The 9th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2024
3–7 June 2024, Lisboa, Portugal

TIME-SPECTRAL EXTENSION TO AN IMPLICIT NAVIER–STOKES SOLVER: INTEGRATION AND EFFICIENCY ASPECTS

Raphael Haupt[1] **and Arthur Stück**[1]

[1] German Aerospace Center (DLR)
Institute of Software Methods for Product Virtualization
Zwickauer Straße 46, 01069 Dresden, Germany
e-mail: [Raphael.Haupt, Arthur.Stück]@dlr.de
web page: https://www.dlr.de/en/sp

Key words: Time-spectral method (TSM), Navier–Stokes, RANS, implicit solution, scalability, periodic flow

Summary. The time-spectral method (TSM) promises a huge efficiency potential for engineering flows in which the solution can be expected to be predominantly periodic in time, such as flows through engines, wind turbines, helicopters or valves. For N harmonics in time, the TSM leads to a nonlinear equation system in the time domain governing a blocked state-vector composed of $2N+1$ steady-state like CFD solution vector blocks. The resulting large, sparse and blocked nonlinear equation systems require scalable and robust solution algorithms. We integrate the TSM in the CFD software CODA, a powerful and modular environment offering a wide range of models, discretization schemes and implicit solution algorithms. We discuss key aspects relevant for the integration of the TSM, focusing on the implicit solution strategy by means of nested algorithms of (non)linear solvers and linear preconditioners. The approach relies on a CFL-relaxed pseudo Newton-algorithm in combination with Krylov/GMRES schemes which, in turn, can be preconditioned by Block-Gauss–Seidel/Jacobi methods with a direct inversion of spatial/time-spectral element blocks. Moreover, a direct line-inversion approach based on the Thomas algorithm, which is provided by the base-line solver, is applied to the spatial/time-spectral line blocks of the TSM problem here. Different solver settings for the hierarchical solution method are investigated and discussed in terms of their efficiency for 2D and 3D aerodynamic TSM problems.

1 INTRODUCTION

For time-periodic configurations like helicopter, propellers and wind-turbine blades, motor engines or oscillating foils, engineers are often primarily interested in the prediction of fully-developed periodic states, e.g. to quantify the (performance) or other key properties of the design. Using time-stepping approaches to reach this goal can be unnecessarily expensive, as the transient history is of little interest in many cases. For such problems tailored methods like the TSM can take advantage of the expected periodicity of the solution over time. These methods do not compute an onset or transient history and can exhibit higher convergence rates than time-stepping approaches. The TSM, for example, results in a large steady-state like equation system that defines a periodic solution by means of a few (collocation) points in time to represent the dynamics of the system. Well-estabilished, alternatives to the TSM [1, 2, 3] that are tailored to time-periodic solutions are the Linear and Non-Linear Frequency Domain ((N)LFD) [4, 5, 6] or the Harmonic Balance method (HBM) [7]. While the Linear Frequency Domain method (LFD) only considers small/infinitesimal perturbations around the mean flow in the first harmonic of the

problem, the HBM allows to consider several harmonics in a coupled way to give a nonlinear prediction for larger amplitudes similar to the TSM. Unlike the HBM and the LFD, which are both formulated in the frequency domain and consequently are based on complex-number representations, the TSM uses a time-domain formulation and does not need complex value types and operators. This facilitates the integration into standard CFD software environments offering (implicit) steady-state solution capabilities.

2 THEORY

We start the derivation of the governing TSM equations from the semi-discrete Naviers–Stokes equations

$$\frac{dq}{dt} + R(q, \nabla q, x, t) = 0 \,, \tag{1}$$

where q represents the state vector (mass, momentum, energy and turbulence variables) and R is the residual of the viscous $\mathbf{F}^v$ and inviscid fluxes $\mathbf{F}^c$

$$R = \oint_{\partial V} (\mathbf{F}^c - \mathbf{F}^v) \cdot n \, dS \,, \tag{2}$$

wherein source-terms and the dependencies on $q, \nabla q, x$ and t were dropped for simplicity. The TSM applies the Fourier ansatz,

$$f_n = f(t_n) = \sum_{k=-H}^{H} \hat{f}_k e^{-i\omega k t_n} \,, \tag{3}$$

to the state $q(t)$, sampling at N equidistantly distributed points on a periodic time interval T. These are also called time instances $t_n = 0, 1, 2, ..., N-1$. The number of time instances needed to resolve the harmonics of interest H can be derived from the Nyquist–Shannon sampling theorem and results in $N = 2H + 1$ time instances. Gathering the states for all time instances into a state vector $\vec{q}$ enables us to denote the resulting system in a single vector equation

$$\vec{q} = \begin{pmatrix} q(t_0) \\ q(t_1) \\ \cdots \\ q(t_{N-1}) \end{pmatrix} = \begin{pmatrix} q^0 \\ q^1 \\ \cdots \\ q^{N-1} \end{pmatrix} . \tag{4}$$

