XI International Conference on Computational Methods for Coupled Problems in Science and Engineering
COUPLED PROBLEMS 2025
A. Larese, S. Perotto and G. Rozza (Eds)

HIGH-ORDER MULTIDISCIPLINARY TIME INTEGRATION TOWARDS ADAPTIVE TIME STEPPING

IMRI SHUVI, FLORIAN ROSS AND ARTHUR STÜCK

German Aerospace Center (DLR)
Institute of Software Methods for Product Virtualization
Nöthnitzer Straße 46b, 01187 Dresden
e-mail: imri.shuvi@dlr.de, florian.ross@dlr.de, arthur.stueck@dlr.de
web page: https://www.dlr.de/en/sp

Key words: MDAO frameworks, implicit solution algorithms, adaptive time stepping, unsteady multidisciplinary design analysis and optimization (U-MDAO)

Abstract. The aim of this work is to carry over adaptive higher-order time-stepping techniques – well-known and established for example in the context of single-disciplinary (CFD) time integration – to multidisciplinary design analysis and optimization (MDAO). To this end, an MDAO framework extension built on OpenMDAO, RKOpenMDAO, was developed to allow for time-accurate multidisciplinary high-order time integration. RKOpenMDAO provides MDAO framework capabilities – such as multidisciplinary gradient-accumulation and monolithic nested solution techniques – usually known from steady-state type MDAO scenarios for multidisciplinary implicit time integration. It was enabled for adaptive time stepping in this work. The multidisciplinary error in time is estimated by means of embedded Runge–Kutta schemes. The new-generation CFD Software by ONERA, DLR and Airbus (CODA) was modularly integrated into the suggested MDAO framework approach for multidisciplinary, adaptive time-stepping in a time-accurate way. Numerical experiments were carried out for an elementary unsteady aeroelastic case of a NACA0012 section in compressible Euler flow coupled to a torsional spring. With the MDAO-framework approach we could show that (a) advanced implicit solution capabilities with nested multidisciplinary solvers can be used on the level of the monolithic multidisciplinary system, (b) a computationally efficient error estimator based on embedded Runge–Kutta methods is able to successfully drive the adaptive time-step control, (c) the targeted temporal accuracy can be achieved by effectively adjusting the time-step, and (d) the suggested MDAO-framework extension, RKOpenMDAO, is applicable for multiphysics analyses with high-fidelity simulation components.

1 Introduction

Advanced time-accurate simulation capabilities for multidisciplinary problems are vital, e.g. to reliably cover the flight envelope in the design analysis and optimization of aircraft. Simulation efficiency is a key factor, particularly in combination with unsteady and high-fidelity predictions. Error-controlled adaptive time stepping is a proven approach to save computational resources while meeting a specified error target in time. It is well established in single-disciplinary

scenarios like computational fluid dynamics (CFD) [10] and/or computational structure dynamics (CSM) [13]. A few examples for multidisciplinary/multiphysics adaptive time-stepping can be found in the literature, e.g. for fluid-structure interaction [11]. In such coupled problems, achieving adaptive time stepping can be difficult as the standard methods require (some) monolithic time integration capabilities. This can be a challenge in conjunction with disciplinary time integrators loosely being coupled together, since interdisciplinary error estimates must be derived from the disciplinary estimates.

In this paper, we present and investigate a monolithic time integration method for coupled problems; to the knowledge of the authors, this is a new contribution in the context of gradient-enabled monolithic frameworks for high-fidelity MDAO. The implementation is based on the MDAO framework *OpenMDAO* [1]. The baseline version of the time stepping extension *RKOpenMDAO* presented previously [4] offers multidisciplinary time integration with homogeneous time stepping and diagonally implicit Runge–Kutta (DIRK) schemes. It integrates directly with the OpenMDAO, extending features like the automation of sensitivity analysis to the unsteady domain, both in forward and reverse mode. It does so by repeatedly solving a pseudo-steady OpenMDAO problem in a memory-efficient monolithic manner. In this work, RKOpenMDAO was further extended to handle embedded Runge–Kutta schemes for error estimation. Embedded Runge–Kutta schemes have, in addition to their primary high-order scheme, a secondary low-order scheme that is computationally cheap and allows to estimate the numerical error by comparing high- and low-order approximations. Based on the error estimate, the time step size is controlled comparing the error estimate to a prescribed tolerance value.

