j$ being the j-th unit vector. This is achieved by regarding the program as a chained sequence of primitive mathematical operations $\varphi_i \in \{+, -, *, /, \sin, \cos, \exp, ...\}, \quad i = 1, .., l$. We first calculate the results of all primal operations

$$v_i = \varphi_i(v_j)_{j \prec i}.$$

Where $(v_j)_{j \prec i}$ is the list of variables on which the i-th operation depends. Afterwards we propagate the derivatives backward through the calculations in a reverse sweep:

$$\bar{v}_j = \sum_{i \prec j} \bar{v}_i \frac{\partial}{\partial v_j} \varphi_i(v_j)_{j \prec i} \quad j = l, ..., 1 \quad .$$

The summation runs over all i which depend on the result of the j-th operation.

From this relation, it can be seen that $\bar{v}_j$ must be calculated in reverse order. However, we need the values of the primal variables $(v_j)_{j \prec i}$ which can only be calculated in forward order. This is typically resolved by storing all operations and intermediate results in the primal sweep in a sequential data structure called tape. Nevertheless, this approach requires enough storage to accomodate this tape. Even though memory can be traded in for run-time through checkpointing,[33] the performance requirements are regarded too large for practical simulations. Therefore performance improvements are necessary. We will briefly repeat the key points here, the techniques are described in detail in the parallel paper.[7]

Finding the steady state solution to the RANS equations is expressed by a flow state with a vanishing residual:

$$R(u) = 0$$

The implicit pseudo time marching scheme used here can be written as

$$u^{n+1} = u^n + P(u^n) \cdot \Delta u. \tag{7}$$

Where P is an approximation to the Jacobian matrix of the residual $\frac{\partial R}{\partial u}\big|_{u^n}$ and the state update Δu is determined by solving the implicit system of equations

$$\left(\Delta t^{-1} + P(u^n)\right) \Delta u = R^n. \tag{8}$$

We can regard this as an iterative procedure which produces a new flow state with each iteration

$$u^{n+1} = G(x, u^n),$$

where G comprises the calculation of the residual and advancing the flow state. Note that the extraction of the iteration G and the identification of u as well as accounting for indirect dependence of G on x inside a real solver may be quite involved.

Simply differentiating the whole iteration loop over G by reverse mode AD would require recording all operations and intermediate results which is infeasibly expensive. Assuming the iteration G converges to an attractive fixed point

$$u^* = G(x, u^*)$$

where

$$R(u^*) = 0,$$

it is sufficient to only record one G iteration on this fixed point and generate an adjoint iteration by repeatedly propagating the adjoint state $\bar{u}$ through this tape. A theorem of Christianson[34] states, given that the primal iteration has converged to machine accuracy, the adjoint iteration procedure approaches the same convergence rate as the primal iteration.

About half of the tape size and computational time of the adjoint recurrence can be attributed to the solution of the implicit system of equations. The implicit matrix $\left(\Delta t^{-1} + P(u^n)\right)$ acts as a preconditioner to the primal iteration procedure, which means that the flow solution will not depend on it - only the convergence rate. Provided u^n is sufficiently close to fixed point, $P(u^n)$ does not have to be recomputed for the primal solver to converge. Consequently the dependencies of the preconditioner can be ignored during the recording of G. For the present solver, this reduces the memory consumption and run-time of the adjoint solver by a factor of two while leaving the convergence rate and results practically unaffected.

While both methods are quite effective in reducing the tape size, their prerequisite of $R(u^*) = 0$ is a theoretical one. Convergence to machine accuracy is never reached due to the stopping criteria of the solver and rounding errors. Some more complex simulations even stall at a larger residual due to the occurrence of periodic flow phenomena which cause the flow state only to converge to a limit cycle oscillation. Even when those primal solutions cannot be called converged, it can be observed that the integral quantities of interest converge sufficiently accurate for practical purposes. While for a lot of these cases the adjoint recurrence still converges, there is no guarantee that it will do so. Stabilization of adjoints for oscillating primal solutions is an actively researched topic.[35,36]

The third measure for reducing the tape size is based on the observation that for some functions it is cheaper to store the complete Jacobian matrix than to store the computational tape. Obviously this is true for functions with few input and output arguments but which comprise many internal operations. These functions are manually identified, must be free of side effects regarding the state vector and their input and output arguments have to be annotated. Then we can automatically calculate and store their Jacobi matrix while recording of the primal iteration, instead of their usual tape representation.

The code of the flow solver under consideration is developed in the C programming language, however after some modifications[8] the code now also complies to the C++11 standard, which is sustained by automatic testing. This allows us to use reverse mode differentiation through operator overloading (OO) by using dco/c++.[37] For validation purposes we also use the reverse sweep of the full convergence trajectory with checkpointing as well as the tangent forward mode.

III. Application

The application example is taken from a design optimization[38] for a contra-rotating turbofan stage called CRISP 2, which has recently been conducted at DLR. Starting from a test-rig designed and tested in the 1990s, an extensive multidisciplinary optimization has been carried out to explore the potential of contra-rotating fan stages under the use of modern design, flow analysis and manufacturing techniques, especially

the use of CFRP-compound materials.[39,40] CRISP 2 is a shrouded fan stage test-rig with two contra-rotating rows, equipped with 10 and 12 blades with 1 meter in diameter at inflow. The final design has been built and is about to be experimentally tested at DLR in the near future.

The original study was conducted using gradient-free meta-modelling in a chain of successive optimizations. In order to use this as a benchmark case for this publication, we take one of the intermediate results from the later design cycles and optimize it using typical design objectives for a counter rotating fan.