Applying it to Equation (1) results in

$$\frac{d\vec{q}}{dt} + \vec{R} = \vec{0} \,, \tag{5}$$

with $\vec{R}$ representing a block vector of the corresponding residuals. As shown in [1], the following TSM time operator is obtained

$$\frac{d}{dt}(q^n) = \sum_{j=0}^{N-1} d_n^j q^j \,, \tag{6}$$

with d_n^j for even numbers of N

$$d_n^j = \begin{cases} \frac{2\pi}{T} \frac{1}{2} (-1)^{n-j} \cot(\frac{\pi(n-j)}{N}) & n \neq j \\ 0 & n = j \end{cases} \tag{7}$$

and for odd numbers of N, respectively,

$$d_n^j = \begin{cases} \frac{2\pi}{T}\frac{1}{2}(-1)^{n-j}\csc(\frac{\pi(n-j)}{N}) & n \neq j \\ 0 & n = j\,. \end{cases} \tag{8}$$

Since an even number of time instances leads to an even-odd decoupling, we only use odd numbers of time instances in this work. These derivative operators can now be computed for every time instance and gathered in a collocation matrix $\mathbf{D}$. Note that each row contains the same values in a shifted order. Thus, it is sufficient to compute them once and shift the rows accordingly, or to use a single vector and use special access patterns to obtain the correct values:

$$\dot{\vec{q}} = \frac{d\vec{q}}{dt} \approx \mathbf{D}\vec{q}\,. \tag{9}$$

Combining Equations (9) and (5), we obtain the effective TSM residual equation

$$\mathbf{D}\vec{q} + \vec{R} = \vec{0}\,. \tag{10}$$

To solve this nonlinear equation, we introduce a pseudo-time stepping method resulting in

$$\frac{\partial\vec{q}}{\partial\tau} + \mathbf{D}\vec{q} + \vec{R} = \vec{0}\,. \tag{11}$$

Choosing an implicit-Euler method with l denoting the pseudo-time step, yields

$$\frac{\Delta\vec{q}}{\Delta\tau} + \mathbf{D}\vec{q}^{\,l} + \vec{R}^l = \vec{0} \tag{12}$$

with $\Delta\vec{q} = \vec{q}^{\,l} - \vec{q}^{\,l-1}$ and $\vec{R}^l = \vec{R}(q^l)$. Since the nonlinear residual $\vec{R}^l$ depends on the unknown solution $\vec{q}^{\,l}$ a linearization is performed

$$\vec{R}^l \cong \vec{R}^{l-1} + \mathbf{J}\Delta\vec{q}\,, \tag{13}$$

with $\mathbf{J}$ referred to as the Jacobian matrix of the spatial residual contributions $\mathbf{J} := \frac{\partial\vec{R}}{\partial\vec{q}}$. Accordingly, we can write the effective equation system as

$$\left(\frac{\mathbf{I}}{\Delta\tau} + \mathbf{D} + \mathbf{J}\right)\Delta\vec{q} = -\vec{R}^{l-1}_{TSM}\,, \tag{14}$$

with $\vec{R}^l_{TSM} := D\vec{q}^{\,l} + \vec{R}(q^l)$ as the full TSM residual, which only depends on the flow quantities of the last pseudo time-step. This Newton-like method relaxed in pseudo-time reproduces a full-Newton approach for $\Delta\tau \to \infty$. We can now use a linear solver to solve

$$\mathbf{A}\,\Delta\vec{q} = \vec{b} \tag{15}$$

with

$$\mathbf{A} = \frac{\mathbf{I}}{\Delta\tau} + \mathbf{D} + \mathbf{J} \quad \text{and} \quad \vec{b} = -\left[\mathbf{D}\vec{q}^{\,l-1} + \vec{R}(q^{l-1})\right]\,. \tag{16}$$

3 IMPLEMENTATION

We implement the TSM as an extension in the C++ fluid dynamic solver library CODA, which is the computational 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. It supports Finite Volume and (higher-order) Discontinuous-Galerkin discretizations on unstructured meshes, implements different PDEs like Euler and Navier–Stokes equations in conjunction with a number of turbulence models as well as time-accurate implicit and explicit time stepping schemes. Selected C++ methods are wrapped to Python to enable their use in Python scripts in the context of the FlowSimulator HPC simulation environment. Among other functionalities, we use Python to define simulation parameters, control of the data flow and to include different add-ons like mesh manipulators or various linear equation solver libraries via the FlowSimulator. The default equation solver in CODA is the inhouse library Spliss [8], which uses a sparse matrix storage format with dense element blocks. This is an important property that also influences the choice to represent the TSM problem. Spliss offers a large variety of linear solvers and preconditioners and is constantly extended. Recently, Spliss has been extended by a mixed-precision mode [9] and an algebraic multigrid method [10], which have not been applied to the TSM yet, but are planed for use in future works. A deeper insight into CODA can be found in [11, 12].

As described in Section 1, the TSM problem is composed of several time instances. A straight-forward implementation would be to append each time instance to the end of both the matrix and vectors and add the collocation matrix contribution afterwards. In this work, we want to leverage the potential of the linear solver library Spliss, which is geared towards the solution of blocked equation systems. Spliss offers a block-storage formats for sparse matrices with dense element blocks together with direct element block/line-inversion capabilities. We want to take advantage of these features to efficiently solve the blocked and strongly-coupled equation systems of the TSM problem. Thus, the approach pursued here is inspired by the algorithm called GMRES-STI by Mundis and Mavriplis [1], in which the time-spectral contributions are bundled together element-wise in the preconditioner to allow for a direct TSM-element block inversion.