The remainder of the paper is organized as follows: in Section 2, we briefly introduce the handling of ordinary differential equations by RKOpenMDAO. It will be described how the extension sees ordinary differential equations (ODEs), how time integration is applied on them, and lastly how the the time integration produces an error estimate for step size control. In Section 3, the multidisciplinary time-step adaptation capabilities are numerically investigated for a coupled airfoil-spring problem, followed by conclusions and outlook in Section 4.

2 Adaptive Time Stepping in RKOpenMDAO

RKOpenMDAO takes an implicit ODE formulation:

$$\begin{aligned} 0 &= F\left(t, x, \dot{x}\right) \,, \\ x(t_0) &= x_0 \,, \end{aligned}$$

where F is the amalgamation of the disciplinary ODEs and their algebraic closure relations. These disciplinary ODEs and relations can originate both as analytical ODEs, or from partial differential equations (PDEs) which were semi-discretized in space beforehand. Here:

$$x = \begin{pmatrix} x^{\text{Disc } 1} \\ x^{\text{Disc } 2} \\ \vdots \\ x^{\text{Disc } m} \end{pmatrix}$$

is the multidisciplinary state vector, comprised of the disciplinary vectors $x^{\text{Disc } i}$ of possibly differing in dimensions. Together with the multidisciplinary initial condition vector x_0, this

c_1	a_{11}	0	...	0
c_2	a_{21}	a_{22}	$\ddots$	0
$\vdots$	$\vdots$		$\ddots$	0
c_S	a_{S1}	a_{S2}	...	a_{SS}
	b_1	b_2	...	b_S

Table 1: Butcher tableau of a general S-stage DIRK scheme

forms an initial value problem (IVP) that is integrated in time via a DIRK scheme. As with every Runge–Kutta scheme, DIRK schemes can be described by a Butcher tableau (Table 1). For DIRK schemes, the Butcher matrix A in the tableau is a lower triangular matrix, leading to a sequential evaluation of the time stages. Thus, one time step reads:

$$\begin{aligned} x_{n+1} &= x_n + \Delta t \sum_{i=1}^{S} b_i k_i \,, \\ t_{n+1} &= t_n + \Delta t \,, \end{aligned}$$

which in turn has multiple stages:

$$\begin{aligned} s_i &= \sum_{j=1}^{i} a_{ij} k_j \,, \\ t_n^i &= t_n + \Delta t c_i, \\ x_n^i &= x_n + \Delta t \left(s_i + a_{ii} k_i \right) \,, \\ 0 &= F\left(t_n^i, x_n^i, k_i \right) \,. \end{aligned}$$

Here, x_m and t_m are the multidisciplinary state vector respectively the time at time step m, x_m^l and t_m^l the state and time at the lth stage of the mth step, k_l the resulting update at the lth stage, and s_l the weighted sum of the stages prior to stage l in the current step.
In order to introduce adaptive step sizes for control of the time discretization error, an estimate for the local time discretization error needs to be introduced. The *local* time integration error originates from the application of a single step of the time integration scheme. In contrast, the *global* time integration error is the deviation from the analytical solution from the start of the time integration until the considered time t^*. The order p of a Runge–Kutta method refers to the *global* error, while the *local* error has order $p + 1$.
For Runge–Kutta methods, a convenient way to get an estimate for the local error is via the use of embedded methods [8, 6]. These methods have a second embedded set of weights, as can be seen in Table 2. This second set of weights represent a lower order scheme, usually of order $p - 1$:

$$\hat{x}_{n+1} = x_n + \Delta t \sum_{i=1}^{S} \hat{b}_i k_i \,.$$

Given the states x_{n+1} of local order $p + 1$ and $\hat{x}_{n+1}$ of local order p, an error estimate of order p can be calculated:

$$e_{n+1} = x_{n+1} - \hat{x}_{n+1} \in O(\Delta t^p) \,.$$

c_1	a_{11}	0	$\dots$	0
c_2	a_{21}	a_{22}	$\ddots$	$\vdots$
$\vdots$	$\vdots$		$\ddots$	0
c_S	a_{S1}	a_{S2}	$\dots$	a_{SS}
	b_1	b_2	$\dots$	b_S
	$\hat{b}_1$	$\hat{b}_2$	$\dots$	$\hat{b}_S$

Table 2: Butcher tableau of a general S-stage DIRK scheme including an embedded scheme

Note that while the weights of the primary and the embedded scheme are different, and thus are the resulting states, the updates k_i calculated at the intermediate stages remain the same. This makes the computation of the error estimate relatively cheap, only requiring an additional sum instead of further function evaluations or nonlinear solves.
Based on the error estimate, the step size can be controlled to obtain a certain local tolerance ϵ. For this control, an integral error controller is used:

$$\Delta t_{new} = \kappa \Delta t_{old} \left(\frac{\epsilon}{||e_{n+1}||_2} \right)^{p+1},$$

where $|| \cdot ||_2$ is the usual euclidean norm, κ is a safety factor that is used to underestimate the step size in order to reduce the number of rejections, and Δt_{new} and Δt_{old} are the new and old step sizes respectively. In case $||e_{n+1}||_2 \leq \epsilon$, the step is accepted, and Δt_{new} is used as the step size for the next time step. However, $||e_{n+1}||_2 > \epsilon$ leads to a rejection of the step, using Δt_{new} for as new step size for a repeated step from the current time.
Note that the implementation in RKOpenMDAO allows for a more flexible choice of both the used norm for the error measure, as well as for the error controller. For norms, e.g. arbitrary p-norms are allowed, while the error controller can take the influence from past time steps into account following a formulation from Söderlind [5].

3 Numerical Experiment

Setup

We considered a 2D NACA0012 airfoil in compressible Euler flow coupled to a torsional spring (see Figure 1). The airfoil was simulated using CODA[7]. CODA is the computational

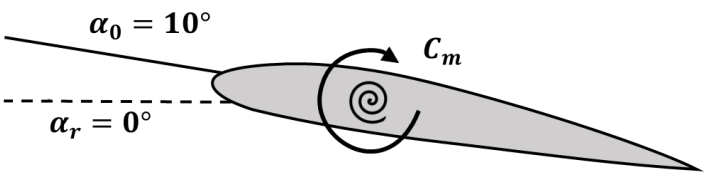

Figure 1: An airfoil coupled to a torsional spring, resulting in a decaying oscillation to the resting angle α_r.

fluid dynamics (CFD) software being developed as part of a collaboration between the French Aerospace Lab ONERA, the German Aerospace Center (DLR), Airbus, and their European research partners. CODA is jointly owned by ONERA, DLR and Airbus. The CFD software is

used to compute the space-discretized residual contributions, $R(W, x, \dot{x})$, to the unsteady Euler equations, resulting in:

$$\dot{W} + R(W, x, \dot{x}) = 0\,, \tag{1}$$

wherein W is the flow state containing the conservative variables, x and $\dot{x}$ are the coordinates and velocities of the mesh nodes. A second-order finite volume method is used in conjunction with a Roe upwinding convection scheme. The CFD mesh for the airfoil consists of approximately 2000 cells (Figure 2). Furthermore, CODA is used to compute the integral pitching