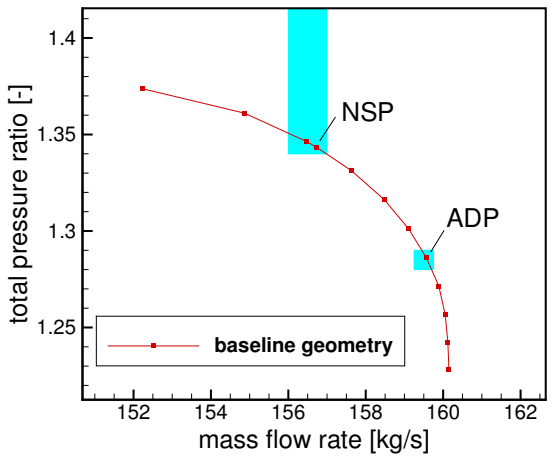


Figure 3: Working line of the baseline member. (Blue areas depict aerodynamic constraints)

Role	Name	Symbol	objective/constraint
1. Objective	isentropic efficiency at ADP	$\eta_{\text{is,ADP}}$	increase/ $[0.75, \infty[$
2. Objective	total pressure ratio at NSP	$\pi_{\text{tot,NSP}}$	increase/ $[1.34, \infty[$
1. Constraint	mass flow rate at ADP	$\dot{m}_{\text{ADP}}$	$[159.25, 159.75]$ kg/s
2. Constraint	total pressure ratio at ADP	$\pi_{\text{tot,ADP}}$	$[1.28, 1.29]$
3. Constraint	mass flow rate at NSP	$\dot{m}_{\text{NSP}}$	$[156, 157]$ kg/s
4. Constraint	maximum Von Mises stress rotor 1	$\sigma_{v,R1}$	$[-\infty, 400]$ MPa
5. Constraint	maximum Von Mises stress rotor 2	$\sigma_{v,R2}$	$[-\infty, 400]$ MPa

Table 1: Objectives and constraints

The baseline design is designed for a freestream velocity around Mach 0.68 with transonic flow (around Mach 1.2) in the relative frame of reference. The stage produces a mass flow rate of 159 kg/s at a total pressure ratio around 1.29. The main objective will be to improve the fan stage's isentropic efficiency η_{is} at the aerodynamic design point (ADP). The ADP is fixed during the optimization by restricting the mass flow rate to a range of 0.5 kg/s and the total pressure ratio to a range of 0.01 around the baseline. A typical trade off in turbomachinery design is that efficiency gains in the ADP reduce the stable operation range. Since the real operation range is hard to predict numerically, we use the stall margin criterion proposed by Cumpsty which relates the operation range to the increase of total pressure ratio towards the surge line. In order to calculate this criterion we calculate a second operating point, the near stall point (NSP), which is defined by a larger back pressure and has a 2.5% lower mass flow rate. A second objective function demands that the total pressure ratio in the NSP should be increased. The NSP is also restricted in its mass flow rate with a tolerance margin of 1 kg/s and a lower bound to the total pressure ratio of 1.34. The operating points and their corresponding restrictions are depicted on the working line of the baseline geometry in Fig. 3.

A. Parameterization

The design is parameterized by a parametric CAD model based on 81 engineering parameters, influencing both the blade and the flow path shape. A summary of the used parameters is given in Table 2.

number of parameters		description
flow path	3	control point hub curve axial-shift
	5	control point hub curve radial shift
	5	control point casing curve radial shift
rotor 1 & 2	2	radial position of construction profile 2
	6	stagger angles
	12	angles at the leading edge
	12	angles at the trailing edge
	12	ellipse semi-axes at the leading edge
	6	radii at the trailing edge
	6	profile chord lengths
	12	spline control point positions on the suction side

Table 2: Summary of the parameters

B. Mechanical modelling

Mechanical stability is a crucial restriction in design optimizations when the result is to be built and tested. Stability is usually assessed by mechanical calculations based on the FEM method, which are regularly performed using third party tools without a corresponding adjoint solver. Furthermore, some meaningful mechanical constraints, like maximum stresses, are not continuously differentiable. In case the mechanical constraints can be evaluated sufficiently faster than the aerodynamic quantities, gradient-free mechanical constraints can still be regarded together with gradient-based aerodynamic objectives. Each constraint or objective is modelled by its own meta-model and the position and number of observations from which the model is built can differ. Therefore we create models for the mechanical constraints based on much more samples than the aerodynamic objectives. In the described optimization we apply this technique by including static calculations for the maximum von Mises stresses inside the blade. Even if an adjoint to the FEM tool were available, this exact constraint would not be differentiable due to the maximum operator. From a design of experiments containing 800 static FEM simulations we build a Kriging model, which is then used by the optimizer to obtain predictions. The mechanical constraints are then evaluated for newly proposed members together with the aerodynamic objectives and are used to update the mechanical meta-model.

C. Practical training of the Kriging models

An efficient implementation of the gradient-enhanced Kriging model is essential, as the training procedure can otherwise consume the computational saving from the adjoint gradients. The implementation inside the Optimization framework AutoOpti is described by Schmitz;[41] the key points are summarized in the following paragraphs.

The main issue is that the Kriging training incorporates operations on large dense matrices, which have a computational complexity of $\mathcal{O}(n^3)$. The evaluation of the log-likelihood makes it necessary to determine the inverse of the covariance matrix. The time spent on the inversion can be largely reduced by the use of computational accelerators, more specifically general purpose graphics processing units (GPGPU). Furthermore highly tuned implementation of the log-likelihood and hand coded partial derivatives are used for the optimization.

In principle, the Kriging hyper-parameters have to be re-adjusted with every new infill from a design evaluation. However, often the information from an additional evaluation is in accordance with the current choice of model parameters, which can be easily determined by including the new point and evaluating the likelihood with the current set of hyper-parameters. In this case the training procedure can be skipped, which considerably reduces the total costs for Kriging training throughout the optimization.

The largest training matrix obtained during the optimizations for this paper is approximately 4700×4700 in size. When the meta-models have to be re-trained at this state, the total training time for all models (~ 12 minutes accelerated by an NVidia Quadro-K6000) is still an order of magnitude smaller than the function and gradient evaluation (~ 5 hours on an Intel Xeon E5-2650).

D. Design evaluation

The design evaluation is performed by a process chain in which all the values and gradients are calculated. This process chain consists, on a high level, of the following steps:

- Generation of solid surfaces
- Check for geometric constraints
- Generation of 3D meshes (FV and FEM)
- Finite element simulations
- RANS simulations
- RANS post-processing

First, the values of the design variables are inserted into a parametric CAD model to create spline surface representations of all solid walls. Subsequently these surfaces are checked if they satisfy a set of geometric properties which are necessary for manufacturing and mechanical robustness. From these surfaces, 3D meshes are generated. This process up to this point is implemented in an in-house tool chain for turbomachinery design, belonging to the optimization suite AutoOpti. For the calculation of the mechanical constraints we use the FEM-Solver CalculiX.[42] CFD Simulations are performed using the Navier-Stokes solver and corresponding postprocessor from DLR's turbomachinery simulation suite TRACE.[18,43] After the simulations and their post-processing are finished, values for all objective functions and constraints are available.