Two options to organize the degrees of freedom are illustrated in Figure 1. On the left side, each colorized square represents a single time instance with N spatial elements inside. Each spatial element, in turn, is a block containing all degrees of freedom of this element. Note that – for visualization reasons – the picture shows only one degree of freedom per spatial element block, e.g. one state variable within a finite-volume discretization. The size of this block is based on the number of state variables of the chosen equation system and the degrees of freedom of the chosen discretization method $(DG : element\ dependent,\ FV : 1)$, resulting in a size of $(N_{state} \cdot N_{disc})^2$. The off-diagonal coefficients are the contributions stemming from the collocation matrix. While this implementation is straight-forward, the resulting patterns would lead to a Gauss–Seidel or Jacobi-type solution applied to the individual time instances. It can be expected to result in an algorithmically weak coupling between the TSM time instances, since the Gauss–Seidel/Jacobi coupling would be on the outermost level of iteration. In order to take advantage of the above-mentioned block-matrix capabilities of Spliss, the element blocks can be extended to contain all the time instance per spatial element of the mesh, which are coupled via the matrix $\mathbf{D}$. Accordingly, the inner element block is resized to $(N_{ti} \cdot N_{state} \cdot N_{disc})^2$ to host all time instances of the element. For the the time being, the maximum default block size is limited to 1024. The resulting pattern can be seen in the right side of Figure 1 (also visualized for one state variable with a finite-volume discretization). This is expected to result in a much tighter algorithmic coupling of time

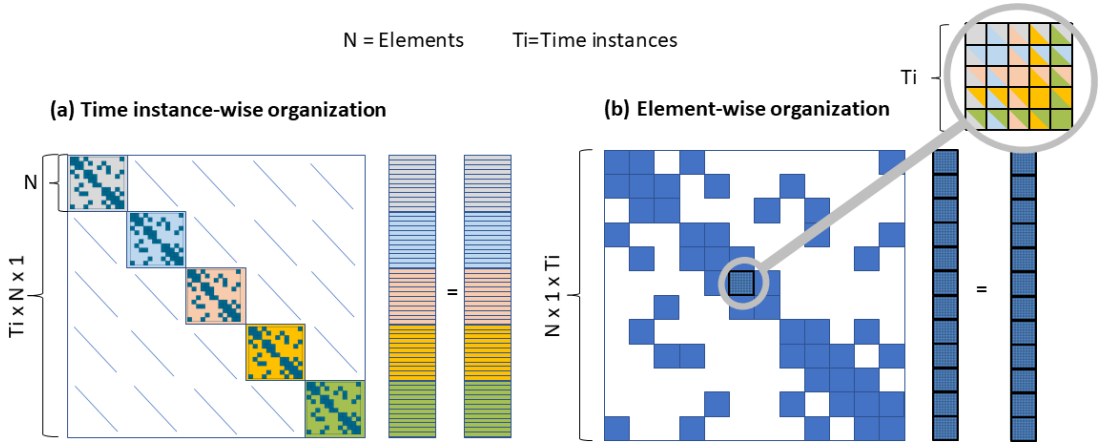


Figure 1: Different organizations of DOFs for TSM. For simplicity, only one DoF per element is depicted here. The TSM equation system is organized time-instance wise on the left-hand side, whereas the DoFs are reorganized element-wise on the right-hand side bundling all time instances per element.

instances and allows us to perform a direct inversion of the full time-spectral/spatial element blocks for the numerical solution of the TSM problem. In this paper, we have extended the method to also allow for time-spectral/spatial blocks of lines-of-elements in the spatial mesh, which are inverted by the Thomas algorithm. Note that we assemble and compute of the physical problem in a time-instance-wise approach in the CFD solver, e.g. to compute spatial fluxes, to apply boundary conditions and to set flow values. With these definitions, we use the CODA matrix constructor to reorganize the problem element-wise – cf. Figure 1, right – such that the information for all time instances is bundled per TSM element block. All TSM extension of CODA supports a fully hybrid, parallel execution with *OpenMP* and *Open MPI*.

4 NUMERICAL EXPERIMENTS

Here we present results of rigid-body motion simulations conducted with our TSM implementation. We present several results for a 2D *RAE*2822 airfoil and show the first results for the Common Research Model (DPW5-CRM based on [13]), both undergoing a prescribed, plunging motion. In both cases the Reynolds-averaged Navier–Stokes (RANS) equations were solved with the negative formulation of the Spalart–Allmaras turbulence model and a 2nd-order finite-volume discretization in space. A Roe upwind scheme was used for the approximation of the convective fluxes. All simulations have been done on the DLR supercomputer *CARA* and utilized a hybrid parallelization of *OpenMP* and *Open MPI*.

4.1 RAE2822 wing section

The RAE2822 test-case has an ambient Mach number of $Ma = 0.734$, a wing chord length based Reynolds-number of $Re = 6.5 \times 10^6$ and an angle of attack $\alpha = 2.79°$. The airfoil was moved with a forced sinusoidal rigid-body motion perpendicular to the mean flow direction

$$W(t) = a\sin\omega t + a\sin 2\omega t + a\sin 3\omega t + \cdots , \tag{17}$$

with $a = 8\%$ of the wing chord length $c = 1$ and a nondimensional passage time of $T = \frac{c}{U_\infty} = 0.86$. The simulations were conducted with different numbers of time instances (ti) on two different meshes. The coarse computational mesh consisted of 16.6 thousand mesh elements per time instance, whereas 129.6

thousand elements were used in the fine mesh per time instance.