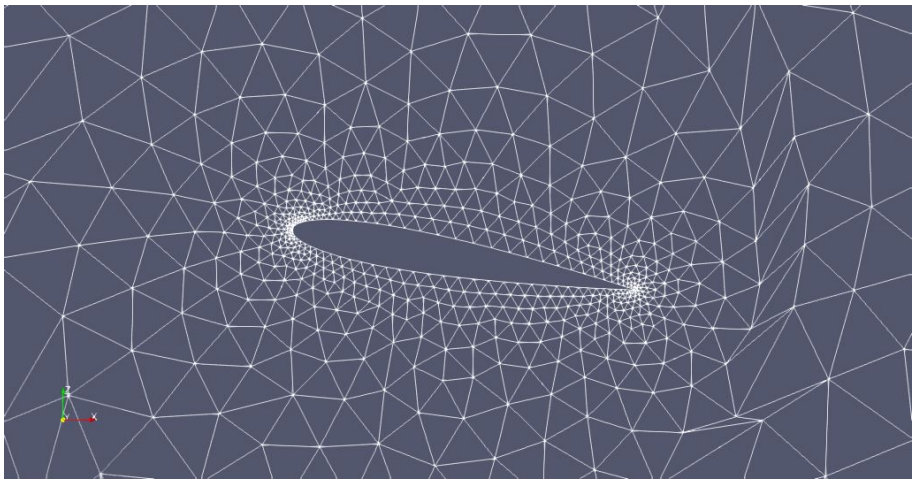

Figure 2: Close-up view of the initial CFD mesh.

moment based on the flow state W:

$$C_m = C_m(W)\,. \tag{2}$$

The integral pitching moment contributes to the equation of motion for the torsional spring,

$$I\ddot{\alpha} + \mu\alpha - C_m = 0\,, \tag{3}$$

wherein α represents the angle of the spring, $\ddot{\alpha}$ its angular acceleration, I the moment of inertia, and μ the torsional stiffness constant. The angle of the spring acts as angle of attack for the airfoil, realized via a rotation of the background mesh by rigid body motion around the aerodynamic center:

$$\begin{pmatrix} x \\ \dot{x} \end{pmatrix} = M(\alpha, \dot{\alpha}, \hat{x})\,. \tag{4}$$

Here, $\hat{x}$ is the original undeformed mesh, $\dot{\alpha}$ the angular velocity of the spring, and M the effect of the deformation caused by α on $\hat{x}$, resulting in the deformed coordinates x and their velocities $\dot{x}$.

This coupled system of equations is integrated in time via RKOpenMDAO. The airfoil is released from a prescribed, non-equilibrium angle of attack of 10° with the flow field initialized from a converged steady-state CFD solution at this angle of attack. This leads to a decaying oscillating motion of the airfoil. Quantities used for the nondimensional description and further parameters can be found in Table 3. Throughout this study, the step-size control for adaptive

Chord length:	$1\ \mathrm{m}$	Mach number:	0.1
Farfield pressure:	$10^5\ \frac{\mathrm{kg}}{\mathrm{m\,s^2}}$	Nondimensional moment of inertia I:	76.93
Farfield temperature:	$300\ \mathrm{K}$	Nondimensional stiffness μ:	10^{-3}
Farfield density:	$1.3\ \frac{\mathrm{kg}}{\mathrm{m^3}}$	Nondimensional final time T:	10

Table 3: Nondimensionalization quantities and parameters.

time stepping always uses the aforementioned integral controller with a safety factor of 0.85. All unsteady simulations were carried out until a nondimensional time $\hat{t}_F = 10$, with the time reference being approximately $0.0036\ \mathrm{s}$ following Table 3. This time interval corresponds to approximately 1.25 free-stream convection times over the chord length, or 1 period of the decaying oscillation of the airfoil. Note that only time discretization errors are considered here by separating space and time according to the method of lines [12]. For comparison, a reference solution was created using the classic fourth-order explicit Runge–Kutta method, using a nondimensional step size of 10^{-3}. The nonlinear systems that have to be solved each time stage are converged using a Newton solver with a prescribed relative tolerance of 10^{-8}. The underlying linear systems are solved by a nested solver stack consisting of a flexible GMRES method preconditioned by a single Block-Jacobi sweep which, in turn, approximately inverts the disciplinary block systems by linear block solvers provided by the disciplines.

