LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-JRNL-820515

Time-periodic steady-state solution of fluid-structure interaction and cardiac flow problems through multigrid-reduction-in-time

A. Hessenthaler, R. D. Falgout, J. B. Schroder, D. Nordsletten, O. Roehrle

March 15, 2021

Computer Methods in Applied Mechanics and Engineering

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Time-periodic steady-state solution of fluid-structure interaction and cardiac flow problems through multigrid-reduction-in-time[1]

Andreas Hessenthaler[a,b,*], Robert D. Falgout[c], Jacob B. Schroder[d], Adelaide de Vecchi[e], David Nordsletten[e,f], Oliver Röhrle[a,b]

[a] *Institute for Modelling and Simulation of Biomechanical Systems, University of Stuttgart, Pfaffenwaldring 5a, 70569 Stuttgart, Germany*
[b] *Stuttgart Center for Simulation Technology, University of Stuttgart, Pfaffenwaldring 5a, 70569 Stuttgart, Germany*
[c] *Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA 94551*
[d] *Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM 87131*
[e] *School of Biomedical Engineering and Imaging Sciences, King's College London, 4th FL Rayne Institute, St. Thomas Hospital, London, SE1 7EH*
[f] *Department of Biomedical Engineering and Cardiac Surgery, University of Michigan, NCRC B20, 2800 Plymouth Rd, Ann Arbor, 48109*

Abstract

In this paper, a time-periodic MGRIT algorithm is proposed as a means to reduce the time-to-solution of numerical algorithms by exploiting the time periodicity inherent to many applications in science and engineering. The time-periodic MGRIT algorithm is applied to a variety of linear and nonlinear single- and multiphysics problems that are periodic-in-time. It is demonstrated that the proposed parallel-in-time algorithm can obtain the same time-periodic steady-state solution as sequential time-stepping. It is shown that the required number of MGRIT iterations can be estimated a priori and that the new MGRIT variant can significantly and consistently reduce the *time-to-solution* compared to sequential time-stepping, irrespective of the number of dimensions, linear or nonlinear PDE models, single-physics or coupled problems and the employed computing resources. The numerical experiments demonstrate that the time-periodic MGRIT algorithm enables a greater level of parallelism yielding faster turnaround, and thus, facilitating more complex and more realistic problems to be solved.

Keywords: Time-periodic parallel-in-time solver, Multigrid-reduction-in-time (MGRIT), Speedup, Fluid-Structure Interaction, Analytic Solutions, Cardiac flow

1. Introduction

Modern computer architectures enable complex and detailed numerical investigations of various problems of interest in science and engineering. Many of these problems include spatial and temporal components that need to be sufficiently resolved to answer important research questions. A significant subset of such models share the presence of periodicity in the time dimension. For example, a simulation model of the human heart may model the time-periodic contraction of the heart tissue due to electrophysiological signals, while a flow model governs the time-periodic evolution of fluid flow in the heart's chambers due to influx, e.g., from the pulmonary veins and the ejection of blood through the aortic valve.

While parallelization techniques (such as the domain decomposition method, e.g., [1, 2, 3]) for reducing the time-to-solution for such numerical models are well-known, well-established and (for the most part)

[1] This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344, LLNL-JRNL-820515.

[*] Corresponding author.
Email address: andreas.hessenthaler@gmail.com (Andreas Hessenthaler)

straightforward to apply for the space dimension, parallelization techniques for the time dimension are less wide-spread and less well-known as their spatial counterpart, especially in the application-driven research communities. With the first explorations into time-parallelism in the work of Nievergelt in 1964 [4], so-called parallel-in-time methods have seen a rise and fall in interest for the decades to follow, e.g., see the excellent review articles by Gander [5] and Ong and Schroder [6]. More recently, the introduction of massively parallel hardware has led to situations where spatial speedups saturate before all available hardware can be employed, such that interest in parallel-in-time algorithms has started increasing steadily.

Nowadays, time-parallel algorithms have seen uptake in a wider range of research fields. Many such parallel-in-time methods exist, e.g., [7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18] and researchers continue to develop and propose new techniques. On the other hand (and similar to numerical algorithms that fall into other categories than time-parallelism), a growing number of researchers develop convergence analysis for their algorithms to enable well-informed parameter choices and thus improve parallel efficiency, e.g., see [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29] and many others.

The number of parallel-in-time techniques that were successfully applied to time-periodic problems is smaller; however, a time-periodic waveform relaxation-type method was introduced in the work of Vandewalle and Piessens [30]. Other algorithms fall into two principal categories: algorithms that employ a periodic-in-time coarse-problem, e.g., [20, 31, 32, 33], or a non-periodic-in-time coarse-problem, e.g., [34, 31, 35].

In this work, we extend the multigrid-reduction-in-time (MGRIT) algorithm for the class of applications that exhibit periodicity in the time dimension using a non-periodic coarse-problem approach. While the proposed method is agnostic to the underlying PDE, the applications in this work stem from the biomedical engineering field. In Section 2, we commence by introducing the (non-periodic) MGRIT algorithm and develop its time-periodic variant. In Section 3, we introduce three application classes covering linear and nonlinear flow and fluid-structure interaction problems in two and three space dimensions. We assess the time-periodic MGRIT algorithm for each of these applications in Section 4, where we compare the time-periodic steady-state solution obtained by using sequential time-stepping and time-periodic MGRIT. We further explore the cost-to-accuracy ratio using analytic solutions and demonstrate the achieved runtime reduction using time-periodic MGRIT (up to 37x speedup) over a range of small- to medium-sized parallel computing hardware (up to 256 processors). We conclude by discussing the benefits of the proposed algorithm and its limitations in Section 5.

2. Methodology

In this section, we discuss the sequential time-stepping solution of a general time-dependent PDE and introduce MGRIT as a parallel-in-time solver that solves the same space-time problem iteratively. We then introduce an extension to standard MGRIT for the solution of general nonlinear systems of PDEs that exhibit time-periodic behavior. By adapting the MGRIT solution process to the time-periodic property of the considered system, we propose to exploit this property at the solver level as a means to exploit the full potential for parallel speedup. In general, MGRIT is a true multilevel method (compare with, e.g., spatial and algebraic multigrid methods [36, 37, 38]). For ease of presentation, however, we restrict ourselves to the two-level setting. Nevertheless, the time-periodic MGRIT algorithm's design encompasses the true multilevel setting [39].

2.1. Sequential time-stepping

First, consider the general form of a time-dependent PDE in one spatial dimension:

$$f(x,t;u,\partial u/\partial x,\partial u/\partial t,\partial^2 u/\partial x^2,\ldots)=0 \qquad \text{for } (x,t)\in\Omega\times[0,T]. \tag{1}$$

Discretizing Equation (1) in space and time, sequential time-stepping can be written as,

$$\boldsymbol{u}_n=\boldsymbol{\Phi}\left(\boldsymbol{u}_n,\boldsymbol{u}_{n-1}\right)+\boldsymbol{g}_n \qquad \text{for } n=1,\ldots,N_0-1, \tag{2}$$

with approximation $\boldsymbol{u}_n$ at time t_n, and forcing function $\boldsymbol{g}_n$ for N_0 time points, $0 = t_0 < t_1 < \ldots < t_{N_0-1} = T$, and final time T. Here, $\boldsymbol{u}_n, \boldsymbol{g}_n \in \mathbb{R}^{N_x}$ with N_x spatial degrees-of-freedom. Further, $\boldsymbol{\Phi}$ is referred to as the *time-stepping operator*.

For linear problems, Equation (2) can be written as,

$$\boldsymbol{u}_n = \boldsymbol{\Phi}_n \boldsymbol{u}_{n-1} + \boldsymbol{g}_n \qquad \text{for } n = 1, \ldots, N_0 - 1. \tag{3}$$

Further, $\boldsymbol{\Phi}_n = \boldsymbol{\Phi}_0$, if a fixed spatial discretization and equidistant time points $t_n = n \cdot \delta_0$ are assumed for $n = 0, \ldots, N_0 - 1$ and time step size $\delta_0 = T/(N_0 - 1)$. Then, Equation (3) can be written in matrix form,

$$A_0 \boldsymbol{u} = \begin{bmatrix} I & & & & \\ -\boldsymbol{\Phi}_0 & I & & & \\ & -\boldsymbol{\Phi}_0 & I & & \\ & & \ddots & \ddots & \\ & & & -\boldsymbol{\Phi}_0 & I \end{bmatrix} \begin{bmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{u}_{N_0-1} \end{bmatrix} = \boldsymbol{g}, \tag{4}$$

with space-time matrix $A_0 \in \mathbb{R}^{N_x N_0 \times N_x N_0}$ and space-time solution vector $\boldsymbol{u} \in \mathbb{R}^{N_x N_0}$. Now, sequential time-stepping can be identified as a *block-forward solve* of (4).

2.2. Multigrid-reduction-in-time (MGRIT)

MGRIT solves (4) iteratively using a linear or near-linear number of block operations [27]. That is, MGRIT is an $O(N_0)$ method [17, 18], similar to sequential time-stepping. To achieve parallelism in the temporal domain, MGRIT introduces two[2] (or more) time grids, accompanied by a coarse-grid correction scheme based on multigrid reduction. The fine grid (or time grid on level $\ell = 0$) is composed of all time points t_n (for $n = 0, \ldots, N_0 - 1$) and the coarse grid (or time grid on level $\ell = 1$) is derived from a uniform coarsening of the fine grid, see Figure 1. The temporal coarsening factor is denoted as $m \in \mathbb{Z}^+$, such that the number of time points on the coarse grid is given by $N_1 = (N_0 - 1)/m + 1$, with corresponding time step size $\delta_1 = m\delta_0$ and time-stepping operator Φ_1. On the fine grid, time points are partitioned into F-points and C-points, where C-points refer to the time points also present on the coarse grid and F-points only exist on the fine grid, see Figure 1.

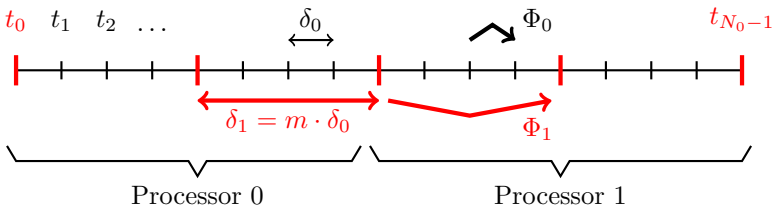


Figure 1: Two-level decomposition of the time grid with time points t_n into F-points (black, only on fine grid) and C-points (red, only on coarse-grid) with fine grid time step size δ_0 (level 0) and coarse grid time step size δ_1 (level 1). Here, the temporal coarsening factor is given as $m = 4$ and the time stepping operators are denoted as Φ_0 and Φ_1.