Verification against implicit time stepping

To validate the TSM solver presented here, we computed the plunging of the RAE2822 with one harmonic using a time-accurate time stepping approach on the coarse mesh. The time stepping approach was carried out in a two-stage approach: first we compute the steady-state solution and then the plunging of the airfoil is activated. An implicit second-order diagonally-implicit Runge–Kutta (DIRK-2) scheme was used to march in time. For the TSM we used three time instances, which are necessary to resolve the problem and in order to demonstrate the consistency up to 41 time instances. Figure 2 shows the evolution of the periodic answer in the lift coefficient. After 200 time periods the oscillation in the answer of the

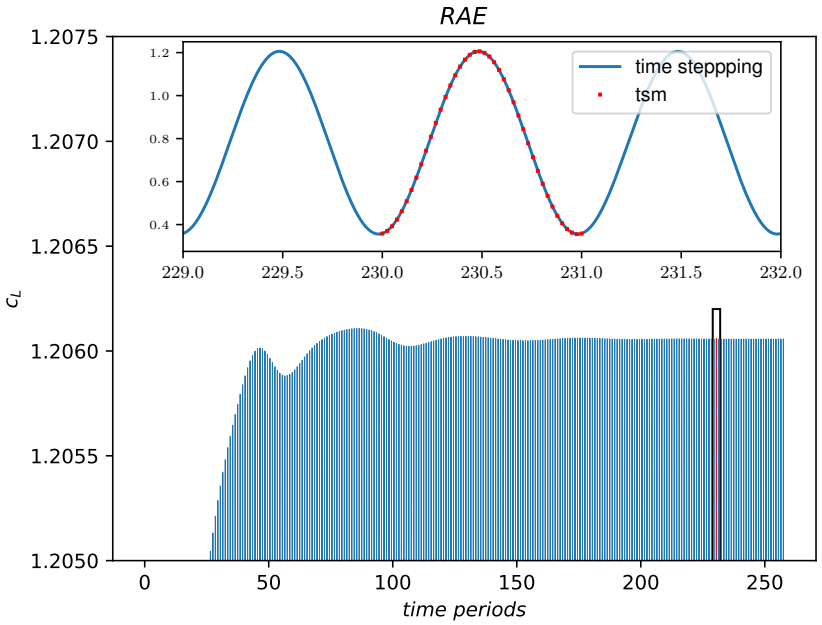
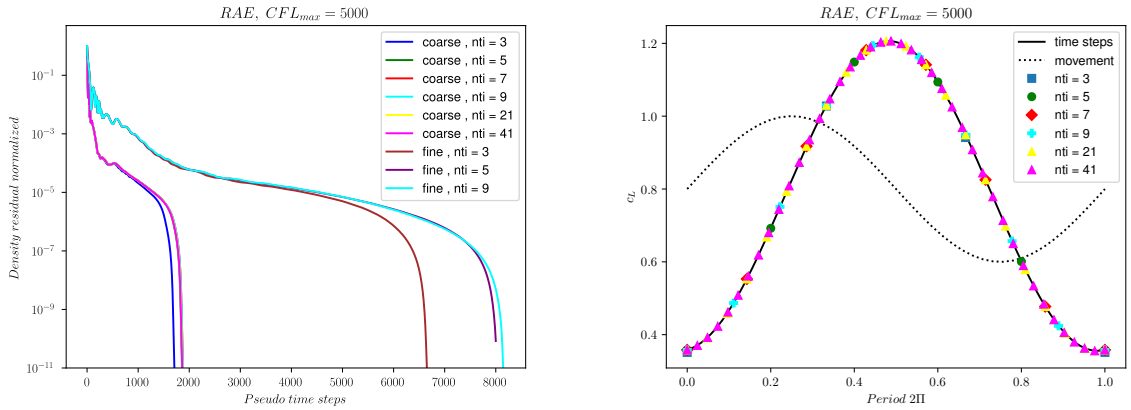


Figure 2: Evolution of lift coefficient for time stepping vs. TSM over time periods

lift coefficient are in the range of the spatial discretization accuracy ($\approx 10^{-4}$) based on a mesh-refinement study performed for the steady-state solution. The relative differences between the TSM and the time stepping prediction of the lift of the order of 10^{-5} for both 3 and 41 TSM time instances. It is interesting to note that – without any parameter optimization of TSM and time-stepping approach – the TSM wall-clock run-time with three time instances was two orders of magnitude faster than the time-stepping run-time of a similar periodic accuracy. However, it must be kept in mind that this is mainly attributed to the simulation time required to reach a periodic response. We have not computed such references for setups with more harmonics nor finer meshes yet, due to the required computational resources. The left plot of Figure 3 shows the convergence history of the density residual for the simulation with one harmonic. It was normalized by the initial value to show the reduction in magnitude. The right plot shows the resulting lift coefficient for the coarse mesh. We need three time instances to properly represent a single harmonic motion. This is confirmed in on the right-hand side of Figure 3: with the solid line representing the

time stepping reference, three time instances in the TSM reproduce the time-stepping curve. Obviously, using more time instances does not improve the quality of the TSM prediction. It is interesting to note that the number of required implicit-Euler iterations to reach a TSM residual convergence of 10 orders of magnitude does hardly depend on the number of time instances (3 to up to 41 considered here). This indicates that the strong coupling between the time instances achieved by the element-blocked nested solution strategy effectively payed off here. On the fine mesh, we observe some dependency on the number of time instances. We think that more iterations would be required on the inner preconditioner iteration levels of the solver stack to eliminate the dependency on the number of time instances. This effect is less pronounced with an increased number of harmonics.