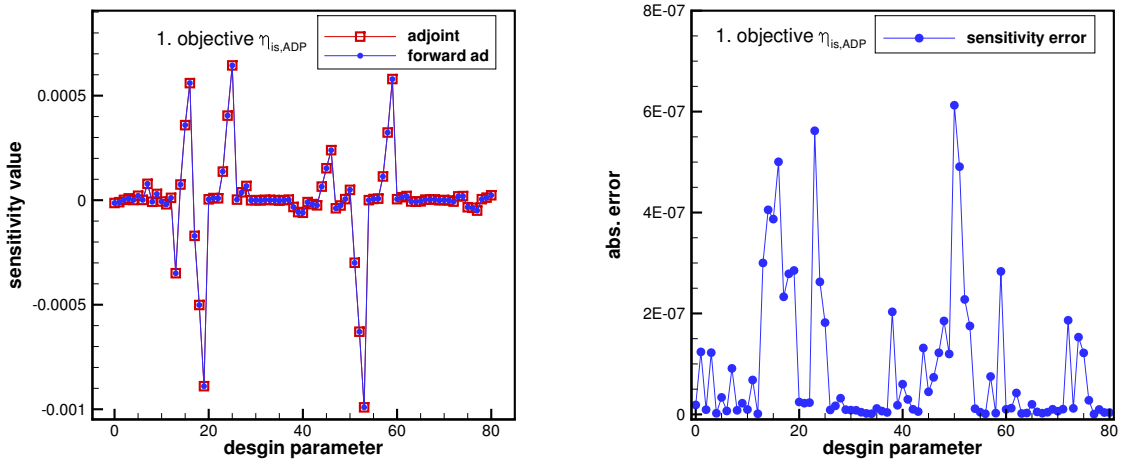
Derivatives of CFD results are calculated by the adjoint method with the following process:

- Adjoint post-processing
- Adjoint RANS simulations
- Finite differences of surface and mesh generation
- Scalar product of perturbed meshes and adjoint solutions

The generation of solid surfaces and the meshes for simulation usually involves third party tools for which no differentiated counterparts are available. Therefore, we apply small perturbations to one parameter at a time and re-run the whole process up to and including the mesh generation. This produces one perturbed mesh per parameter. These meshes can then be combined with each mesh sensitivity result from the adjoint process by means of a dot product. This can be interpreted as the projection of the mesh sensitivity on a mesh deformation obtained by finite differencing of the pre-process. This is based on the assumption, that the dependence of mesh nodes on the design parameters can be approximated by a linear relation in a certain range and that the choice of step-width is much easier for the pre-process.

E. Gradient Validation

It is crucial for the optimization, that the adjoint gradients are consistent with the primal solver. Experience shows that otherwise the optimization progress is slowed down up to a point where a gradient-free optimization advances faster. Inconsistent gradients also tend to worsen the ill-conditioning of the Kriging correlation matrix. To validate the gradient information, we employ calculations using the forward mode of AD, as this mode is most simple to implement and therefore least error prone. Applying finite differences would be as simple; however the choice of step size is a very cumbersome task when fluid simulations are involved. Nevertheless, finite differences served as a tool for plausibility checks during the implementation of the full forward mode. Since the forward mode validation requires running 81 primal calculations, we



(a) Comparison of sensitvity values

(b) Absolute error of the sensitivities

Figure 4: Validation of sensitivities for objective 1 at the baseline design by comparing forward AD and adjoint

chose to limit the validation to the ADP of the baseline geometry. Results can be seen in Fig. 4, where the results from the reverse mode differentiation including all performance optimizations are compared to a forward mode differentiation where no such optimizations are done and everything is consistently differentiated. Comparison of the gradient values (Fig. 4a) shows no visual difference, while in (Fig. 4b) it can be seen that a difference exists, but it is smaller than $6 \cdot 10^{-7}$. This difference is believed to be due to the stopping criteria of both the adjoint and forward mode differentiated solvers.

An impression how the convergence rate of the adjoint solver, which is itself linked to the primal solver residual by Christianson's theorem, relates to the evolution of the sensitivity value is given in Fig. 5.

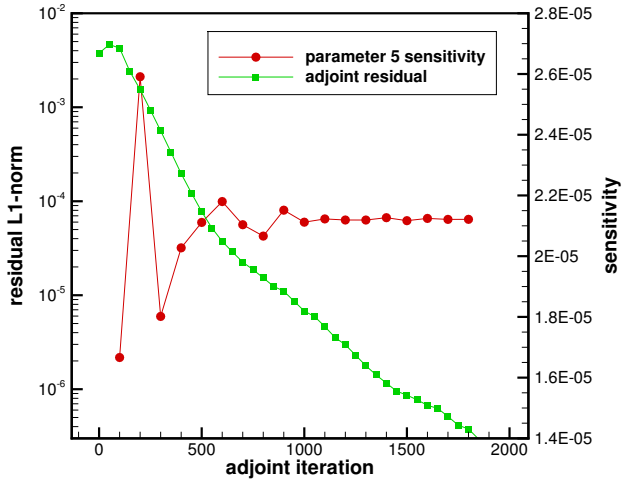


Figure 5: Convergence of the sensitivity value of parameter 5 in relation to the adjoint residual L1-norm

F. Optimizations

The main purpose of the application part of this paper is to demonstrate that adjoint simulations can be employed for optimizations in a realistic setting. The second aim is to demonstrate, that the gradient information significantly improves the prediction quality of the meta-models and therefore improves the optimization.