Instead of applying the fine-grid time-stepping operator m times per coarse-grid time step, MGRIT approximates the exact coarse-grid time-stepping operator,

$$\Phi_1 \approx \Phi_0^m, \tag{5}$$

in order to reduce complexity. A Schur complement decomposition of A_0 yields *ideal* multigrid restriction

[2]For a detailed introduction of MGRIT for an arbitrary number of time grids, the interested reader is referred to [17, 18] and the developer's manual of the XBraid library [40], an open-source implementation of the MGRIT algorithm.

and interpolation operators. That is, ideal restriction from level 0 to 1 is given as,

$$R = \begin{bmatrix} I & & & & & & & & \\ & \Phi_0^{m-1} & \Phi_0^{m-2} & \cdots & I & & & & \\ & & & & & \ddots & & & \\ & & & & & & \Phi_0^{m-1} & \Phi_0^{m-2} & \cdots & I \end{bmatrix} \in \mathbb{R}^{N_x N_1 \times N_x N_0}, \tag{6}$$

and restricts the approximate solution at C-points from the fine grid to the coarse grid. Ideal interpolation from level 1 to level 0 is given as,

$$P = \begin{bmatrix} [I, \Phi_0, \cdots, \Phi_0^{m-1}]^T & & & \\ & [I, \Phi_0, \cdots, \Phi_0^{m-1}]^T & & \\ & & \ddots & \\ & & & I \end{bmatrix} \in \mathbb{R}^{N_x N_0 \times N_x N_1}, \tag{7}$$

and interpolates the approximate solution from the coarse grid to the C-points and its neighboring F-points on the fine grid.

Note, that in general $\Phi_\ell \neq \Phi_\ell^T$ and thus $P \neq R^T$ [41, Equation (28)]. Then, the ideal[3] multigrid coarse-grid operator, RA_0P approximates the space-time matrix A_1 on level 1 [25],

$$RA_0P = R_I A_0 P \quad = \begin{bmatrix} I & & & & \\ -\Phi_0^m & I & & & \\ & -\Phi_0^m & I & & \\ & & \ddots & \ddots & \\ & & & -\Phi_0^m & I \end{bmatrix} \approx \begin{bmatrix} I & & & \\ -\Phi_1 & I & & \\ & -\Phi_1 & I & \\ & & \ddots & \ddots \end{bmatrix} = A_1 \in \mathbb{R}^{N_x N_1 \times N_x N_1} \tag{8}$$

where the first equality is given by Equation (SM1) and restriction by injection is given by the operator,

$$R_I = \begin{bmatrix} I & & & & & & & & & \\ & 0 & 0 & \cdots & I & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & 0 & 0 & \cdots & 0 & I \end{bmatrix} \in \mathbb{R}^{N_x N_1 \times N_x N_0}. \tag{9}$$

R_I in Equation (9) has a similar block structure as R in Equation 6, but with all blocks Φ_1^k (for $k = 1, \ldots, m-1$) set to zero. Thus, R_I restricts the C-points from level 0 to level 1, omitting the respective F-points.

Coarse-grid correction reduces error at C-points and is thus coupled with a process referred to as *F-relaxation* to reduce error at F-points. Algebraically, this is an application of the idempotent operator,

$$F = PR_I. \tag{10}$$

When F-relaxation does not sufficiently reduce error at F-points, a stronger relaxation scheme can be used, called FCF-relaxation. FCF-relaxation refers to subsequently applied F-, C- and F-relaxation steps. C-relaxation updates every C-point based on taking one time step from the corresponding previous F-point. Algebraically, this corresponds to an application of $C = I - R_I^T (R_I A_0 R_I^T)^{-1} R_I A_0$. FCF-relaxation can then be written as,

$$FCF = P(I - RA_0P)R_I. \tag{11}$$

[3]The coarse-grid operator RA_0P is ideal since one step on the coarse grid is performed by m fine grid time steps, which yields the exact solution with respect to the fine grid. Without approximating the operator RA_0P, the method would be as expensive, which motivates the approximation in Equation (8).

F-relaxation reduces error at the F-points following a given C-point, whereas C-relaxation reduces error at the C-point following the previous F-point; thus, both can be seen as block Jacobi applied to the respective rows in A_0.

Now that we have the basic ingredients for MGRIT, we can describe a standard two-level cycle in the following pseudocode:

Algorithm 1 Pseudocode for two-level MGRIT algorithm, adapted from [25].

1: **repeat**
2: Relax on $A_0\boldsymbol{u} = \boldsymbol{g}$ using F-/FCF-relaxation with $\boldsymbol{\Phi}_0$. ▷ In parallel
3: Compute coarse-grid residual $\boldsymbol{r}^{\Delta} = R(\boldsymbol{g} - A_0\boldsymbol{u})$.
4: Solve coarse grid correction problem $A_1\boldsymbol{e}^{\Delta} = \boldsymbol{r}^{\Delta}$ using $\boldsymbol{\Phi}_1$.
5: Correct solution at fine-grid C-points with $\boldsymbol{e}^{\Delta}$. ▷ In parallel
6: Update the solution at the F-points with F-relaxation and $\boldsymbol{\Phi}_0$. ▷ In parallel
7: **until** norm of residual is small enough

It is clear from the pseudocode in Algorithm 1, that more work needs to be done compared to sequential time-stepping; however, this work can be done in parallel, which highlights MGRIT's potential for parallel speedup.

Remark 1. *MGRIT does not require the same time-stepping operator on the fine and coarse time grids. In fact, employing a* cheaper *(w.r.t. computational work) time-stepping operator and / or spatial solver can improve parallel performance, e.g., see [42, 43].*

Remark 2. *MGRIT is neither restricted to the linear PDE case nor to simple two-level cycles. Similar to other multigrid methods, MGRIT can accomodate various cycling strategies, such as V- and F-cycles. Multilevel V-cycles are achieved by calling the algorithm recursively in Step 4 in Algorithm 1. Multilevel F-cycles can be achieved by performing (at least) one V-cycle as postrelaxation at each level [44]. Similar to other multigrid reduction problems, solving the nonlinear setting is achieved by solving Equation* (2) *and employing a full approximation storage approach. For more details, refer to [17, 18].*

Remark 3. *To simplify terminology, we refer to the exact solution as the solution obtained from sequential time-stepping on the fine grid. Note, that sequential time-stepping can be identified as a direct solution approach with respect to Equation* (4)*. In that regard, MGRIT can be interpreted as an interative solver, which obtains the same solution as sequential time-stepping on the fine grid.*

2.3. The XBraid library

For the work presented in this article, the open-source implementation XBraid [40] (release v2.3.0) was employed and extended, e.g., to implement the time-periodic MGRIT variant proposed in the following. To successfully link an existing simulation package with XBraid, the user has to implement a number of wrapper routines. The wrapper steers and calls the simulation code to perform a number of basic tasks (e.g., norm computation, data I/O, etc.) and to perform time integration. Time integration routines from the simulation code are exposed through the so-called XBraid *step function* that represents the application of the time stepping operator $\boldsymbol{\Phi}_\ell$, see Equation (2) and Equation (3).

The *step function* can be summarized as the following pseudocode:

Algorithm 2 Pseudocode for XBraid's *step function* on level ℓ.

1: **function** STEP_FUNCTION($\boldsymbol{u}(\cdot, t)$, t, δ_ℓ)
2: Set $\boldsymbol{u}(\cdot, t)$, time t and time step size δ_ℓ
3: Compute solution $\boldsymbol{u}(\cdot, t+\delta_\ell) = \boldsymbol{\Phi}_\ell\left(\boldsymbol{u}(\cdot, t+\delta_\ell), \boldsymbol{u}(\cdot, t); t, t+\delta_\ell\right) + \boldsymbol{g}(\cdot, t+\delta_\ell)$
4: **return** $\boldsymbol{u}(\cdot, t+\delta_\ell)$

A realistic *step function* may have many more steps or customizations depending on the considered setting; however, Algorithm 2 highlights the fundamental algorithmic steps. Overall, the required implementation overhead to enable parallel-in-time integration through MGRIT is relatively low. For example, the finite element package CHeart [45, 46] has about 98500 lines of Fortran code (includes comments), whereas the wrapper routines only amount to about 2050 lines of Fortran code (includes comments).[4]

2.4. Extension for time-periodic problems

In this section, we describe how the MGRIT algorithm can be extended to solve time-periodic problems more efficiently by making the fine grid periodic.[5] Instead of solving the problem in a traditional manner using sequential time-stepping and simulating q cycles (i.e. $t \in [0, qT]$) to achieve a time-periodic steady-state, the proposed time-periodic MGRIT algorithm approximates the space-time solution over *one* periodic cycle only (i.e. $t \in [0, T]$).

First, an initial space-time guess is obtained (XBraid iteration 0) by performing sequential time-stepping on the coarse-grid,[6] which can be configured in the XBraid library (refer to the *skip-first-down* option in the XBraid manual [40]). XBraid cycling then continues as normal. Convergence to the time-periodic steady-state is achieved by *updating* the initial condition at $t = 0$ on the *fine grid*, whenever a new value is computed / available at $t = T$ on the *fine grid*. That is, the step function from Algorithm 2 is used on the coarse grid, whereas a modified step function is employed on the fine grid:

Algorithm 3 Pseudo-code for XBraid's *step function* for the time-periodic case on level 0.

1: **function** STEP_FUNCTION($\boldsymbol{u}(\cdot, t)$, t, δ_0)
2: Set $\boldsymbol{u}(\cdot, t)$, time t and time step size δ_0
3: **if** time $t = 0$ and iteration $i > 0$ **then**
4: **if** initial condition at $t = 0$ not converged **then**
5: Receive update as $\boldsymbol{u}(\cdot, t)$ using blocking MPI_Recv
6: **if** iteration $i = 1$ and using F-relaxation **then**
7: Discard first update and receive second update as $\boldsymbol{u}(\cdot, t)$ using blocking MPI_Recv
8: Compute solution $\boldsymbol{u}(\cdot, t + \delta_0) = \boldsymbol{\Phi}_0 \left(\boldsymbol{u}(\cdot, t + \delta_0), \boldsymbol{u}(\cdot, t); t, t + \delta_0\right) + \boldsymbol{g}(\cdot, t + \delta_0)$
9: **if** time $t + \delta_0 = T$ **then**
10: **if** initial condition at $t = 0$ not converged **then**
11: **if** $\|\boldsymbol{u}(\cdot, T) - \boldsymbol{u}(\cdot, 0)\|_2 < tol$ **then**
12: Set initial condition as converged
13: Send update of $\boldsymbol{u}(\cdot, T)$ using nonblocking MPI_iSend
14: **return** $\boldsymbol{u}(\cdot, t + \delta_0)$

In the following, we provide a more detailed description of the modified fine grid step function for the time-periodic case to accompany the pseudocode in Algorithm 3: First, if $t = 0$ and the initial condition at $t = 0$ is not converged, an update of the initial condition is received in Step 5. During the first iteration and if F-relaxation is employed, a second update of the initial condition is received in Step 6. Then, perform time-integration in Step 8 as usual. If the current time equals the cycle time, i.e. $t + \delta_0 = T$, check if the initial condition has converged (Step 11) and send an update in Step 13, if required. Finally, return the new value in Step 14.

Remark 4. *Since the time-periodic MGRIT algorithm starts a forward-solve on the coarsest-grid, no updates are available for iteration* 0 *in Step 3 of Algorithm 3. In the case that no time-parallelism is employed (i.e.*

[4]Note, that additional changes or additions to the CHeart source code (e.g., I/O in the time-parallel setting) have been made since this computation.