(a) Convergence of normalized density TSM residuals (b) Lift coefficients on coarse mesh and wing movement

Figure 3: TSM results for a forced plunging oscillation of a single harmonic

Implicit TSM solution with nested solver-stacks

In order to investigate the solution behaviour and efficiency of the TSM CFD method, we set up different solver stacks enabled by the modular software implementation and compared the overall run-time. We considered the above-mentioned 2D RAE2822 configuration with one harmonic oscillation of the airfoil and three time instances to resolve the first harmonic. In general, *Spliss* [8] allows arbitrarily deep nestings of linear solvers and preconditioners within the nonlinear implicit-Euler solution procedure. However, in this study, we limited the number of nesting levels in the linear solver to three. On the outermost iteration level, we deploy Block-Jacobi (Jacobi), Block Gauss–Seidel (GS), Generalized Minimal RESidual (GMRES) or BIConjugate Gradient Stabilized (BICGS) method. On the intermediate iteration level, we took into consideration either none, Element Block-Jacobi (Jacobi) or Element Block Gauss–Seidel (GS) procedures. For the innermost (preconditioning) iteration level, we applied a direct solution method embedded in the block-Jacobi/Gauss–Seidel sweeps to invert the main block diagonal of the block-sparse system. Note that the block-matrix coefficients on the main diagonal either represent the full time-spectral/spatial element blocks, or a generalized matrix-block of lines (in the boundary layer) of time-spectral/spatial elements geometrically found in the spatial mesh. The lines inversion (LI) was achieved by means of a Thomas Algorithm. In this numerical experiment, a total of 24 different

configurations were computed to compare the run-times in wall-clock time. For all cases, the pseudo-time step of the implicit-Euler method was limited to 5000 with a CFL-ramping applied in the transient phase of the iteration. All runs were carried out with up to 1000 linear iterations until the linear residual was reduced by six orders of magnitude per pseudo-time step. The nonlinear implicit-Euler scheme was iterated until the L2-norm of the overall TSM residual has was reduced by 12 orders of magnitude. Figure 4 shows the run-times of the different solver configurations considered in this study. The run-times are normalized by the fastest configuration (GMRES-Jacobi-LI). Note that only a subset of the possible