In order to demonstrate this, three optimizations have been performed:

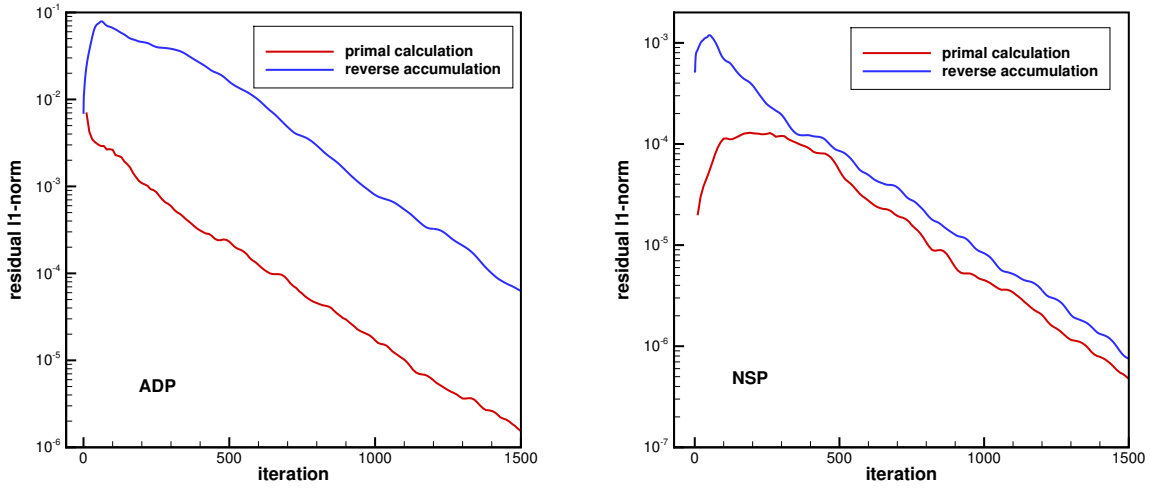
1. without gradient information
2. with adjoint calculations for all sample points (fully gradient-enhanced)
3. with adjoint calculations for one in ten sample points (partly gradient-enhanced)

While the first and second optimization serves to show the influence of gradient information on the prediction, the third optimization is included to demonstrate the potential of sampling strategies that decide where to calculate gradient information.

All optimizations start by calculating the baseline design, which fulfills all constraints. Afterwards 3 samples around the baseline member are created by randomly varying parameters of the baseline design. Each parameter is selected for variation with a probability of 20%. When selected, the parameter is modified by a normally distributed random offset with a standard deviation of 10% of the parameters variation range. Afterwards the optimization loop is started which comprises the following steps

1. Re-train meta-models based on the available samples if necessary.
2. Perform an optimization on the meta-models and try to find N members with the largest expected volume gain, with zero (or minimal) predicted distance to constraints.
3. Evaluate all objectives and constraints for the predicted members and calculate gradient information if desired.
4. Include this information in a database of all evaluated samples.

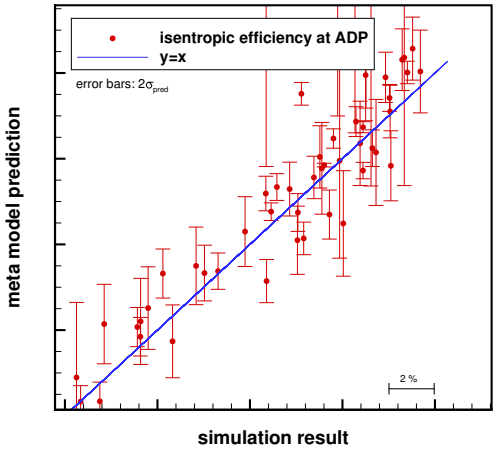
Note that the steps are parallelized, which mean that at any time there are up to N evaluation processes, and steps 1 and 2 are run whenever one evaluation finishes.



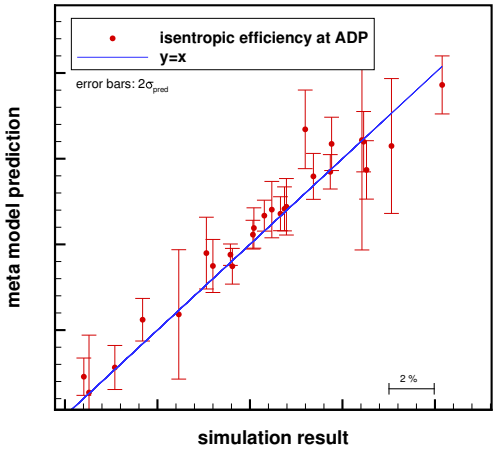
(a) ADP (b) NSP

Figure 6: Convergence of the adjoint solver for different operating points at the baseline design

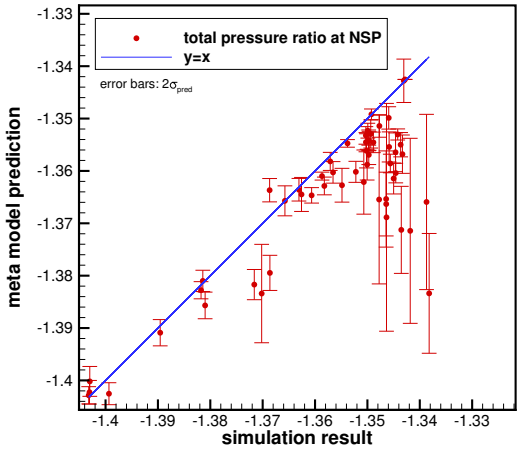
Figure 6 shows the convergence of the adjoint solver for two operating points. The left part shows the simulations for the ADP while the right part displays the convergence in the near stall point. The adjoint



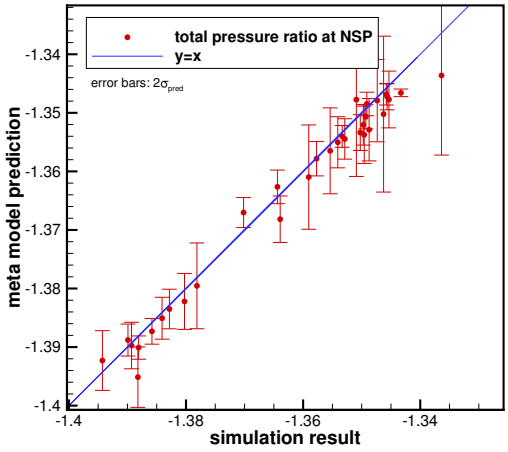
(a) First objective, gradient-free model