[5]By modifying the algorithm only on the fine grid, it naturally extends to the true multilevel setting.

[6]The cost for obtaining the initial space-time guess for a linear problem is roughly one m^{th} of the cost of running sequential time stepping on the fine grid, where m is the temporal coarsening factor.

one processor in time), the initial condition is directly updated from the value computed for $t = T$ without MPI communication.

Remark 5. *In iteration* 1*, processor* 0 *waits for two messages in the case of F-relaxation, see Algorithm 3, Step 6. Preliminary experiments showed that this was required to achieve a robust residual reduction with a convergence factor smaller than one. The following iterations, however, proceed similarly for F- and FCF-relaxation and ensure that updates are incorporated as they become available.*

Remark 6. *Step 11 of Algorithm 3 allows for an approximate convergence to the time-periodic steady-state with a given tolerance tol. To achieve this, a* callback *function was implemented in XBraid that lets the simulation code compute the size of the jump and therefore communicate to XBraid whether it may terminate (initial condition converged) or not (initial condition* not *converged). In this work, convergence of the initial condition is evaluated using an absolute convergence criterion, however, a relative tolerance could be similarly appropriate.*

Remark 7. *A similar idea to exploit time-periodicity was applied for the Parareal algorithm [19, 31]. In our approach, however, we only require an* approximate *update of the initial condition. In this work, the solution at $t = 0$ is considered converged (indicating a time-periodic steady state) if the absolute difference of subsequent approximations is less than 10^{-10} in the Euclidean norm.*

In the following, we only consider two-level MGRIT using F- and FCF-relaxation, a temporal coarsening factor of $m = 8$ and an initial space-time guess based on sequential time stepping on the coarse grid, unless noted otherwise.

3. Numerical experiments

In this section, we introduce three application classes for which we will assess the proposed time-periodic MGRIT algorithm in Section 4. To assess the new algorithm for a simple test problem, we commence with a simple linear PDE model defined over a rigid two-dimensional flow domain (resembling flow in the ascending aorta) in Section 3.1. We proceed by adding complexity in a multiphysics problem in Section 3.2: linear and nonlinear fluid-structure interaction models in a 2D channel and 3D vessel, where analytic solutions are available to highlight the space-time error reduction properties of the algorithm. We complete the assessment by applying a nonlinear flow model over a realistic left atrium / left ventricle (LA / LV) geometry taken from 3D MRI data in Section 3.3, to demonstrate that the new algorithm is applicable to more complex applications without further modifications.

3.1. Simplified flow in the ascending aorta

Consider flow in the ascending aorta in a simplified two-dimensional geometry, see Figure 2: Pulsatile blood flow is entering the domain from the left ventricle. The blood flows past two rigid valves,[7] the aortic sinus (anatomic dilations of the ascending aorta) and enters the ascending aorta. The outflow domain is slightly extended where the aortic arch would be located.

Further, consider the Stokes[8] equations for incompressible Newtonian flow:

$$\rho_f \partial_t \boldsymbol{v}_f - \nabla \cdot \left[\mu_f \left(\nabla \boldsymbol{v}_f + \nabla^T \boldsymbol{v}_f\right) - p_f \boldsymbol{I}\right] = 0 \qquad \text{in } \Omega_f, \tag{12}$$

$$\nabla \cdot \boldsymbol{v}_f = 0 \qquad \text{in } \Omega_f, \tag{13}$$

[7] Here, we simplify the problem by making the geometry rigid to be able to test the algorithm for a simple flow model over a complex geometry.

[8] For this type of application, a simple Stokes flow model may not be the most accurate choice. This choice, however, was motivated by the aim of investigating the time-periodic MGRIT algorithm in a number of different settings, e.g., by starting off from a simple linear model and adding complexity in a step-by-step manner, to ultimately highlight that the algorithm performs well in various settings.

with velocity $\boldsymbol{v}_f$, pressure p_f, density $\rho_f = 10^{-3}\ g/mm^3$ and viscosity $\mu_f = 0.04\ g/(mm \cdot s)$. Pulsatile parabolic inflow with cycle length $T = 1.024\ s$ is prescribed at the inflow,

$$\boldsymbol{v}_f = \begin{bmatrix} 200 \\ 0 \end{bmatrix} \frac{mm}{s} \left(0.5 + 0.5 \cdot \cos\left(\frac{2\pi t - \pi}{T}\right)\right) \left[1 - \left(\frac{y - 0.685\ mm}{14.415\ mm}\right)^2\right], \tag{14}$$

and pulsatile outflow[9] is prescribed at the left and right coronary arteries,

$$\boldsymbol{v}_f = \begin{bmatrix} 0 \\ 10 \end{bmatrix} \frac{mm}{s} \left(0.5 + 0.5 \cdot \cos\left(\frac{2\pi t - \pi}{T}\right)\right) \operatorname{sign}(y). \tag{15}$$

Zero Neumann conditions are prescribed at the outflow and zero Dirichlet conditions are prescribed at all other boundaries. The initial condition is set as zero.

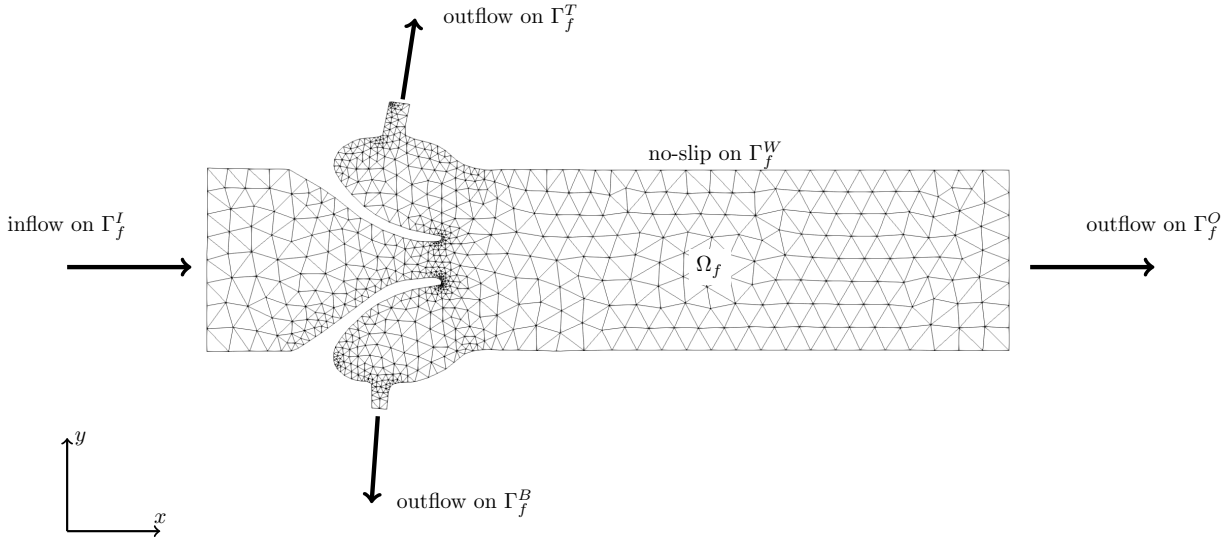


Figure 2: Simplified ascending aorta geometry in two dimensions with rigid maximally dilated valves. Inflow occurs from the left ventricle on the left. Minor outflows are observed toward the left and right coronary arteries at the top and bottom, and main outflow toward the aortic arch on the right.

3.1.1. Space-time discretization and weak formulation

The fluid domain Ω_f is discretized using finite elements with 1388 inf-sup stable $\mathbb{P}^2 - \mathbb{P}^1$ Taylor-Hood elements and $N_x = 6879$ degrees-of-freedom, see Figure 2. A backward Euler time discretization scheme is employed to discretize the temporal domain with equidistant time points $t_i = i\delta_0$ for $i = 0, 1, \ldots, 1024$, such that the time step size is given as $\delta_0 = T/1024 = 1\ ms$. Then the general discrete weak form of Equation (12) and Equation (13) can be written as follows:

Find $\boldsymbol{s}^{n+1} := (\boldsymbol{v}_f^{n+1}, p_f^{n+1}) \in \boldsymbol{\mathcal{V}}_D^h \times \mathcal{W}_f^h$, such that for every $\boldsymbol{d} := (\boldsymbol{w}_f, q_f) \in \boldsymbol{\mathcal{V}}_0^h \times \mathcal{W}_f^h$:

$$R(\boldsymbol{s}^{n+1}, \boldsymbol{s}^n, \boldsymbol{d}) = \int_{\Omega_f^h} \rho_f \frac{\boldsymbol{v}_f^{n+1} - \boldsymbol{v}_f^n}{\delta_0} \cdot \boldsymbol{w}_f + \left[\mu_f D\boldsymbol{v}_f^{n+1} - p_f^{n+1}\boldsymbol{I}\right] : \nabla \boldsymbol{w}_f + q_f \nabla \cdot \boldsymbol{v}_f^{n+1}\ d\Omega_f^h = 0, \tag{16}$$

where $D\boldsymbol{v} = \operatorname{sym}(\boldsymbol{v})$ and Ω_f^h refers to the discretized domain and the definition of the function spaces are

[9] A constant flow profile was applied across the outflow boundary for ease of setting up the numerical problem, as the overall flow problem was already simplified.

given as,

$$S^p(\Omega_f^h) = \{f : \Omega_f^h \to \mathbb{R} \mid f \in \mathcal{C}^0(\bar{\Omega}_f^h), f|_{\tau_e} \in \mathbb{P}^p(\tau_e), \forall\ \tau_e \in T_f^h\}, \tag{17}$$

which represent the general continuous p^{th}-order piecewise polynomial spaces defined on Ω_f^h, and T_f^h is the set of triangles composing Ω_f^h. Consequently,

$$\boldsymbol{\mathcal{V}}^h = [S^2(\Omega_f^h)]^2 \qquad \text{and} \qquad \mathcal{W}_f = S^1(\Omega_f^h), \tag{18}$$

and incorporating the boundary conditions yields,

$$\boldsymbol{\mathcal{V}}_0^h = \{\boldsymbol{v} \in \boldsymbol{\mathcal{V}}^h \mid \boldsymbol{v} = \mathbf{0} \text{ on } \Gamma_f^W\}, \tag{19}$$

$$\boldsymbol{\mathcal{V}}_D^h = \{\boldsymbol{v} \in \boldsymbol{\mathcal{V}}^h \mid \boldsymbol{v} = \boldsymbol{v}_f \text{ according to (14) and (15)}\}. \tag{20}$$

3.1.2. Convergence to a time-periodic steady state

For the linear flow problem, the periodicity is boundary-driven due to the pulsatile inflow. Using sequential time-stepping, the system can be driven to a periodic steady-state by simulating multiple cardiac cycles, such that each cycle starts off with an improved initial state. In contrast, time-periodic MGRIT as an iterative method achieves a time-periodic steady-state solution by continuously updating the initial condition during each iteration for a single cardiac cycle, see Section 2.4.