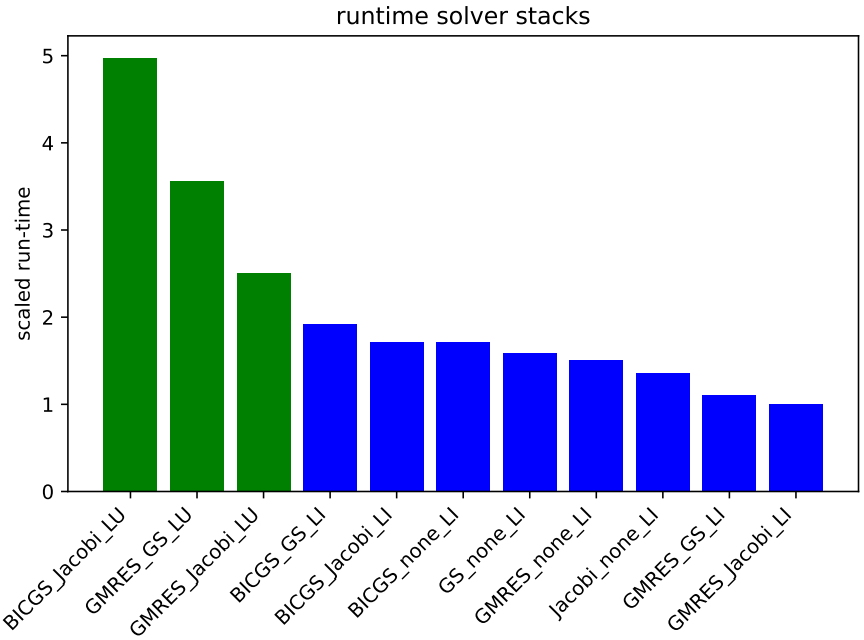


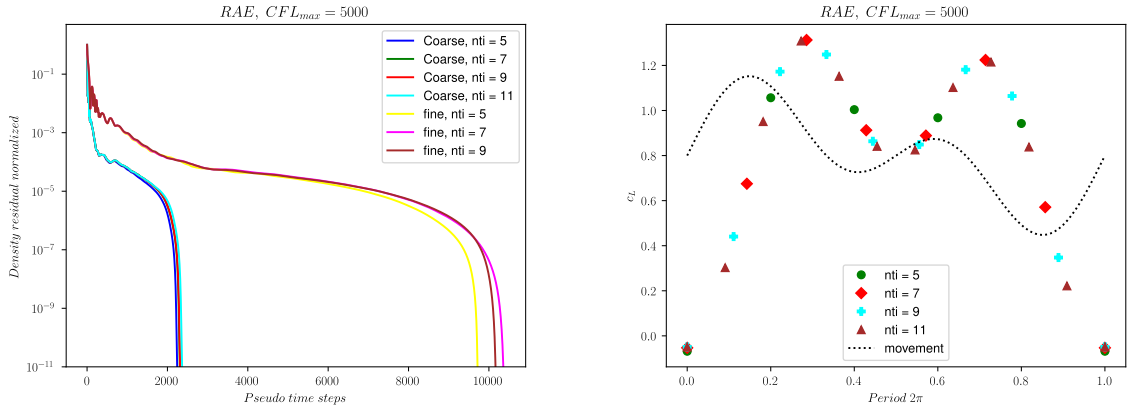
Figure 4: Run-time comparison of different solver stacks for one harmonic with three time instances. Green bars for direct inversion of TSM elements (LU), blue bars for direct inversion of lines-of-TSM-elements.

solver combinations is provided. All combinations not present in the plot either resulted in a scaled run-time higher than 5 or stagnated in conjunction with the given solver parameters. It is interesting to note that only in conjunction with Line Inversion (blue) set as preconditioner, which is using bigger blocks also containing off-diagonal matrix coefficients along the directly inverted mesh lines, two-level solvers reached the targeted convergence goal. A rather poor convergence performance was observed using BICGS methods, which did not reach the goal in combination with many solver stacks. Using GMRES with a block-Jacobi on the intermediate iteration level resulted in the best convergence for different pre-conditioning methods. Mind that two different direct solvers were used in this study to carry out the TSM element-block inversions, being an LU decomposition and a straightforward direct block-inversion approach offered by the *Spliss* library. We observed slight run-time differences between the direct LU and the direct block-inversion method. So far, we did not systematically investigate the run-time effects of the two direct block-solvers. However, both LU and the direct block-inversion approach compute an exact block-inversion on the innermost level, so that the behaviour of the outer levels should not be

affected from an algorithmic point-of-view. For this particular test-case, the best solver hierarchy turned out to be GMRES on the outer level, preconditioned by block-Jacobian on the intermediate level and a direct time-spectral/spatial line inversion on the innermost level. It should be mentioned, that the TSM Line Inversion method comes with an overhead in both memory consumption and run time compared to the direct solvers applied to the TSM elements alone. In case of a line-length of one TSM element – e.g. in areas of the mesh where the search for lines was not successful – the TSM Lines Inversion approach effectively falls back to a simple TSM element Block-Inversion. Since the Line Inversion did not work reliably in the TSM extension, when we started the investigation, for consistency reasons we used an outer GMRES with inner block-Jacobian and TSM LU block decomposition for most of the presented simulations.

TSM with forced plunging oscillations of higher harmonics

Whereas we restricted ourselves to one harmonic (with three TSM time instances) in the study of nested solver stacks above, the TSM implementation is used below for periodic analyses of plunging movements with up to six harmonics in conjunction with the RAE2822 test-case on the coarse and the fine mesh. A periodic movement in two harmonics is considered first using 5 to 11 time instances in the TSM with the linear solver stack GMRES-Jacobi-LI. According to the left-hand side of Figure 5, the required number of implicit-Euler pseudo-time iterations only differed by a few iterations for different numbers of time instances. A similar behavior is observed on the fine mesh, for which a larger number of implicit-Euler steps is required, once again independent of the number of time instances. However, it has to be kept in mind that the run-time to solution increases with larger numbers of time instances as, for example, the system matrix – including the size of the TSM element blocks to be inverted – grows with the number of time instances. Moreover, the required number of inner solver iterations increases. The time-periodic solution of the lift coefficient predicted by the TSM, cf. right-hand side of Figure 5, is equivalent for all numbers of time instances. As expected, five time instances are enough to capture the periodic flow for a plunging motion in two harmonics for the 2D case at hand. Subsequently, a harmonic

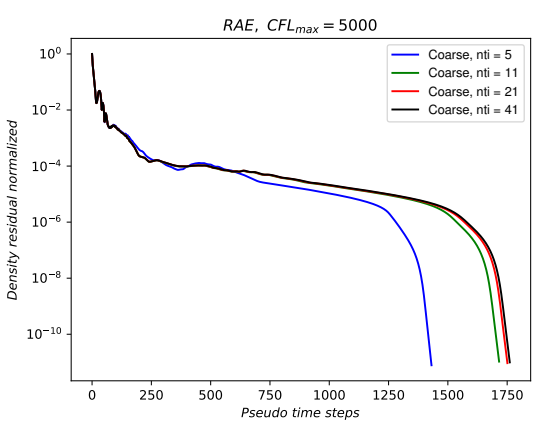


(a) Convergence of normalized TSM density residuals (b) Lift coefficients on coarse mesh and wing movement