(b) First objective, gradient-enhanced model



(c) Second objective, gradient-free model

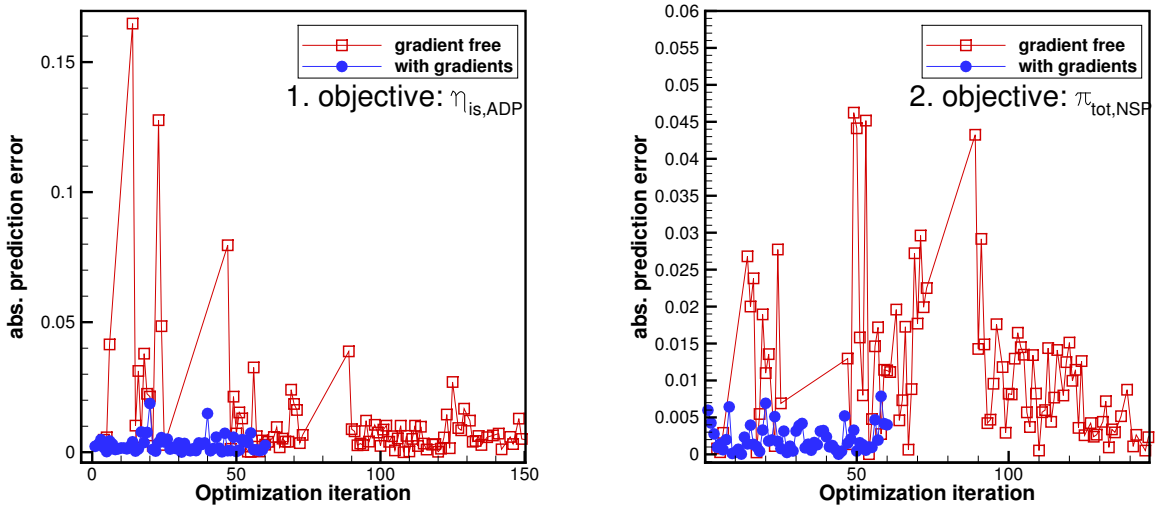


(d) Second objective, gradient-enhanced model

Figure 7: Influence of gradient information on the prediction of mean and variance over the real simulation result for both objectives.

solver converges satisfactory in both operating points and exhibits the same residual reduction rate as the primal solver.

The key assessment criterion for the prediction quality of a meta-model is the error between the statistical prediction and the evaluation of the simulation model. Figure 7 displays the predictions produced by the meta-models for the objectives in the gradient-free and fully gradient-enhanced optimization. The prediction is displayed as the mean value together with the error bar denoting the confidence interval of the prediction $2\widehat{\sigma}$ on the y axis. The observed simulation is represented on the x axis. With a perfect model both would be identical and would therefore lie on the blue line. The closer a point is to the blue line, the better was the corresponding prediction. However, a good prediction of the variance is equally important, since this influences the choice of sampling through the expected volume gain criterion. It can be seen that the gradient-free optimization (7a and 7c) has points with greater distance to the line of perfect prediction and also more points for which the true value does not lie within the confidence interval of $2\widehat{\sigma}$. Both criteria are much better met for the gradient based optimization (7b, 7d). Note that for the gradient-free optimization, only every other point is plotted to maintain clarity. The evolution of the meta-models prediction error during



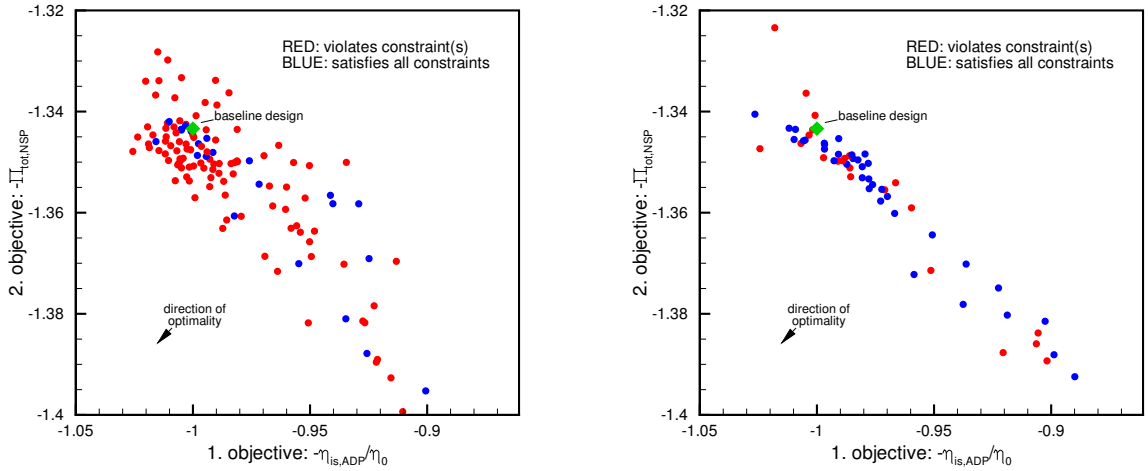
(a) 1. Objective Function

(b) 2. Objective Function

Figure 8: Evolution of the meta-model prediction error over the optimization iterations, comparing gradient-free and gradient-enhanced optimization

the course of both optimizations is depicted in figure 8. For both objectives one can see the prediction error without gradients (red curve) and the gradient based optimization (blue curve). The prediction error is an order of magnitude smaller for the gradient-enhanced optimization, directly at the start, while the gradient-free optimization reaches this accuracy only after roughly 150 evaluations. Note that both plots (Fig. 8a and Fig. 8b) start after the initial design of experiments which included 3 members in both cases. It seems as the amount of information from 3 adjoint evaluations is sufficient to reduce the prediction error as much as 150 primal evaluations. Note that the prediction error does not sink much further. This may be interpreted as a (local) saturation of the model - additional information does not reduce the prediction error any more. It would be inefficient to spend time on further adjoint calculations in the vicinity of such points. However it might be beneficial to add further gradient information at later optimization stages, as the correlation of new points with the initial samples might be reduced.