We obtain a time-periodic steady-state solution by simulating 10 subsequent cardic cycles using sequential time-stepping. The solution over the 10^{th} cycle is thus set as the steady-state reference solution. The reference solution is then compared with the sequential time-stepping solution for each cycle, as well as the time-periodic MGRIT approximation over one cycle for each MGRIT iteration. For comparison, we employ the $L_2(\Omega_t^h; L_2(\Omega_f^h))$-norm over the velocity field over a whole temporal cycle. For a better understanding of the initial condition convergence criterion, we further compute the time point-wise difference $L_2(\Omega_f^h)$-norover the velocity field for each sequential time-stepping cycle as well as time-periodic MGRIT iteration.

3.1.3. Numerical setup

To solve the spatial problem described above, we employ the finite element solver CHeart [45, 47], which is based on the matrix solver MUMPS [48]. For sequential time-stepping runs, CHeart steers the simulation by itself and it is set up to run 10 subsequent cardiac cycles. For the time-periodic MGRIT runs, wrapper code was written that implements the time-periodic MGRIT algorithm (see Section 2.4) based on the open-source library XBraid, to solve the space-time problem over one cardiac cycle iteratively. At the lowest level, the wrapper code calls CHeart to solve one time step according to the step function defined in Algorithm 3, using an initial condition, the current time and the current time step size (i.e. fine or coarse grid time step size).

Measures that indicate the distance to the time-periodic steady-state reference solution (as defined in Section 3.1.2) are computed *a posteriori* and are thus not part of the observed runtimes of the respective algorithms.

3.2. Fluid-structure interaction problem with analytic solution

In this section, we consider two cases of a class of analytic solutions for fluid-structure interaction [49]: the transient interaction of flow with a linear solid in a 2D channel (Section 3.2.1) and the transient interaction of flow with a nonlinear solid in a 3D vessel (Section 3.2.2).

3.2.1. Transient interaction between flow and linear solid

First, consider the time-periodic interaction of flow with a linear-elastic solid. The space-time domain corresponds to a 2D channel (see Figure 3) with extents 1×1.2 and cycle time of $T = 1.024$. Fluid density and viscosity are selected as $\rho_f = 1$ and $\mu_f = 10^{-2}$. Solid density and stiffness are selected as $\rho_s = 1$ and $\mu_s = 10^{-1}$.[10]

[10] In line with [49, Section 5.1], parameters "*were selected to highlight key solution features and for conducting a space-time convergence analysis. But they are arbitrary in the sense that they were chosen without a particular application area in*

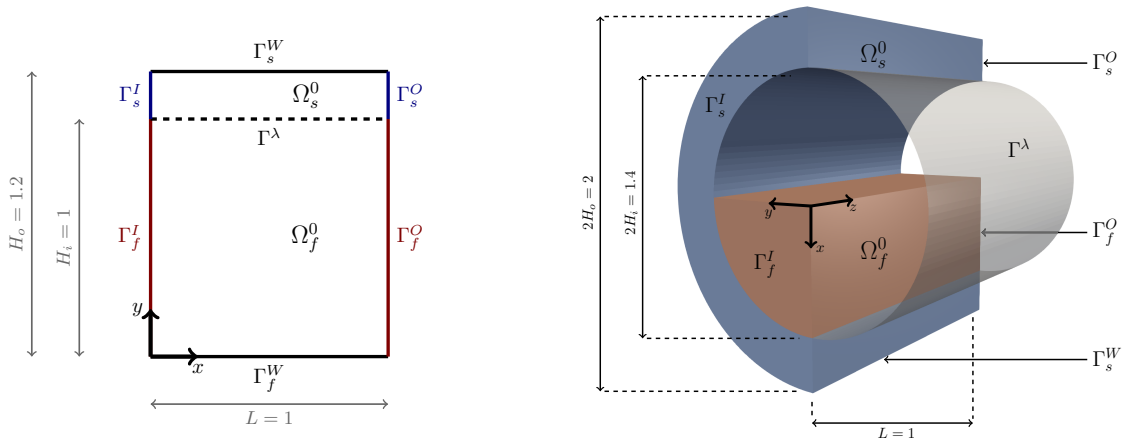


Figure 3: Fluid and solid reference domains, Ω_f^0 and Ω_s^0, in two dimensions with respective boundaries at the inlet (Γ_f^I and Γ_s^I), the outlet (Γ_f^O and Γ_s^O) and the wall (Γ_f^W and Γ_s^W). The common interface boundary is denoted as Γ^λ. Further, the domain length is denoted as L, the fluid domain height as H_i and the fluid / solid domain height as H_o. Modified from: [39, Figure 2.1].

The incompressible flow is modeled using the Stokes equations (see Equation (12) and Equation (13)). On the other hand, the incompressible, isotropic and homogeneous solid deformations are governed by the transient linear elasticity equations in first-order form:

$$\partial_t \boldsymbol{u}_s = \boldsymbol{v}_s \qquad \text{in } \Omega_s, \tag{21}$$

$$\rho_s \partial_t \boldsymbol{v}_s = \nabla \cdot [\mu_s D\boldsymbol{u}_s - p_s \boldsymbol{I}] \qquad \text{in } \Omega_s, \tag{22}$$

$$\nabla \cdot \boldsymbol{v}_s = 0 \qquad \text{in } \Omega_s, \tag{23}$$

with displacement $\boldsymbol{u}_s$, velocity $\boldsymbol{v}_s$ and hydrostatic pressure p_s. The two equation sets are coupled using the following interface conditions:

$$\boldsymbol{v}_f - \boldsymbol{v}_s = \boldsymbol{0}, \tag{24}$$

$$[\mu_f D\boldsymbol{v}_f - p_f \boldsymbol{I}] \cdot \boldsymbol{n}_f + [\mu_s D\boldsymbol{u}_s - p_s \boldsymbol{I}] \cdot \boldsymbol{n}_s = \boldsymbol{0}, \tag{25}$$

with outer boundary normal vectors for the fluid and solid subdomains $\boldsymbol{n}_f$ and $\boldsymbol{n}_s$, respectively.

Space-time discretization and weak formulation. Similar to Section 3.1, an inf-sup stable Taylor-Hood finite element discretization is employed in space with weak enforcement of the coupling constraints (24) / (25) through the introduction of a Lagrange multiplier variable, $\boldsymbol{\lambda} = \boldsymbol{t}_f = -\boldsymbol{t}_s$, see [50]. Quadratic quadrilateral elements are used for the fluid velocity, solid velocity and solid displacement variables, linear quadrilateral elements for the fluid pressure and solid pressure variables and quadratic line elements for the Lagrange multiplier variable. The spatial step size is selected as 0.025. For the time discretization, a backward Euler time discretization scheme [41] is selected with time step size 0.002. Overall, the space-time discretization amounts to 5120 time steps per cycle and 18088 spatial degrees-of-freedom.

The derivation of the weak form equations is analogeous to Section 3.1.1 and is therefore omitted.

mind. Thus, they are given in a unitless format." One key solution feature is that the parameter set yields significant solid deformations, whereas physiologically relevant parameters may not result in significant solid deformations, see [49, Section 5.3]. Furthermore, results in the corresponding results section are likewise given in unitless format.

3.2.2. Transient interaction between flow and nonlinear solid

Now adding complexity, we consider the time-periodic interaction of flow with a hyperelastic solid in a 3D vessel geometry. The spatial domain is derived by rotating the domain from Section 3.2.1 around the x-axis (see Figure 3) with extents 1×1 (solid thickness of 0.3). The cycle length is 1.024. Fluid density and viscosity are selected as $\rho_f = 2.1$ and $\mu_f = 3 \cdot 10^{-2}$. Solid density and stiffness are selected as $\rho_s = 1$ and $\mu_s = 10^{-1}$.[11]

The incompressible flow is modeled using the Navier-Stokes equations in Arbitrary Lagrangian-Eulerian (ALE) form. On the other hand, the incompressible, isotropic and homogeneous solid deformations are governed by the transient finite elasticity equations using a Neo-Hookean material law. Then, the FSI problem takes the following general form in accordance to [49]:

$$\rho_f \partial_t \boldsymbol{v}_f + \rho_f(\boldsymbol{v}_f - \boldsymbol{w}_f) \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}_f + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{\sigma}_f = \boldsymbol{0} \qquad \text{in } \Omega_f, \tag{26}$$

$$\nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}_f = 0 \qquad \text{in } \Omega_f, \tag{27}$$

$$[\boldsymbol{v}_f]_k = 0 \qquad \text{on } \Gamma_f^I \cup \Gamma_f^O, \text{ for } k \in \{x, y\}, \tag{28}$$

$$(\boldsymbol{\sigma}_f \cdot \boldsymbol{n}_f) \cdot \boldsymbol{e}_z = [\boldsymbol{t}_f]_z \qquad \text{on } \Gamma_f^I \cup \Gamma_f^O, \tag{29}$$

$$\boldsymbol{v}_f(\cdot, 0) = \boldsymbol{v}_f^0 \qquad \text{in } \Omega_f(0), \tag{30}$$

$$\rho_s \partial_{tt} \boldsymbol{u}_s - \nabla_{\boldsymbol{X}} \cdot \boldsymbol{P}_s = \boldsymbol{0} \qquad \text{in } \Omega_s(0) \times [0, T], \tag{31}$$

$$\det \boldsymbol{F} - 1 = 0 \qquad \text{in } \Omega_s(0) \times [0, T], \tag{32}$$

$$\boldsymbol{u}_s(\cdot, t) = \boldsymbol{0} \qquad \text{on } \Gamma_s^W, \tag{33}$$

$$[\boldsymbol{u}_s]_k = 0 \qquad \text{on } \Gamma_s^I \cup \Gamma_s^O, \text{ for } k \in \{x, y\} \tag{34}$$

$$(\boldsymbol{P}_s \cdot \boldsymbol{N}_s) \cdot \boldsymbol{e}_z = \boldsymbol{t}_s \qquad \text{on } \Gamma_s^I \cup \Gamma_s^O, \tag{35}$$

$$\boldsymbol{u}_s(\cdot, 0) = \boldsymbol{u}_s^0 \qquad \text{in } \Omega_s(0), \tag{36}$$

$$\boldsymbol{v}_s(\cdot, 0) = \boldsymbol{v}_s^0 \qquad \text{in } \Omega_s(0), \tag{37}$$

$$\boldsymbol{\sigma}_f \cdot \boldsymbol{n}_f + \boldsymbol{\sigma}_s \cdot \boldsymbol{n}_s = \boldsymbol{0} \qquad \text{on } \Gamma^\lambda, \tag{38}$$

$$\boldsymbol{v}_f - \boldsymbol{v}_s = \boldsymbol{0} \qquad \text{on } \Gamma^\lambda, \tag{39}$$

with fluid velocity $\boldsymbol{v}_f$, fluid domain velocity $\boldsymbol{w}_f$, fluid Cauchy stress tensor $\boldsymbol{\sigma}_f = \mu_f D\boldsymbol{v}_f - p_f \boldsymbol{I}$ with fluid pressure p_f, solid displacement $\boldsymbol{u}_s$, the first Piola-Kirchhoff stress tensor,

$$\boldsymbol{P}_s = \frac{\mu_s}{(\det \boldsymbol{F})^{2/3}} \left[\boldsymbol{F} - \frac{\boldsymbol{F} : \boldsymbol{F}}{3} \boldsymbol{F}^{-T} \right] - p_s \boldsymbol{F}^{-T},$$

the deformation gradient tensor $\boldsymbol{F} = \nabla_{\boldsymbol{X}} \boldsymbol{u}_s + \boldsymbol{I}$, the solid Cauchy stress tensor $\boldsymbol{\sigma}_s = \frac{1}{\det \boldsymbol{F}} \boldsymbol{P}_s \boldsymbol{F}^T$, the solid pressure p_s and the outer boundary normals $\boldsymbol{n}_f$ on the fluid domain and $\boldsymbol{n}_s$ and $\boldsymbol{N}_s$ on the solid and solid reference domain, respectively.