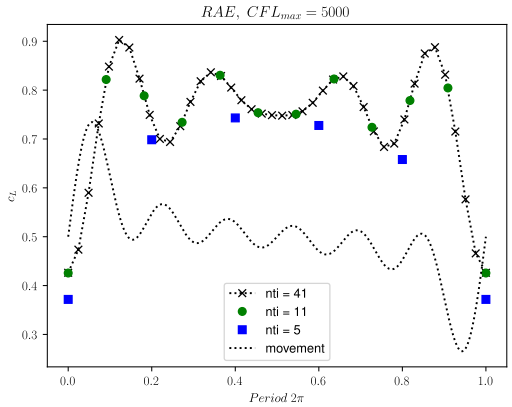
Figure 5: TSM results for forced plunging oscillations of two harmonics

plunging in 6 harmonics is computed with the TSM extension of *CODA* using 5, 11, 21 and 41 time

instances with the linear solver stack GMRES-Jacobi-LI. In order to correctly resolve six harmonics in the TSM solution, a total of (at least) 13 time instances is required. Nonetheless, we included TSM computations with five and eleven time instances in this study which can be expected to give an under-resolved TSM approximation. According to Figure 6, right-hand side, a fair TSM representation is obtained with 11 time instances, whereas with five time-instances the TSM is not able not capture the solution well. A TSM solution of 41 time instances included in the analysis is clearly over-resolved for the given problem. Figure 6, left-hand side, shows that all residuals behave similarly in combination with 11 to 41 time instances. The under-resolved computation with five time instances leads to a slight reduction in the required number of outer implicit-Euler iterations. In the time-periodic forced-motion configurations considered, it is straight-forward to set the number of time instances according to the periodic motion or excitation imposed. However, in the case of solution-dependent periodic effects like interacting vortices or boundary layers, the number of required time instances is not necessarily clear in advance. In such cases, we think that scalable TSM solvers are required, cf. [1], which allow for an efficient and robust computation of problems with a large number of harmonics to make sure that all the periodic features in the flow are sufficiently captured.



(a) Convergence of normalized TSM density residuals



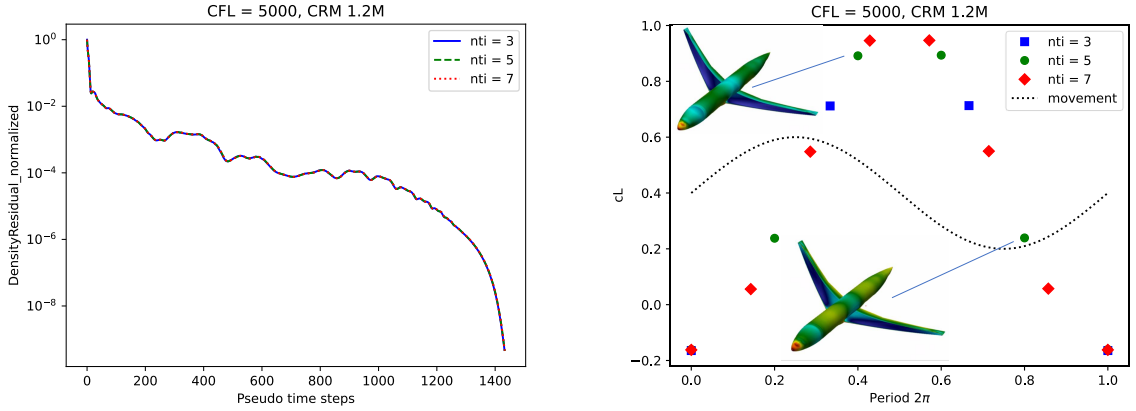
(b) Periodic lift coefficients on coarse mesh and plunging movement

Figure 6: TSM results for forced plunging oscillation of six harmonics

4.2 DPW-5 Common Research Model

To show the applicability of the TSM extension of the baseline CFD solver, a forced rigid-body plunging motion was imposed to the *NASA* Common Research Model (DPW5-CRM) [13] that was originally defined in steady-state flow at $\alpha = 2.209^\circ$ angle of attack, a Mach number of $Ma = 0.85$ and a wing Reynolds number of $Re = 5 \times 10^6$. We applied a forced plunging like for the RAE2822, limited to the first harmonic in this case. We applied an amplitude $a = 0.04\%$ of the wing chord length $c = 275.8$ and a nondimensional passage time of $T = \frac{c}{U_\infty} = 1.05$. The case was computed with 3, 5 and 7 time instances for the fully unstructured mesh configuration consisting of 1.2×10^6 cells. The solver settings used in the RAE2822 case were reapplied here, with a targeted relative reduction of the density residual of 10 orders of magnitude. Figure 7 indicates that the periodic prediction of the lift coefficient does not depend on the

number of time steps (or harmonics) for this case with the forced-motion only in the first harmonic. It is interesting to note, that the number of pseudo-time steps needed are identical for all three cases. This feasibility study shows that the implicit solution procedure with nested solver stacks is able to converge the coupled TSM equation systems for a fully-turbulent 3D test case in transonic conditions. However, only the coarse mesh configuration of 1.2×10^6 mesh nodes was considered here and further studies have to be carried out for a better understanding of the solution behavior and the relevant parameters.



(a) Convergence of density residuals, normalized by initial value

(b) Lift coefficients with an illustration of the plunging movement

Figure 7: First results for the Common Research Model (DPW5-CRM)

5 Conclusions

A TSM extension to the CFD software CODA was presented in this paper. In order to solve large TSM problems, the method allows to set up nested solver stacks consisting of implicit-Euler pseudo-time stepping in combination with different block-preconditioned Krylov schemes. The blocking is organized element-wise such that all time instances can be bundled together to apply a direct inversion of time-spectral/spatial blocks within the iterative solution procedures. The element-wise blocking of the TSM equations was generalized here to also support a direct inversion of time-spectral/spatial lines-of-elements blocks by the Thomas algorithm. Numerical experiments were carried out for an RAE2822 airfoil in fully-turbulent flow on two different meshes with different numbers of harmonics resolved. For these cases we studied different combinations of solvers and preconditioners in the solver stack. The results obtained here indicate that the more advanced solution schemes that include three levels of linear solvers and a line inversion of the time-spectral/spatial blocks are superior in run-time performance.