Figure 9 shows the results from the gradient-free and the full gradient optimization in objective function space. Since the optimizer is formulated as a minimizer the objectives are negated and the target of the objective is towards the lower left corner. Each dot represents one design evaluation; red dots show an evaluation which violates at least one constraint, whereas points which fulfill all constraints are colored blue. Points where one of the primal calculations failed to converge are not displayed. In both figures it can be



(a) Without gradient information

(b) With gradient information

Figure 9: Influence of gradient information by comparing Pareto fronts

observed, that the resulting Pareto front is nearly located on a straight line which means that for the chosen set of constraints, both objectives contradict each other. It can be seen, that the gradient based optimization produces a much higher rate of designs which satisfy all constraints and lie on the front of non-dominated members. However some members of the gradient-free optimization would still improve the Pareto front of the gradient based optimization. This may be due to the larger scattering of points due to the larger number of evaluations that could be performed in the gradient-free case. A good measure for judging the progress of multi-objective optimizations is the cumulative volume gain. The cumulative volume gain is calculated by adding in each step all members' volume gains, where the volume gain is calculated as in Equation 6 from simulation results. This measure is therefore suited to gauge the success of the predictions from the optimizer. In Figure 10 we see the evolution of the cumulative volume gain over the course of the three optimization

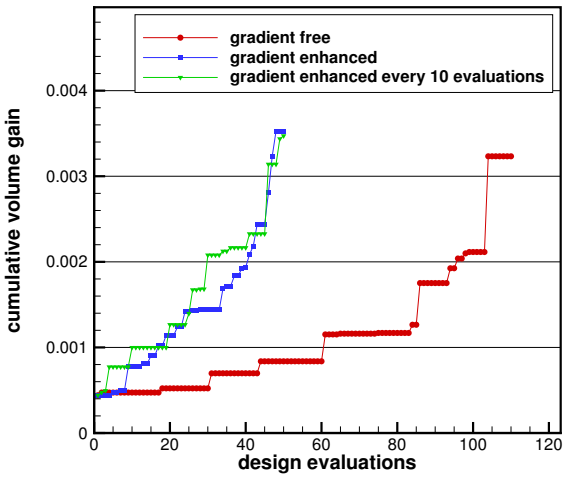


Figure 10: Cumulative volume gain for the three optimizations

studies: The gradient-free, the fully gradient-enhanced and the partially gradient-enhanced from calculating gradients only in one of ten design evaluations. First of all, it can be observed, that the curves corresponding

to gradient-enhanced optimizations feature a much higher slope and therefore show that both gradient-enhanced optimizations lead to more improvement per evaluation than the gradient-free optimization. One can observe various plateaus in each curve which stem from evaluations that either violate a constraint or turned out not to lie on the Pareto front. Both phenomena are caused by the prediction error of the meta-models. Both gradient-enhanced optimizations are comparable in slope, from which we can conclude, that evaluating gradients only for every tenth member delivers sufficient information. Further research into good decision criteria on where to calculate gradient information seems promising. It can be observed, that the partly gradient-enhanced optimization sometimes even advances faster than the fully gradient-enhanced optimization - however this could very well be an artifact from the stochastic optimization on the meta-models. Owing to this stochastic nature of the optimization measurements of the time saving through the adjoint method must be repeatedly performed under controlled circumstances and will be therefore left to another publication. However, it can be stated that in case of the fully gradient-enhanced optimization, all performance gains are absorbed by the additional effort for the adjoint evaluations. However the partially gradient-enhanced optimization offers a performance improvement over the gradient-free optimization.

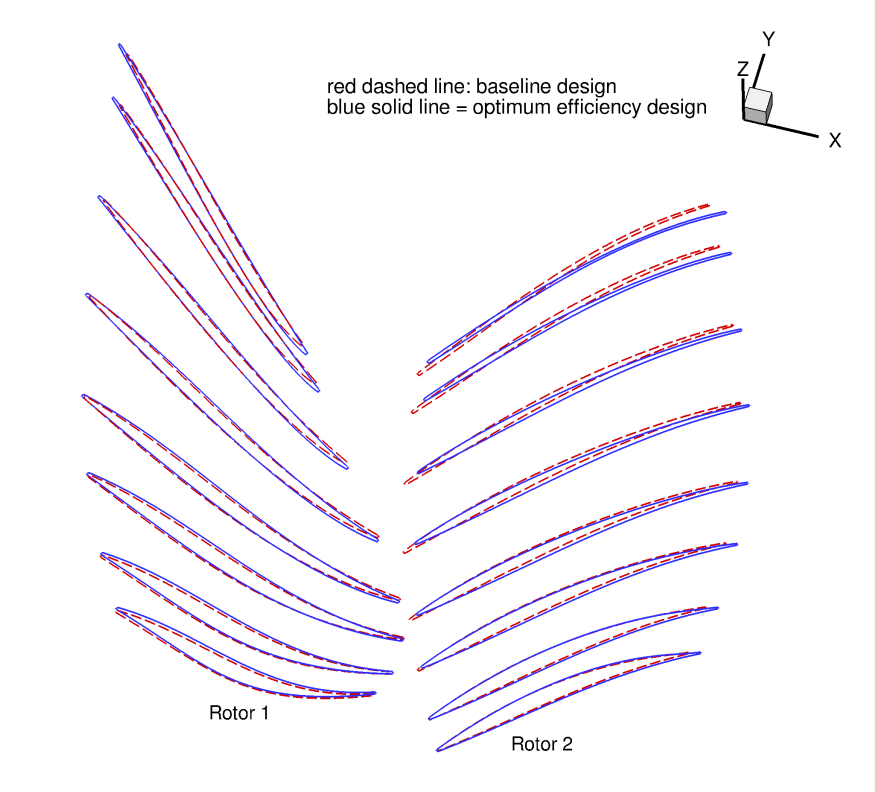
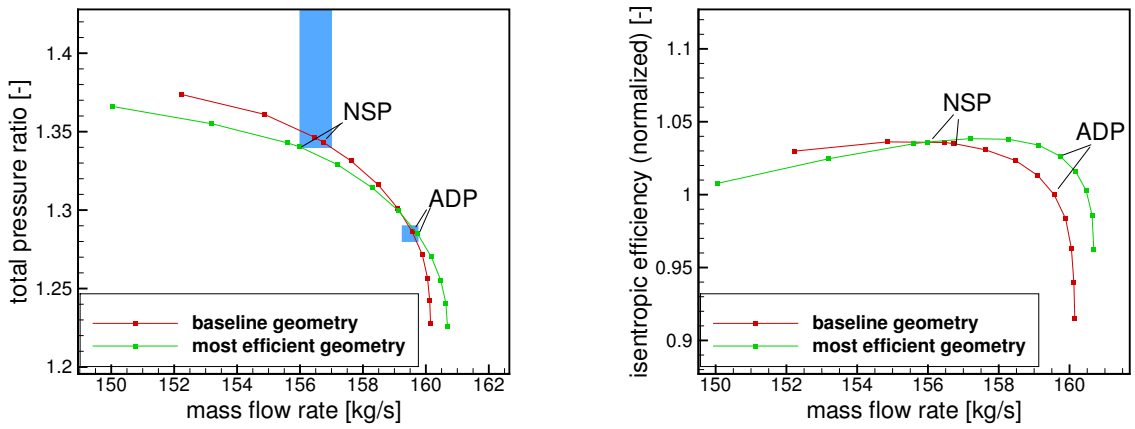


Figure 11: Comparison of cross sections on both rotors of baseline and most efficient member from the gradient based optimization

The optimization result is depicted in Fig. 11 as a comparison of all construction profiles between the baseline geometry and the most efficient member from the gradient-enhanced optimization.