In the discrete setting, care must be taken in order to properly manage coupling between (fluid and solid) meshes. This is achieved by mapping both coupling surfaces to a common interface surface Γ^λ. Further, note that using Nanson's formula (i.e. $\boldsymbol{n}_s = (\det \boldsymbol{F}) \boldsymbol{F}^{-T} \boldsymbol{N}_s$) in Equation 38, the solid traction can be written using the Cauchy and first Piola-Kirchhoff stress tensor as $\boldsymbol{\sigma}_s \cdot \boldsymbol{n}_s = \boldsymbol{P}_s \cdot \boldsymbol{N}_s$.

Here, we omit the expansive definition of initial, Dirichlet and Neumann boundary conditions (i.e. $\boldsymbol{v}_f^0$, $\boldsymbol{t}_f$, $\boldsymbol{t}_s$, $\boldsymbol{u}_s^0$, $\boldsymbol{v}_s^0$), as those are adopted unmodified from the original work [49].[12]

[11] Similar to Section 3.2.1, parameters were selected according to [49, Section 5.2].

[12] For the time-periodic MGRIT run, the initial condition is considered to be the analytic solution for all variables except for the fluid velocity, which is initially zero everywhere. This is to achieve a good initial condition and initial guess with a

Space-time discretization and weak formulation. Again, using inf-sup quadratic-linear Taylor-Hood finite elements (fluid: tetrahedral mesh; solid: hexahedral mesh; Lagrange multiplier: triangular mesh) and backward Euler time integration, we consider two mesh *refinement levels*: *coarse* and *fine*. The domains are discretized using approximate space-time step sizes of $(0.314, 0.002)$ and $(0.157, 0.0005)$ for the *coarse* and *fine* meshes, which results in 32793×512 and 245009×2048 space-time degrees-of-freedom, respectively.

For conciseness, we omit the weak form of (26) – (39) as it is derived similarly to Section 3.1, however, note that due to the ALE term in Equation (39) the weak form is expanded by an ALE diffusion problem to account for the fluid domain motion according to [49, Equation (85)].

The fluid-solid coupling across these discrete interfaces requires care, as the discretized form consists of triangles (fluid side) and curvilinear quadrilaterals (solid side). This is achieved by developing an interface discretization with a bijective map formed between both fluid and solid discretizations (see [50] for details). The fluid-solid systems are coupled through the use of a Lagrange multiplier (representing the surface traction) and solved as a monolithic system [50, 49].

3.2.3. Convergence to a time-periodic steady state

Due to the presence of an analytic solution [49], it is possible to compute the space-time error for each algorithm at each cycle and iteration, respectively. Thus, the solution progress can be tracked and the occurence of a time-periodic steady-state can be detected. This is done by computing the spatial ℓ_2 error at each time point (over the length of a cycle), as well as by numerically integrating the (time) pointwise spatial errors over the temporal cycle length. Thus, it is possible to track where errors occur in time (depending on the distance to the initial condition) and how the space-time error is reduced *cycle-to-cycle* and *iteration-to-iteration*.

3.2.4. Numerical setup

Similar to Section 3.1, CHeart and XBraid are employed to simulate the space-time system using sequential time-stepping (10 and 7 cardiac cycles for the linear and nonlinear case, respectively) and time-periodic MGRIT (up to 12 iterations) with FCF-relaxation. In the nonlinear PDE case, a Newton-Raphson-Shimanskii solver [51] is employed to effectively reduce the computational cost through reuse of the Jacobian matrix [47]. In the MGRIT case, this is handled by storing separate matrix objects [39], such that matrix reuse can be applied on each time grid independently.

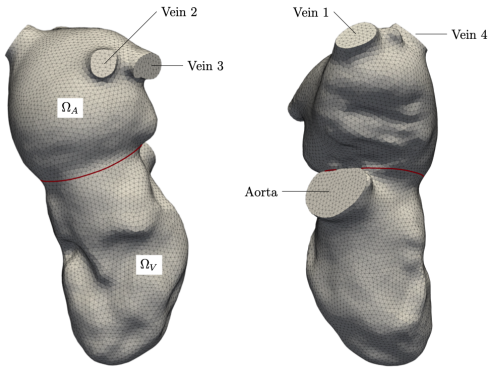
By adding a solver step to evaluate the analytic solution in CHeart, the space-time error is computed on-the-fly. Therefore, reported runtimes in Section 4 include the time it takes to compute the current space-time error.

3.3. Flow in a left atrium / left ventricle geometry

Lastly, we consider nonlinear flow in a patient-specific left atrium (LA) / left ventricle (LV) geometry. The geometry (see Figure 4), as well as the time-dependent deformation of the LA and LV were extracted from computed tomography (CT) data. The CT data set stems from a Cardiac Resynchronization Therapy (CRT) patient that suffers from an abnormal wall motion and a reduced ejection fraction (see Section 4.3). To reduce the exposure of the patient to ionizing radiation during a CT scan, the acquisition time for such patients is kept as short as possible. Thus, no (additional) flow data (e.g., using Doppler echocardiography [52]) could be acquired, such that the data from numerical models are the only data available to assess cardiac function and, in particular, how blood flow through the LA and LV are affected.

For the computational model, flow over the LA and LV subdomains are modeled separately and coupling is enforced at the common interface using appropriate coupling constraints defined in the following. The common interface is selected as the planar section, where the mitral valve (MV) would be located. The MV geometry and deformation were not available from the data set provided, and thus, the MV opening at $t = 0\ s$ is approximated as an ellipse with minor and major axes of $l_{MV}^{min} \approx 2.66\ cm$ and $l_{MV}^{max} \approx 4.64\ cm$.

pertubation to simulate a practical case where some (experimental) data are available but convergence to a periodic steady-state is required.



Ω_A	:	LA domain
Γ_{V1}	:	Vein 1 boundary
Γ_{V2}	:	Vein 2 boundary
Γ_{V3}	:	Vein 3 boundary
Γ_{V4}	:	Vein 4 boundary
Γ_{Vn}	:	$\Gamma_{V1} \cup \Gamma_{V2} \cup \Gamma_{V3} \cup \Gamma_{V4}$
Γ_{CA}	:	Common surface with LV
Γ_{CAo}	:	Coupling surface of LA that is open during diastole
Γ_{WA}	:	Endocardial wall of LA domain
Ω_V	:	LV domain
Γ_{AV}	:	Aortic valve boundary
Γ_{CV}	:	Common surface with LA
Γ_{CVo}	:	Coupling surface of LV that is open during diastole
Γ_{WV}	:	Endocardial wall of LV domain
Ω_{LM}	:	$\Omega_A \cap \Omega_V$, Lagrange multiplier (LM) domain
Γ_{LM}	:	LM boundary

Figure 4: Patient-specific model of the left atrium (Ω_A) and left ventricle (Ω_V), showing initial mesh and the different inflow and outflow boundaries at pulmonary veins and the aorta. The red line indicates the coupling surface between atrium and ventricle. Figure modified from [53, Figure 3].

The MV is open for $t \in (0, 0.37]$ s (diastole) and closed for $t \in (0.37, 0.8]$ s (systole). To switch between diastole and systole, we introduce a scalar function,

$$\alpha(t) = \begin{cases} 0 & \text{for } t \in (0, 0.37], \\ 1 & \text{for } t \in (0.37, 0.8]. \end{cases} \tag{40}$$

The domains and domain boundaries are denoted as illustrated in Figure 4. Blood is modeled as an incompressible Newtonian fluid with density $\rho = 1.025$ g/cm^3 and viscosity $\mu = 0.004$ $Pa \cdot s$. Flow in the atrium and ventricle is governed by the nonconservative ALE Navier-Stokes equations for a Newtonian fluid. This enables the solution of fluid flow in the moving domains Ω_A and Ω_V, with the motion given by domain velocities w_A and w_V. The surface motion of both domains were extracted from CT data, with the internal motion defined using a pseudo-elastic problem with element based stiffening (see [54]).

The strong form equations, coupling, boundary and initial conditions are given as,

$$\rho\partial_t \boldsymbol{v}_A + \rho\left(\boldsymbol{v}_A - \boldsymbol{w}_A\right)\cdot\nabla_{\boldsymbol{x}}\boldsymbol{v}_A - \nabla_{\boldsymbol{x}}\cdot\boldsymbol{\sigma}_A = \boldsymbol{0} \quad \text{in } \Omega_A, \tag{41}$$

$$\nabla_{\boldsymbol{x}}\cdot\boldsymbol{v}_A = 0 \quad \text{in } \Omega_A, \tag{42}$$

$$\boldsymbol{v}_A = \boldsymbol{w}_A \quad \text{on } \Gamma_{WA}, \tag{43}$$

$$\boldsymbol{\sigma}_A\cdot\boldsymbol{n}_A = \boldsymbol{t}_A \quad \text{on } \Gamma_{Vn}, \tag{44}$$

$$\boldsymbol{v}_A(\cdot,0) = \boldsymbol{w}_A \quad \text{on } \Omega_A(0), \tag{45}$$

$$\rho\partial_t \boldsymbol{v}_V + \rho\left(\boldsymbol{v}_V - \boldsymbol{w}_V\right)\cdot\nabla_{\boldsymbol{x}}\boldsymbol{v}_V - \nabla_{\boldsymbol{x}}\cdot\boldsymbol{\sigma}_V = \boldsymbol{0} \quad \text{in } \Omega_V, \tag{46}$$

$$\nabla_{\boldsymbol{x}}\cdot\boldsymbol{v}_V = 0 \quad \text{in } \Omega_V, \tag{47}$$

$$\boldsymbol{v}_V = \boldsymbol{w}_V \quad \text{on } \Gamma_{WV}, \tag{48}$$

$$\boldsymbol{v}_V = \boldsymbol{w}_V \quad \text{on } \Gamma_{AV} \text{ for } t \in (0, 0.37], \tag{49}$$

$$\boldsymbol{\sigma}_V\cdot\boldsymbol{n}_V = \boldsymbol{t}_V \quad \text{on } \Gamma_{AV} \text{ for } t \in (0.37, 0.8], \tag{50}$$

$$\boldsymbol{v}_V = \boldsymbol{w}_V \quad \text{on } \Gamma_{CV}\setminus\Gamma_{CVo} \text{ for } t \in (0, 0.37], \tag{51}$$

$$\boldsymbol{v}_V(\cdot,0) = \boldsymbol{w}_V \quad \text{on } \Omega_V(0), \tag{52}$$

$$\boldsymbol{v}_A - \alpha\boldsymbol{w}_A - (1-\alpha)\boldsymbol{v}_V = \boldsymbol{0} \quad \text{on } \Omega_{LM}, \tag{53}$$

$$\alpha\left(\boldsymbol{v}_V - \boldsymbol{w}_V\right) + (1-\alpha)\left(\boldsymbol{\sigma}_V\cdot\boldsymbol{n}_V + \boldsymbol{\sigma}_A\cdot\boldsymbol{n}_A\right) = \boldsymbol{0} \quad \text{on } \Omega_{LM}, \tag{54}$$

with velocity $\boldsymbol{v}_A$ (atrium) and $\boldsymbol{v}_V$ (ventricle), domain velocity $\boldsymbol{w}_A$ (atrium) and $\boldsymbol{w}_V$ (ventricle) obtained from CT data (see Section 3.3), outward boundary normal $\boldsymbol{n}_A$ (atrium) and $\boldsymbol{n}_V$ (ventricle), and Cauchy stress tensor,

$$\boldsymbol{\sigma}_A = \mu D\boldsymbol{v}_A - p_A\boldsymbol{I}, \tag{55}$$

$$\boldsymbol{\sigma}_V = \mu D\boldsymbol{v}_V - p_V\boldsymbol{I}, \tag{56}$$

with pressure p_A (atrium) and p_V (ventricle). Further, the scalar function $\alpha = \alpha(t)$ in Equation (53) and Equation (54) is given by Equation (40) and enables switching the coupling constraints between diastole and systole: (i) during diastole, continuity of velocity and traction is enforced across the coupling boundary and (ii) during systole, a no-slip condition is prescribed on the coupling boundary.