With the tight coupling of TSM time instances, it was observed that the required number of nonlinear solver iterations did only weakly depend on the number of harmonics resolved in the system. A 3D feasibility study was conducted for the DPW5-CRM test case. It demonstrates that the approach is capable of solving fully-turbulent 3D problems at high Reynolds numbers in transonic flow. Further investigations and code optimizations need to be carried out to leverage the potential of the method for larger time-periodic 3D problems. Taking advantage of the modularity of the underlying CFD software CODA, we intend to integrate the additional TSM capabilities into a framework approach for high-

fidelity multidisciplinary analyses and optimizations to efficiently address time-periodic problems.

Acknowledgements

The authors gratefully acknowledge the scientific support and HPC resources provided by the German Aerospace Center (DLR). The HPC system CARA is partially funded by "Saxon State Ministry for Economic Affairs, Labour and Transport" and "Federal Ministry for Economic Affairs and Climate Action".

References

[1] Nathan L. Mundis and Dimitri J. Mavriplis. "Toward an optimal solver for time-spectral fluid-dynamic and aeroelastic solutions on unstructured meshes". In: *Journal of Computational Physics* 345 (2017), pp. 132–161. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2017.04.067.

[2] Arathi Gopinath and Antony Jameson. "Time Spectral Method for Periodic Unsteady Computations over Two- and Three- Dimensional Bodies". In: *43rd AIAA Aerospace Sciences Meeting and Exhibit*. 2005, p. 1220. DOI: 10.2514/6.2005-1220.

[3] Arathi Gopinath and Antony Jameson. "Application of the Time Spectral Method to Periodic Unsteady Vortex Shedding". In: *44th AIAA Aerospace Sciences Meeting and Exhibit*. 2006, p. 449. DOI: 10.2514/6.2006-449.

[4] Reik Thormann and Markus Widhalm. "Linear-Frequency-Domain Predictions of Dynamic-Response Data for Viscous Transonic Flows". In: *AIAA Journal* 51.11 (2013), pp. 2540–2557. DOI: 10.2514/1.J051896.

[5] Aloïs Bissuel et al. "Linearized Navier–Stokes equations for aeroacoustics using stabilized finite elements: Boundary conditions and industrial application to aft-fan noise propagation". In: *Computers & Fluids* 166 (2018), pp. 32–45. ISSN: 0045-7930. DOI: https://doi.org/10.1016/j.compfluid.2018.01.011.

[6] M.S. McMullen and A. Jameson. "The computational efficiency of non-linear frequency domain methods". In: *Journal of Computational Physics* 212.2 (2006), pp. 637–661. ISSN: 0021-9991. DOI: https://doi.org/10.1016/j.jcp.2005.07.021.

[7] Kenneth C. Hall, Jeffrey P. Thomas, and W. S. Clark. "Computation of Unsteady Nonlinear Flows in Cascades Using a Harmonic Balance Technique". In: *AIAA Journal* 40.5 (2002), pp. 879–886. DOI: 10.2514/2.1754.

[8] Olaf Krzikalla et al. "Spliss: A Sparse Linear System Solver for Transparent Integration of Emerging HPC Technologies into CFD Solvers and Applications". In: *22nd STAB/DGLR Symposium on New Results in Numerical and Experimental Fluid Mechanics XIII*. Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Springer International Publishing, July 2021, pp. 635–645. DOI: 10.1007/978-3-030-79561-0_60.

[9] Johannes Wendler et al. "Accelerating the FlowSimulator: Mixed Precision Linear Solvers in Industrial Grade CFD". In: *9th European Congress on Computational Methods in Applied Sciences and Engineering*. June 2024.

[10] Arne Rempke. "Deformation of CFD Meshes with Anisotropic Cells in a Viscous Boundary Layer Using Line-Implicit Methods". In: *23rd STAB/DGLR Symposium on New Results in Numerical and Experimental Fluid Mechanics XIV*. Ed. by Andreas Dillmann et al. Vol. 154. Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Springer Nature Switzerland AG, Sept. 2023, pp. 273–283. DOI: 10.1007/978-3-031-40482-5_26.

[11] Tobias Leicht et al. "DLR-Project Digital-X - Next Generation CFD Solver 'Flucs'". In: *Deutscher Luft- und Raumfahrtkongress 2016*. Feb. 2016.

[12] Immo Huismann, Stefan Fechter, and Tobias Leicht. "HyperCODA – Extension of Flow Solver CODA Towards Hypersonic Flows". In: *New Results in Numerical and Experimental Fluid Mechanics XIII*. Ed. by Andreas Dillmann et al. Cham: Springer International Publishing, 2021, pp. 99–109. ISBN: 978-3-030-79561-0. DOI: 10.1007/978-3-030-79561-0_10.

[13] NASA. *Common Research Model*. http://commonresearchmodel.larc.nasa.gov. NASA, 2012.