Results

A key condition for adaptive step-size control in time to be successful is that the design-order of the local error estimate of the embedded Runge–Kutta scheme is achieved. To show that the condition is met for this problem, various embedded DIRK schemes with orders in the range of 2 to 4 were simulated until $\tilde{t}_f = 10$ with homogeneous step sizes in the range of 0.01 to 0.1. The used schemes can be found in [6], the naming used here indicates whether the scheme is SDIRK or ESDIRK, as well as the numerical design order of their primary scheme. The results found in Figure 3 show that the expected design order is achieved, with only small deviations observed for the schemes of order 4.

As the design order of the error estimator is met, error control is feasible for the considered problem.

Comparing against a homogeneous scheme, adaptive time stepping is expected to lead to a more uniform distribution of global error over time. The time evolution of the global error (top) is plotted together with the time-step size (bottom) in Figure 4for both homogeneous and adaptive time step sizes. An adaptive scheme in combination with a third-order SDIRK method is used, with a prescribed tolerance of 10^{-6} per time step. The homogeneous step was defined to be the average step size of the adaptive run in order to arrive at a comparable computational effort. It can be seen that the adaptive scheme provides a more uniform global error distribution. In the early time steps, the homogeneous scheme shows a relatively high error at the start. Note that the problem converges towards a stable, steady-state solution. This is reflected in the time step size of the adaptive scheme, leading to a lower error that is almost preserved over the simulation time. Accordingly, the the adaptive scheme produces small step sizes until $\tilde{t} \approx 2$. Toward the end of the simulation time, the solution of the adaptive scheme even accepts larger

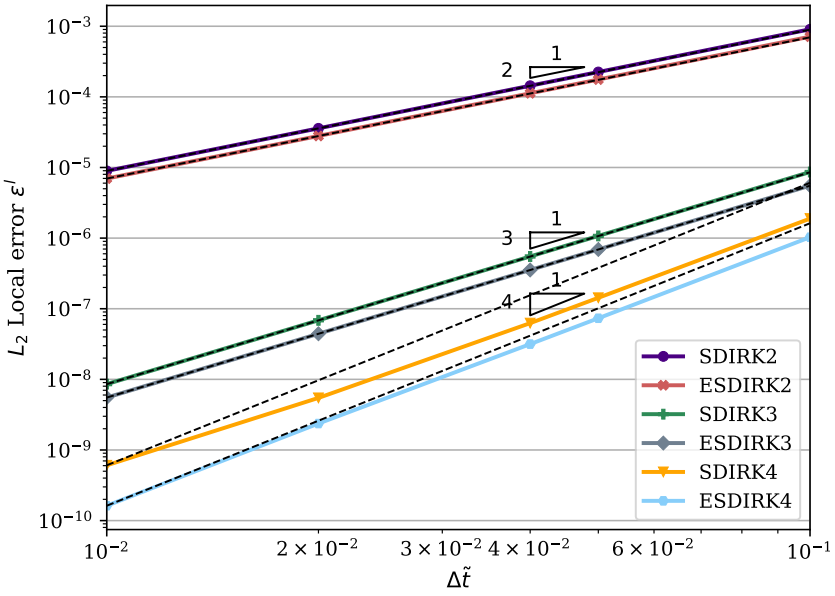

Figure 3: Achieved order of the error estimator for various DIRK schemes.