Further, outflow stabilization is employed at the veins (see Equation 44) to deal with potential reflow on Γ_{Vn}, and at the aortic valve boundary during systole (see Equation (50)),

$$\boldsymbol{t}_A = \frac{\rho\beta}{2}\frac{\left(\boldsymbol{v}_A\cdot\boldsymbol{n}_A\right)^2}{\left(\boldsymbol{v}_A\cdot\boldsymbol{n}_A\right)^2 + 0.01}\min\left\{\boldsymbol{v}_A\cdot\boldsymbol{n}_A, 0\right\}\boldsymbol{v}_A, \tag{57}$$

$$\boldsymbol{t}_V = \frac{\rho\beta}{2}\frac{\left(\boldsymbol{v}_V\cdot\boldsymbol{n}_V\right)^2}{\left(\boldsymbol{v}_V\cdot\boldsymbol{n}_V\right)^2 + 0.01}\min\left\{\boldsymbol{v}_V\cdot\boldsymbol{n}_V, 0\right\}\boldsymbol{v}_V. \tag{58}$$

Note, that Equation (57) and Equation (58) provide a modified version of the outflow stabilization proposed in [55] with a scaling of $\beta = 0.2$, as suggested in [56].

On the coupling domain, Lagrange multiplier variables are defined as $\boldsymbol{\lambda}_A = \boldsymbol{\sigma}_A\cdot\boldsymbol{n}_A$ and $\boldsymbol{\lambda}_V = -\boldsymbol{\sigma}_V\cdot\boldsymbol{n}_V$ to enforce the coupling constraints, such that during dyastole: $\boldsymbol{\lambda}_A - \boldsymbol{\lambda}_V = \boldsymbol{0}$. The flow is modeled by a stabilized general Galerkin scheme (instead of using, e.g., an inf-sup stable Taylor-Hood finite element discretization scheme) for the incompressible Navier-Stokes equations; namely the cG(1)cG(1) scheme as

given in the study of Hoffman et al. [57]. The scheme was implemented in CHeart [46, 45], and validated in a previous work [47].

3.3.1. Spatiotemporal discretization

The temporal domain is discretized using 800 equidistant time steps $0\ s = t_0 < t_1 < \ldots < t_{N_0-1} = 0.8\ s$ with constant time step size $\delta_0 = t_{n+1} - t_n = 0.001\ s$ for $n = 0, \ldots, N_0 - 2$. A backward Euler time discretization scheme is employed to discretize the flow problem in time.[13] The atrial and ventricular domains, Ω_A and Ω_V, are discretized using 66484 and 69716 tetrahedral elements with spatial step sizes in the range of $0.163 - 0.290\ cm$ and $0.194 - 0.287\ cm$. The coupling domain Ω_{LM} is discretized using 945 triangular elements that conform with the tetrahedral elements at the coupling boundary of the atrium and ventricle. In the following, we use Ω_A^0 and Ω_V^0 to refer to the initial meshes discretizing atrium and ventricle; further, Ω_A^n and Ω_V^n refer to the respective current meshes.

Finite element discretizations were constructed using $\mathbb{P}^1 - \mathbb{P}^1$ elements for fluid velocity and pressure and $\mathbb{P}^1$ elements for the Lagrange multipliers on the coupling domain, resulting in 55842 degrees-of-freedom (DOFs). Here, we omit the weak form as it is quite expansive, however, refer the interested reader to the Supplementary Materials SM2.

3.3.2. Convergence to a time-periodic steady state

Similar to the problems described in the previous sections, the system has to be driven to a periodic steady-state because no data is available to prescribe a meaningful initial flow condition. Thus, zero initial flow is assumed and multiple cycles are simulated (sequential time-stepping) or multiple time-periodic MGRIT iterations over the length of one cycle are performed. To detect a time-periodic steady-state, the rate of inflow / outflow at the pulmonary veins[14] is measured over the entire cycle length at 10 ms increments.

3.3.3. Numerical setup

Similar to Section 3.1 and Section 3.2, CHeart and XBraid are employed to simulate the space-time system using sequential time-stepping (10 cardiac cycles) and time-periodic MGRIT (10 iterations) with FCF-relaxation. Again, a Newton-Raphson-Shimanskii solver [51] is employed to efficiently solve the coupled nonlinear system. Here, we add a solver step to evaluate the flow rate at all pulmonary veins on-the-fly. Thus, reported runtimes in Section 4 include the time it takes to compute the flow rates.

4. Results

In this section, we present results for each of the application classes described in Section 3, using the methods described in Section 2. For each application, we start by describing the numerical solution itself. We continue by highlighting how the sequential time-stepping solution converges to a time-periodic steady-state (by running multiple cycles consecutively) and how this compares to applying the time-periodic MGRIT algorithm to only a single time period. Finally, we present timing and speedup results for both the sequential time-stepping and the parallel-in-time algorithm.

4.1. Simplified flow in the ascending aorta

Figure 5 shows a snapshot of the flow field at 3/8 of the 10^{th} cycle due to the pulsatile flow. The parabolic inflow profile (see Equation (14)) is constricted between the rigid valves to form a jet. The jet quickly widens to an approximately parabolic flow profile, similar to Poiseuille flow in a channel; for example, note the (close

[13] Preliminary numerical tests showed that using a backward Euler-like scheme with midpoint rule in combination with parallel-in-time integration can result in unphysical oscillations at inflow and outflow boundaries, similar to those seen in literature for other problems (see [41]), causing the scheme to be nonconvergent. Several works have found the benefit of time-discretization schemes with the property of L-stability for parallel-in-time discretization schemes [25, 26, 27]. In this work, however, a pure backward Euler and thus, L-stable time discretization scheme was employed.

[14] Note, that the flow rates at the MV and AV surfaces can be captured with machine precision since the fluxes reflect the prescribed volume change in the ventricle during diastole (flow between LA and LV) and systole (flow across AV surface).

to) parallel streamlines. In the linear flow case considered here, the pulsatile inflow effectively scales the observed dynamic velocity field across the flow domain. Further note the small reflow regions between the valves and the coronary arteries.

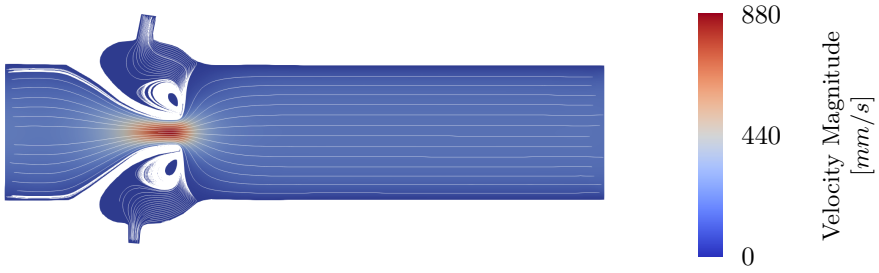


Figure 5: Stokes flow: Velocity magnitude at 3/8 of the 10^{th} cycle. The parabolic inflow becomes a jet between the valves. The jet quickly widens to an approximately parabolic flow profile, similar to Poiseuille flow in a channel.

4.1.1. Time-periodic steady-state

Figure 6 illustrates convergence of the solution at the beginning of each cycle to the solution at the end of the 10^{th} cycle for the Stokes flow model. Here, the average error reduction is approximately one order of magnitude for each simulated cycle. This highlights the fact that the accuracy of the periodic steady-state solution is directly linked to the number of simulated cycles. As we will see later, each reduction of the cycle-to-cycle error by approximately one order of magnitude increases the total computational cost by the computational cost of adding one cycle length to the temporal domain. In the context of using parallel-in-time integration with MGRIT, this motivated reducing the computational cost by simulating only one cardiac cycle but making the time grid periodic, see Section 2.4.

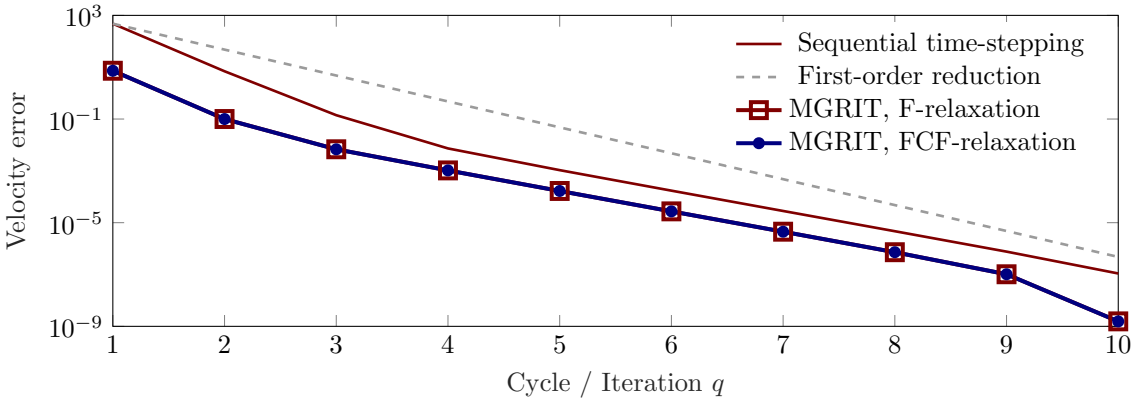


Figure 6: Convergence of the velocity field at the beginning of each cycle at $t = (q-1)T$ for $q = 1, 2, \ldots$ sequential time-stepping cycles and at $t = 0$ for $q = 0, 1, 2 \ldots$ time-periodic MGRIT iterations (note, that the zeroth cycle initializes the space-time solution and is omitted here): Velocity error with respect to the solution at the end of the 10^{th} cycle (at $t = 10T$) from sequential time-stepping.