An aerodynamic comparison of the working lines for both configurations is displayed in Fig. 12. From the working line (Fig. 12a) it can be seen that the optimized geometry is at the upper limit of the mass flow rate in the ADP and at the lower limits of mass flow rate and total pressure ratio in the NSP. This is another indication, that efficiency gains in the ADP (can be seen in Fig. 12b) are partly reached through at the expense of worsening the stall margin criterion. Note that the optimized geometry seems to feature a longer operating line. However this cannot be determined from the plots due to the large offset between the calculated operating points at the stall margin.



(a) total pressure ratio

(b) isentropic efficiency

Figure 12: Comparison of the initial member and most efficient member as speed lines

IV. Conclusion

The main objective of this paper is to demonstrate that the adjoint method is suited to improve the design of turbomachinery components in a realistic scenario. We show how practical necessities influence the choice of methods. To maintain the consistency to a constantly changing primal solver we use reverse mode algorithmic differentiation with subsequent performance improvements. We base the optimization on gradient-enhanced meta-modelling in order to achieve robustness against failure in the evaluation chain. The processes are described, assuming that non-adjoined tools are involved. We show that the surface- and mesh-generation tools can be included by finite differencing and finite element calculations without adjoints using a meta-modelling with finer sampling. The results show, that the gradients significantly improve the predictions for objective function and for the constraints. This leads to a steeper increase of volume gain as well as a higher rate of members that satisfy the constraints as well as a denser Pareto front. Results from the analysis of the evolution of the prediction indicate, that it is more efficient to calculate gradients only at selected sample points. Research into schemes for choosing where to calculate gradient information may offer significant improvement over the methods described here.

V. Acknowledgement

This project has received funding from the Clean Sky 2 Joint Undertaking under the European Unions Horizon 2020 research and innovation programme under grant agreement No [CS2-Engines ITD-2014-2015-01].

References

[1]Newman III, J. C., Taylor III, A. C., Barnwell, R. W., Newman, P. A., and Hou, G. J.-W., "Overview of sensitivity analysis and shape optimization for complex aerodynamic configurations," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 87–96.

[2]Duta, M. C., Shahpar, S., and Giles, M. B., "Turbomachinery design optimization using automatic differentiated adjoint code," *ASME Turbo Expo 2007: Power for Land, Sea, and Air*, American Society of Mechanical Engineers, 2007, pp. 1435–1444.

[3]Marta, A. C., Shankaran, S., Holmes, D. G., and Stein, A., "Development of Adjoint Solvers for Engineering Gradient-Based Turbomachinery Design Applications," *Proceedings of the ASME Turbo Expo 2009*, Vol. Volume 7: Turbomachinery, Parts A and B, June 2009.

[4]Peter, J. E. and Dwight, R. P., "Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches," *Computers & Fluids*, Vol. 39, No. 3, 2010, pp. 373–391.

[5]Wang, D. X., He, L., Li, Y. S., and Wells, R. G., "Adjoint Aerodynamic Design Optimization for Blades in Multistage Turbomachines—Part II: Validation and Application," *Journal of Turbomachinery*, Vol. 132, No. 2, 2010, pp. 021012.

[6]Backhaus, J., Aulich, M., Frey, C., Lengyel, T., and Voß, C., "Gradient Enhanced Surrogate Models Based on Adjoint CFD Methods for the Design of a Counter Rotating Turbofan," *Proceedings of the ASME Turbo Expo 2012*, 2012.

[7]Sagebaum, M., Özkaya, E., Gauger, N. R., Backhaus, J., Frey, C., Mann, S., and Nagel, M., "Efficient Algorithmic Differentiation Techniques for Turbo-machinery Design," *submitted for publication, AIAA-Paper*, Vol. ???, 2017, pp. ???

[8]Sagebaum, M., Özkaya, E., and Gauger, N. R., "Challenges in the automatic differentiation of an industrial CFD solver," *Evolutionary and Deterministic Methods for Design, Optimization and Control with Application to Industrial and Societal Problems (EUROGEN 2013)*, 2013.

[9]Sóbester, A. and Forrester, A., *Aerospace Series : Aircraft Aerodynamic Design : Geometry and Optimization*, Wiley, 2014.

[10]Yamazaki, W., Rumpfkeil, M. P., and Mavriplis, D. J., "Design optimization utilizing gradient/hessian enhanced surrogate model," *AIAA Paper*, Vol. 4363, 2010, pp. 2010.

[11]Han, Z.-H., Gortz, S., and Zimmermann, R., "Improving adjoint-based aerodynamic optimization via gradient-enhanced Kriging," *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, Nashville, Tennessee*, 2012.

[12]Zimmermann, R., "On the maximum likelihood training of gradient-enhanced spatial gaussian processes," *SIAM Journal on Scientific Computing*, Vol. 35, No. 6, 2013, pp. A2554–A2574.

[13]Riedmiller, M. and Braun, H., "A direct adaptive method for faster backpropagation learning: The RPROP algorithm," *Neural Networks, 1993., IEEE International Conference On*, IEEE, 1993, pp. 586–591.

[14]Forrester, A., Sóbester, A., and Keane, A., *Engineering Design via Surrogate Modelling: A Practical Guide*, Wiley, 2008.

[15]Jones, D. R., Schonlau, M., and Welch, W. J., "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, Vol. 13, No. 4, 1998, pp. 455–492.

[16]Dalbey, K. R., "Efficient and Robust Gradient Enhanced Kriging Emulators," SANDIA REPORT SAND2013-7022, Sandia Natianal Laboratories, Sandia National Laboratories Albuquerque, New Mexico 87185 and Livermore, California 94550, August 2013.