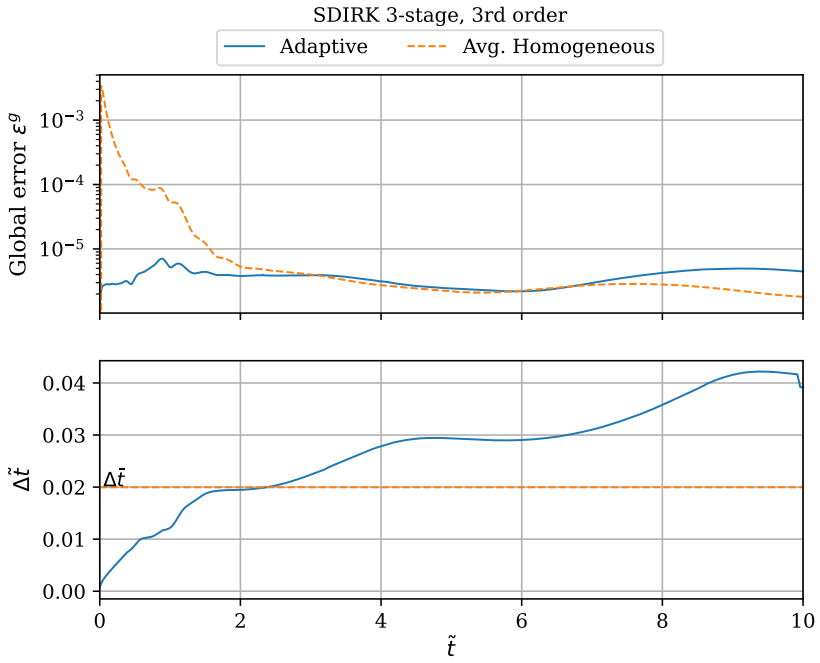

Figure 4: Comparison of the global error and step size of an adaptive against an homogeneous scheme.

time discretization errors than the homogeneous scheme. Since the prescribed local tolerance is met, the controller prioritizes an increase of step size over lowering the error.

Even though the prescribed tolerance was set to 10^{-6}, the adaptive scheme only achieved global errors in the range of 10^{-5}. This is to some sense counter-intuitive, but it can be explained by

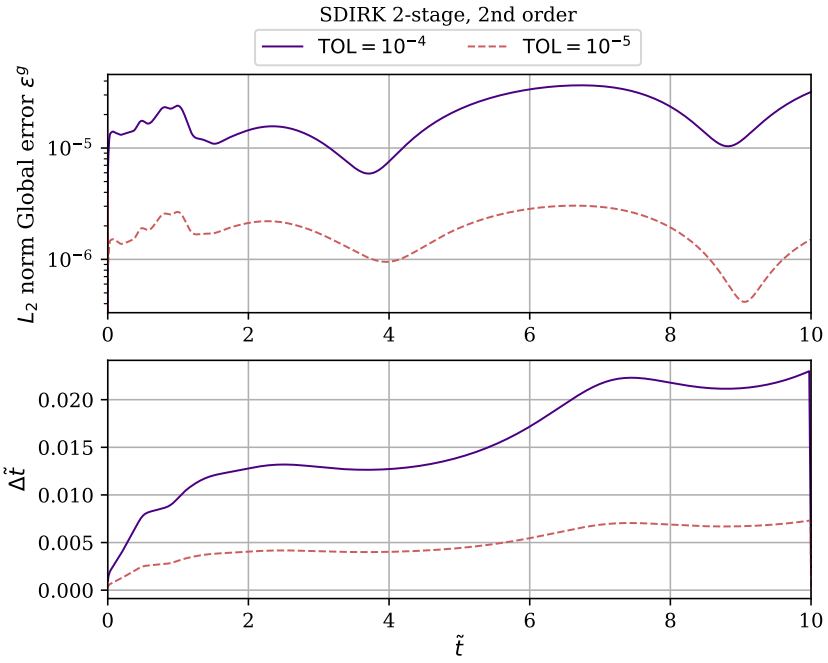

Figure 5: Comparison of global error and step size for an adaptive scheme with two different tolerances.

the fact that the prescribed tolerance is related to the *local* error – and only indirectly to the the *global* error. Thus, the prescribed tolerance can not be observed. However, the prescribed tolerance still controls the global error. In Figure 5, a second-order SDIRK scheme is run with local tolerances of 10^{-4} and 10^{-5}. Even though the prescribed local tolerances cannot be observed by looking at the global tolerances, the change of the order of magnitude in the error still becomes obvious in the logarithmic plot.