On the other hand, two-level MGRIT converges at a similar rate, see Figure 6. That is, one time-periodic MGRIT iteration reduces the error at the beginning of the cycle similarly quickly compared to one cycle using sequential time-stepping. Here, F- and FCF-relaxation don't show any significant difference in periodic steady-state convergence.

Now, considering how the time point-wise MGRIT residual is distributed, we can connect updating the initial condition during each MGRIT iteration with the largest observed time-point wise residual: Figure 7 illustrates how large residuals occur at the beginning of the temporal cycle for each MGRIT iteration. This

makes sense as the largest mismatch / discontinuity in the space-time solution will occur at $t = 0$ due to the continuously updated initial condition; it is this mismatch that is then propagated in time and appears as a relatively larger residual norm, which we refer to as a *spike*. Looking closer at Figure 7, it is seen that the spikes occur at the fourth time point for FCF-relaxation (and at second time point for F-relaxation; plot omitted), which is due to the fact that MGRIT propagates the exact solution across two coarse-grid intervals for FCF-relaxation (and one coarse-grid interval for F-relaxation). This is in contrast to traditional (non-periodic) MGRIT, where the largest residual norms tend to occur towards the end of the temporal domain, i.e. with greater distance to the initial condition.

Since the spike only occurs very locally and right at the beginning of the temporal domain, skipping the updates of the initial condition once it is considered converged (see Section 2.4) very effectively zeroes the residual due to the exactness property of traditional MGRIT [25]. This leaves fairly balanced MGRIT residuals across the temporal domain.

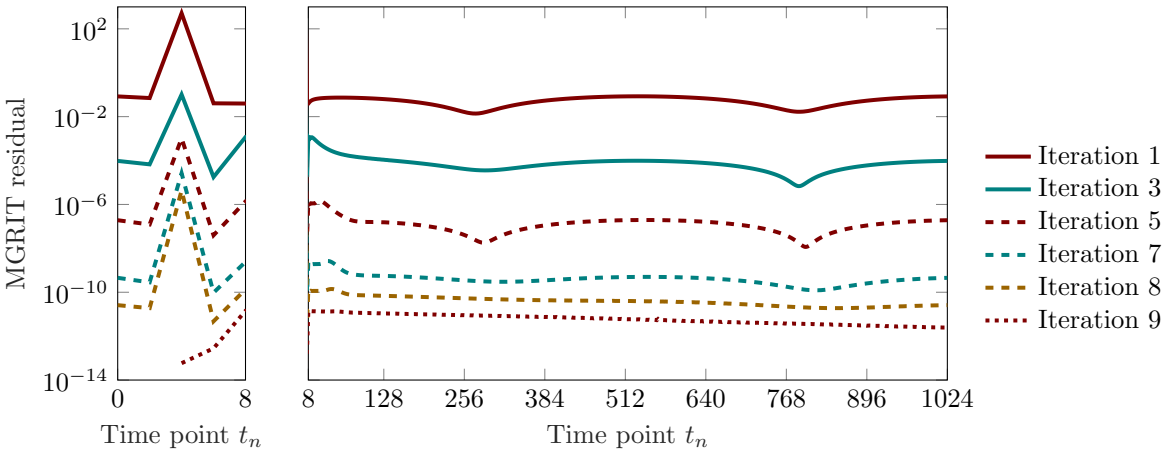


Figure 7: Two-level convergence of MGRIT with FCF-relaxation and temporal coarsening factor 2 for the time-periodic Stokes problem: MGRIT residual at C-points for multiple iterations if IC tolerance 10^{-10} is used.

4.1.2. Parallel performance

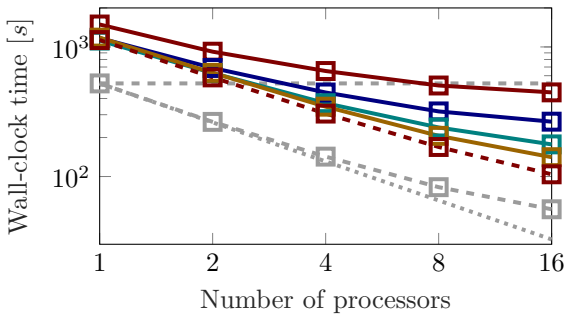
Let's now consider parallel performance with respect to parallelism applied to the spatial domain, the temporal domain and the spatiotemporal domain. Runtimes reported here were obtained using two small-scale clusters: LEAD and TheoSim (see Section SM3.1). The reported runtime of the serial algorithm is the best of five runs and of the parallel algorithms it is the best of one run. Reported runtimes were further rounded to the nearest integer number in seconds.

If no parallelism is employed, the sequential time-stepping algorithm takes 523 *s* on LEAD and 332 *s* on TheoSim (referred to as *baseline*). Employing spatial parallelism, the time-to-solution can be reduced (see Figure 8) to 56 *s* on LEAD (speedup: 9.34x) and 43 *s* on TheoSim (speedup: 7.72x), using 16 processors.

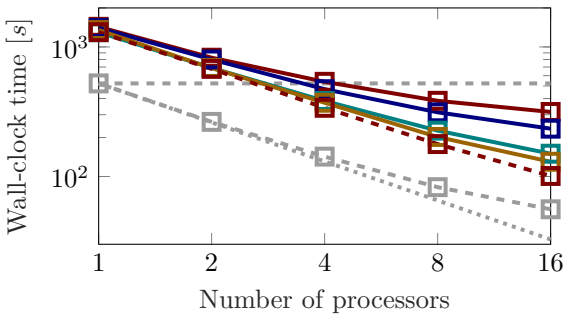
Using two-level MGRIT without parallelism yields a more expensive algorithm, e.g. see Figure 8a. The benefit of MGRIT is exploited by using more processors and by coarsening more aggressively. For time-only parallelism, the largest speedup of 6.92x can be observed for MGRIT with FCF-relaxation and coarsening factor $m = 32$ when using 32 processors on TheoSim. For space-time parallelism, the largest speedup of 36.89x can be observed for MGRIT with F-relaxation and coarsening factor $m = 32$ when using 256 processors (i.e. 16 processors are used for parallelizing in the temporal domain, while each spatial problem is solved in parallel using 16 processors) on TheoSim.

More detailed runtime data are reported in Table 5. From the data, we can conclude that for time-only parallelism, FCF-relaxation gives a slightly better speedup than F-relaxation, however, F-relaxation gives a slightly better speedup if space-time parallelism is employed.[15] Overall, the achieved speedups for two-level

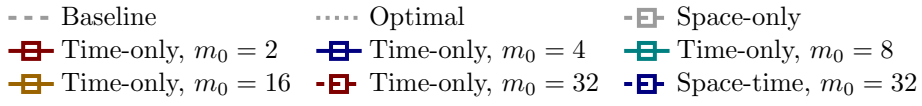
[15]F- and FCF-relaxation yield a similar convergence behavior, see Figure 6. Thus, it is likely that the additional cost

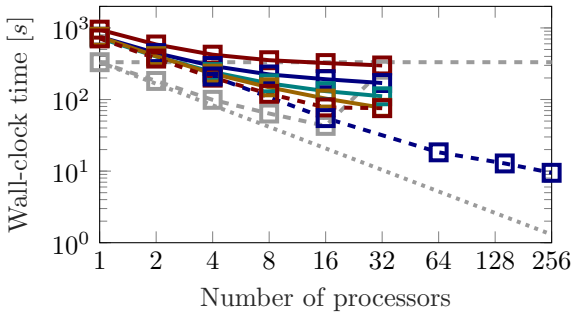


(a) LEAD: MGRIT with F-relaxation. Maximum observed speedup with time-only parallelism is 5.03x with 16 processors.

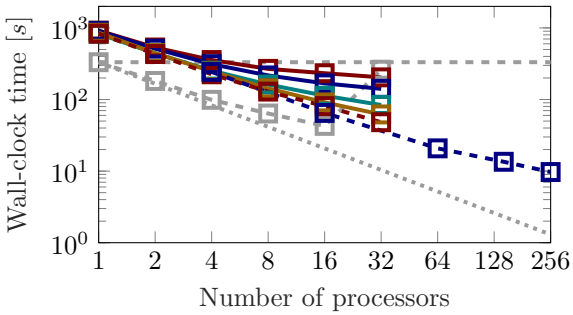


(b) LEAD: MGRIT with FCF-relaxation. Maximum observed speedup with time-only parallelism is 5.18x with 16 processors.





(c) TheoSim: MGRIT with F-relaxation. Maximum observed speedup with time-only parallelism is 4.37x with 32 processors. Speedup with space-time parallelism is 36.89x with 256 processors.



(d) TheoSim: MGRIT with FCF-relaxation. Maximum observed speedup with time-only parallelism is 6.92x with 32 processors. Speedup with space-time parallelism is 33.20x with 256 processors.

Figure 8: Runtimes on LEAD (top row) and TheoSim (bottom row) for sequential time-stepping (10 cycles) and time-periodic two-level MGRIT (q iterations, such that the initial condition tolerance 10^{-10} was satisfied) with F-relaxation (left column) and FCF-relaxation (right column).

MGRIT are quite large compared to results reported in literature. For example, for a two-dimensional compressible fluid dynamics application Falgout et al. [58] reported a speedup of 7.53x using time-only parallelism on 4096 processors, i.e. on a considerably larger number of processors. While the setting is slightly different (e.g., different Re number, different space-time resolution, etc.), the increased parallel efficiency is arguably achieved through exploiting the periodicity of the solution in time, and thus, to the development of the new time-periodic MGRIT variant in Section 2.4.

4.2. Fluid-structure interaction problem with analytic solution

In this section, we report results for the linear and nonlinear FSI problems with analytic solutions in Sections 4.2.1 and 4.2.2, respectively.

associated with FCF-relaxation impacts the runtime more negatively if more processors are assigned to the spatial component as opposed to the temporal component.

4.2.1. Transient interaction between flow and linear solid

The coupled linear FSI problem is driven by time-periodic boundary conditions with a time-periodic steady-state solution. For the given parameter setting, the flow is pulsatile with strong gradients near the fluid-solid coupling surface, see Figure 9, whereas the deformation rate is nonlinear along the y-axis. The flow and deformation rate are further equal at the interface (as enforced by the coupling conditions) and constant along the main flow direction (i.e. the x-axis), see [49].

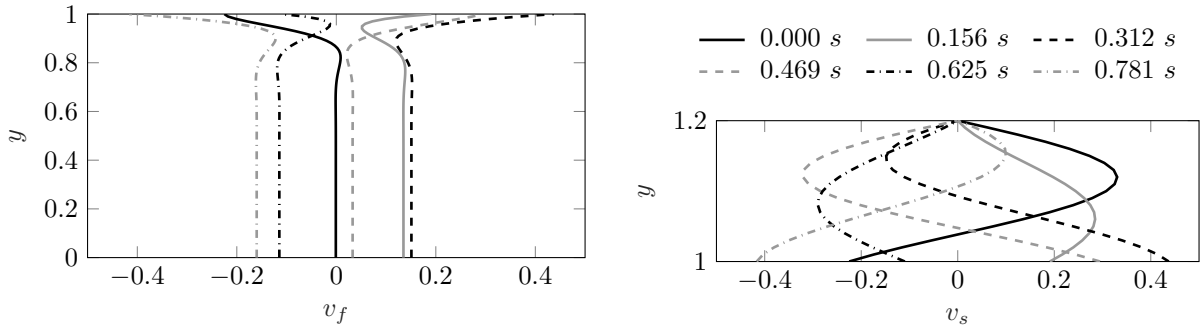


Figure 9: The analytic solution for the transient linear FSI case in two dimensions with density $\rho_f = \rho_s = 1$, fluid viscosity $\mu_f = 0.01$ and solid stiffness $\mu_s = 0.1$: Fluid and solid velocity, v_f (left) and v_s (right) along the y-axis over one cycle $T = 1.024$.