[17]Yang, H., Nürnberger, D., and Kersken, H.-P., "Towards Excellence in Turbomachinery Computational Fluid Dynamics," *Journal of Turbomachinery*, Vol. 2006, No. 128, 2006, pp. 390–402.

[18]Becker, K., Heitkamp, K., and Kügeler, E., "Recent Progress In A Hybrid-Grid CFD Solver For Turbomachinery Flows," *Proceedings Fifth European Conference on Computational Fluid Dynamics ECCOMAS CFD 2010*, 2010.

[19]Wilcox, D. C., "Reassessment of the Scale-Determining Equation for Advanced Turbulence Models," *AIAA Journal*, Vol. 26, No. 11, Nov. 1988, pp. 1299–1310.

[20]Kato, M. and Launder, B. E., "The Modeling of Turbulent Flow Around Stationary and Vibrating Square Cylinders," *9th Symposium on Turbulent Shear Flows*, 1993, pp. 10.4.1–10.4.6.

[21]Bardina, J., Ferziger, J. H., and Rogallo, R. S., "Effect of rotation on isotropic turbulence: computation and modelling," *J. Fluid Mech.*, Vol. 154, 1985, pp. 321–336.

[22]Engels-Putzka, A., Backhaus, J., and Frey, C., "On the usage of finite differences for the development of discrete linearised and adjoint CFD solvers," *Proceedings of the 6th. European Conference on Computational Fluid Dynamics - ECFD VI*, edited by E. Oñate, X. Oliver, and A. Huerta, Juli 2014.

[23]Denton, J. D., "The Calculation of Three-Dimensional Viscous Flow Through Multistage Turbomachines," *Journal of Turbomachinery*, Vol. 114, No. 1, Jan. 1992, pp. 18–26.

[24]Giles, M. B., "Nonreflecting Boundary Conditions for Euler Calculations," *AIAA Journal*, Vol. 28, No. 12, 1990, pp. 2050–2058.

[25]Lengyel-Kampmann, T., Bischoff, A., Meyer, R., and Nicke, E., "Design of an economical counter rotating Fan: comparison of the calculated and measured steady and unsteady results," *ASME Turbo Expo 2012: Turbine Technical Conference and Exposition*, American Society of Mechanical Engineers, 2012, pp. 323–336.

[26]Giles, M. B. and Pierce, N. A., "An introduction to the adjoint approach to design," *Flow, turbulence and combustion*, Vol. 65, No. 3-4, 2000, pp. 393–415.

[27]Griewank, A., *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation (Frontiers in Applied Mathematics)*, Society for Industrial and Applied Mathematics, 2000.

[28]Giles, M., Ghate, D., and Duta, M., "Using automatic differentiation for adjoint CFD code development," 2005.

[29]Mader, C. A., RA Martins, J., Alonso, J. J., and Der Weide, E. V., "ADjoint: An approach for the rapid development of discrete adjoint solvers," *AIAA journal*, Vol. 46, No. 4, 2008, pp. 863–873.

[30]Backhaus, J., Engels-Putzka, A., and Frey, C., "a code-coupling approach to the implementation of discrete adjoint solvers based on automatic differentiation," *VII European Congress on Computational Methods in Applied Sciences and Engineering, ECCOMAS Congress 2016*, June 2016.

[31]Frey, C., Kersken, H.-P., and Nürnberger, D., "The discrete adjoint of a turbomachinery RANS solver," *ASME Turbo Expo 2009: Power for Land, Sea, and Air*, American Society of Mechanical Engineers, 2009, pp. 345–354.

[32]Dwight, R. P. and Brezillon, J., "Effect of approximations of the discrete adjoint on gradient-based optimization," *AIAA journal*, Vol. 44, No. 12, 2006, pp. 3022–3031.

[33]Griewank, A. and Walther, A., "Algorithm 799: revolve: an implementation of checkpointing for the reverse or adjoint mode of computational differentiation," *ACM Transactions on Mathematical Software (TOMS)*, Vol. 26, No. 1, 2000, pp. 19–45.

[34]Christianson, B., "Reverse accumulation and attractive fixed points," *Optimization Methods and Software*, Vol. 3, No. 4, 1994, pp. 311–326.

[35]Dwight, R. P. and Brezillon, J., "Efficient and robust algorithms for solution of the adjoint compressible Navier–Stokes equations with applications," *International journal for numerical methods in fluids*, Vol. 60, No. 4, 2009, pp. 365–389.

[36]Albring, T., Dick, T., and Gauger, N. R., "Assessment of the Recursive Projection Method for the Stabilization of Discrete Adjoint Solvers," *submitted for publication, AIAA-Paper*, Vol. ???, 2017, pp. ???

[37]Naumann, U., "DCO Short Introduction," Tech. rep., NAG, 2015.

[38]Lengyel-Kampmann, T., *Vergleichende aerodynamische Untersuchungen von gegenläufigen und konventionellen Fanstufen für Flugtriebwerke*, Phd thesis, Ruhr-Universität Bochum, August 2015.

[39]Görke, D., Le Denmat, A.-L., Schmidt, T., Kocian, F., and Nicke, E., "Aerodynamic and mechanical optimization of CF/PEEK blades of a counter rotating fan," *Proceedings of the ASME Turbo Expo 2012*, 2012.

[40]Aulich, A.-L., Goerke, D., Blocher, M., Nicke, E., and Kocian, F., "Multidisciplinary automated optimization strategy on a counter rotating fan," *ASME Turbo Expo 2013: Turbine Technical Conference and Exposition*, American Society of Mechanical Engineers, 2013, pp. V06BT43A007–V06BT43A007.

[41]Schmitz, A., "Entwicklung eines objektorientierten und parallelisierten Gradient Enhanced Kriging Ersatzmodells," Tech. rep., Fernuniversität Hagen, September 2013, in German, http://elib.dlr.de/85734/.

[42]Dhondt, G., *The finite element method for three-dimensional thermomechanical applications*, John Wiley & Sons, 2004.

[43]Voigt, C., Wellner, J., Morsbach, C., and Kügeler, E., "Conquer the Terabyte-scale: Post-processing of high resolution unsteady CFD data for turbomachinery analysis," *6th European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2012)*, Vienna, Austria, Sept. 2012.