4 Conclusions and Outlook

We presented an MDAO framework approach for adaptive higher-order implicit time stepping. The method was investigated for a CFD-coupled problem consisting of a NACA0012 airfoil in compressible Euler flow connected to a torsional spring. The airfoil section was released from an initial spring deflection to reach a steady solution. It was shown that the error estimate achieves the expected design order, fulfilling a necessary prerequisite for the used adaptive time integration scheme. As expected, the adaptive time-stepping led to a more uniform error distribution over the simulation compared to a homogeneous time stepping approach. Furthermore, it was investigated how the prescribed local tolerance influences the achieved global error. In ongoing work, the added value of the multidisciplinary adaptive time-stepping approach is being investigated in terms of accuracy and robustness.
In the future, we want to apply the presented adaptive multidisciplinary time-stepping approach implemented in the open-source library RKOpenMDAO for the simulation of flexible aircraft, e. g. for gust encounter scenarios. The approach is deemed promising for such cases showing almost stationary behaviour over large parts of the simulated time, whereas the unsteady physics must be captured at certain points, e. g. when the gust directly interacts with the aircraft. Beyond the CFD component, the structural model of the aircraft and the equations of motion need

to be considered in the MDAO problem. We also intent to implement the necessary extensions to the adaptive time-stepping in RKOpenMDAO to support gradient-based optimization. To this end, unsteady multidisciplinary adjoint problems have to be calculated in an efficient manner. RKOpenMDAO already supports offline checkpointing via the *revolve* algorithm [9] to trade CPU runtime for memory in an optimal way.

REFERENCES

[1] Gray, J., Hwang, J., Martins, J., Moore, K. & Naylor, B. OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization. *Structural And Multidisciplinary Optimization*. **6** (2019)

[2] Hwang, J. & Munster, D. Solution of ordinary differential equations in gradient-based multidisciplinary design optimization. *2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, And Materials Conference*. (2018)

[3] Falck, R., Gray, J., Ponnapalli, K. & Wright, T. dymos: A Python package for optimal control of multidisciplinary systems. *Journal Of Open Source Software*. **6(59)** (2021)

[4] Roß, F., Büchner, A., Gottfried, S. & Stück, A. A High-Fidelity Framework Approach Enabling High-order Implicit Time Stepping For MDAO. *ECCOMAS Congress 2024 - 9th European Congress On Computational Methods In Applied Sciences And Engineering*. (2024), https://elib.dlr.de/210060/

[5] Söderlind, G. Automatic control and adaptive time-stepping. *Numerical Algorithms*. **31** (2002)

[6] Kennedy, C. & Carpenter, M. Diagonally implicit Runge-Kutta methods for ordinary differential equations. A review. (2016)

[7] Leicht, T., Vollmer, D., Jägersküpper, J., Schwöppe, A., Hartmann, R., Fiedler, J. & Schlauch, T. DLR-Project Digital-X: Next generation CFD solver 'Flucs'. (2016,1)

[8] Butcher, J. Numerical Methods for Ordinary Differential Equations. (Wiley, 2016)

[9] Griewank, A. & Walther, A. Algorithm 799: revolve: an implementation of checkpointing for the reverse or adjoint mode of computational differentiation. *ACM Trans. Math. Softw.*. **26(1)**, (2000,3), https://doi.org/10.1145/347837.347846

[10] John, V. & Rang, J. Adaptive time step control for the incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering* **199(9)** (2000)

[11] Birken, P., Quint, K.J., Hartmann, S. et al. A time-adaptive fluid-structure interaction method for thermal coupling. *Comput. Visual Sci.* **13** (2010)

[12] Schiesser, W.E. The Numerical Method of Lines. (Academic Press, 1991)

[13] Rossi, D.F., Ferreira, W.G., Mansur, W.J., Calenzani, A.F.G. A review of automatic time-stepping strategies on numerical time integration for structural dynamics analysis. *Engineering Structures* **80** (2014)