Time-periodic steady-state. In this section, we compare the error in the fluid velocity variable[16] at the end of each cycle using sequential time-stepping and the end of a single time-periodic cycle for each iteration using time-periodic MGRIT. Using sequential time-stepping, a time-periodic steady-state can be achieved by running 10 cycles sequentially, see Figure 10. A similar space-time error can be achieved by running 10 time-periodic MGRIT iterations.

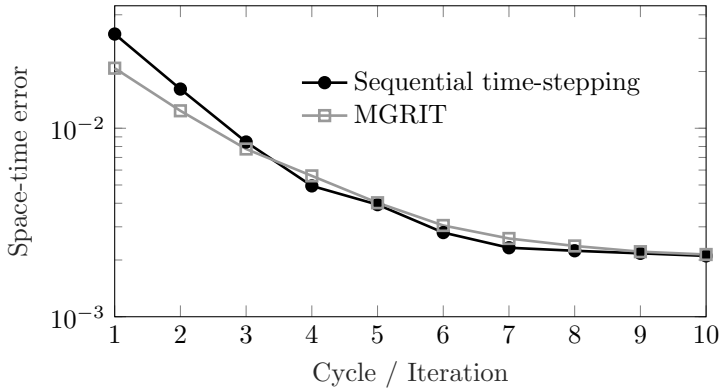


Figure 10: Space-time error reduction (velocity) for sequential time-stepping over 10 cycles and time-periodic MGRIT over 10 iterations.

Figure 10 further illustrates that MGRIT yields a smaller error after the first iteration compared to sequential time-stepping after the first cycle. This is due to the fact that the initial setup step for MGRIT is a coarse-grid solve, and thus, MGRIT starts off from a better starting position for the first iteration.

[16]Here, we choose to track the error in the fluid velocity variable since the fluid flow field exhibits larger spatial gradients than the solid deformation rate, see Figure 9.

Parallel performance. Next, we compare the wall clock times (when using ORCA, see Section SM3.1) for sequential time-stepping over 10 cycles and for the time-periodic MGRIT algorithm over 10 iterations (see Table 1) with no data export (i.e. only measuring elapsed times for computation and communication). The wall clock time of the sequential time-stepping algorithm was 458.4 seconds for 10 cycles (approximately 45.8 seconds per cycle). On the other hand, the time-periodic MGRIT algorithm takes 216.0 seconds when no parallelism is employed and only 25.5 seconds when using 32 processors in the temporal domain (i.e. no spatial parallelism), yielding a maximum speedup of 17.98x, see Table 1. Figure 11 further highlights the achieved space-time error for each simulated cycle of sequential time-stepping or each iteration of time-periodic MGRIT. From Figure 11 it is clear, that time-periodic MGRIT can achieve the same order of accuracy at a lower runtime.

Algorithm	Number of cycles	Number of iterations	Number of processors	Wall clock time	Speedup vs. 10 cycles
Time-stepping	10	-	1	458.4 s	-
MGRIT	1	10	1	216.0 s	2.12x
MGRIT	1	10	4	68.8 s	6.66x
MGRIT	1	10	16	30.5 s	15.03x
MGRIT	1	10	32	25.5 s	17.98x

Table 1: Wall clock time and speedups for two-level MGRIT with FCF-relaxation and temporal coarsening factor $m = 8$.

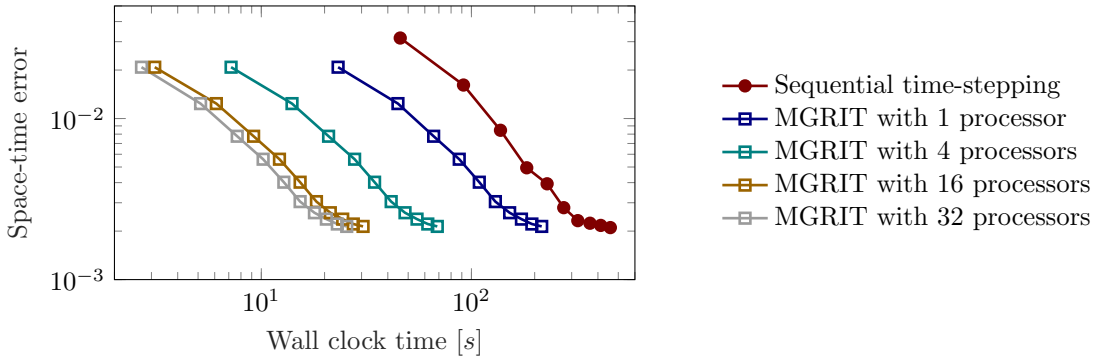


Figure 11: Space-time error (velocity) compared to wall clock time of sequential time stepping and time-periodic MGRIT with zero initial condition. Better accuracy can be achieved by simulating additional cycles or performing more MGRIT iterations, In general, time-periodic MGRIT can achieve the same order of accuracy at a lower cost.

4.2.2. Transient interaction between flow and nonlinear solid

The coupled nonlinear FSI problem is driven by time-periodic boundary conditions with a time-periodic steady-state solution. Similar to the linear case, the flow is pulsatile; however, with less pronounced gradients near the fluid-solid coupling surface, see Figure 12, whereas the deformation rate is nonlinear along the y-axis. The flow and deformation rate are further equal at the interface (as enforced by the coupling conditions) and constant along the main flow direction (i.e. the z-axis), see [49]. In contrast to the linear case, the fluid and solid pressure are discontinuous across the coupling interface and the solid pressure is varying along the y-axis.

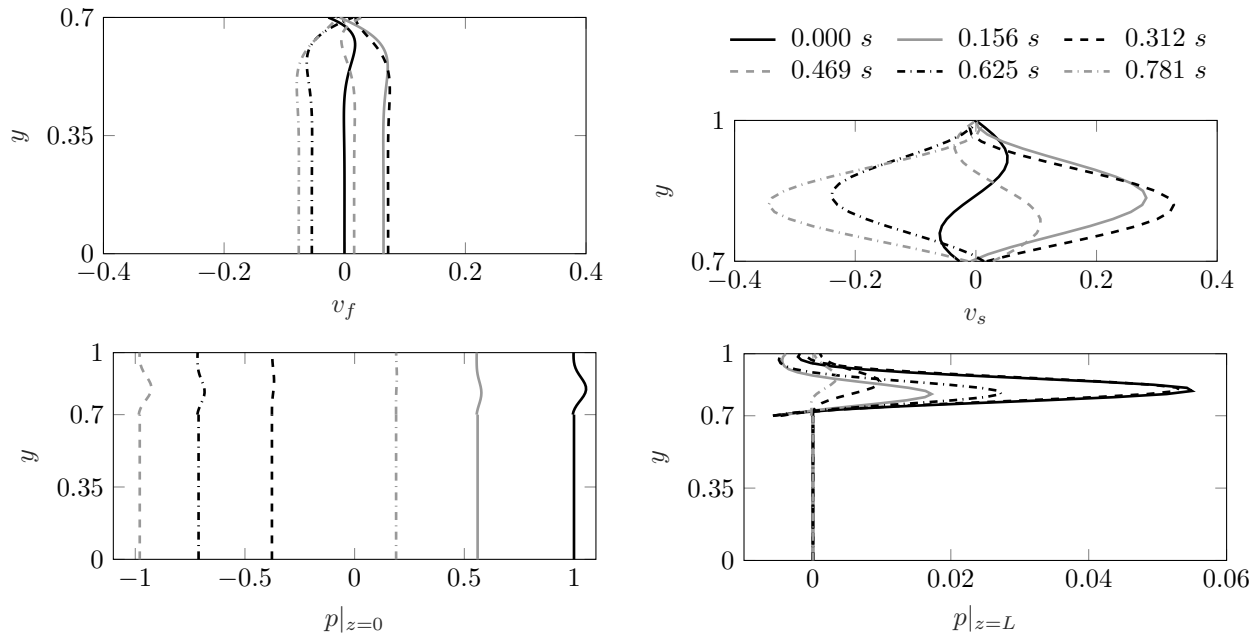
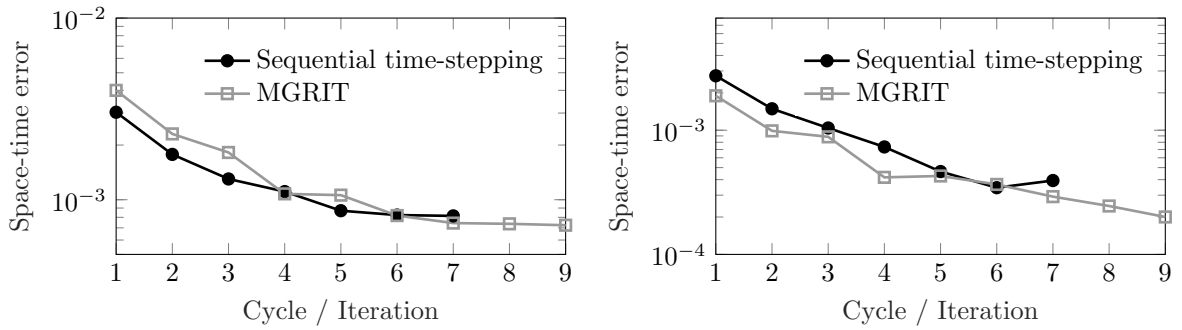


Figure 12: The analytic solution for the transient nonlinear FSI case in three dimensions with fluid and solid density $\rho_f = 2.1$ and $\rho_s = 1$, fluid viscosity $\mu_f = 0.03$, solid stiffness $\mu_s = 0.1$ and cycle length $T = 1.024$: Fluid and solid velocity, v_f (top left) and v_s (top right), along the y-axis. Further, fluid and solid pressure (bottom), $p = p_f$ for $y \in [0, 0.7]$ and $p = p_s$ for $y \in [0.7, 1]$ over time t at the inlet ($z = 0$) and the outlet ($z = L$).

Time-periodic steady-state. First, consider the convergence of the fluid velocity solution to its time-periodic steady-state for the *coarse* mesh. Figure 13a illustrates how the fluid velocity error is reduced over 7 consecutive cycles using sequential time-stepping. Starting with a slightly larger error at iteration 1 (relative to the error after one sequential time-stepping cycle), time-periodic MGRIT can reduce the space-time error similarly rapidly and eventually reduces the error further. A comparable order of error can be achieved after 6 time-periodic MGRIT iterations.

Considering a refined space-time discretization, the same observation holds true although the initial MGRIT error is already smaller than the error obtained from sequential time-stepping.

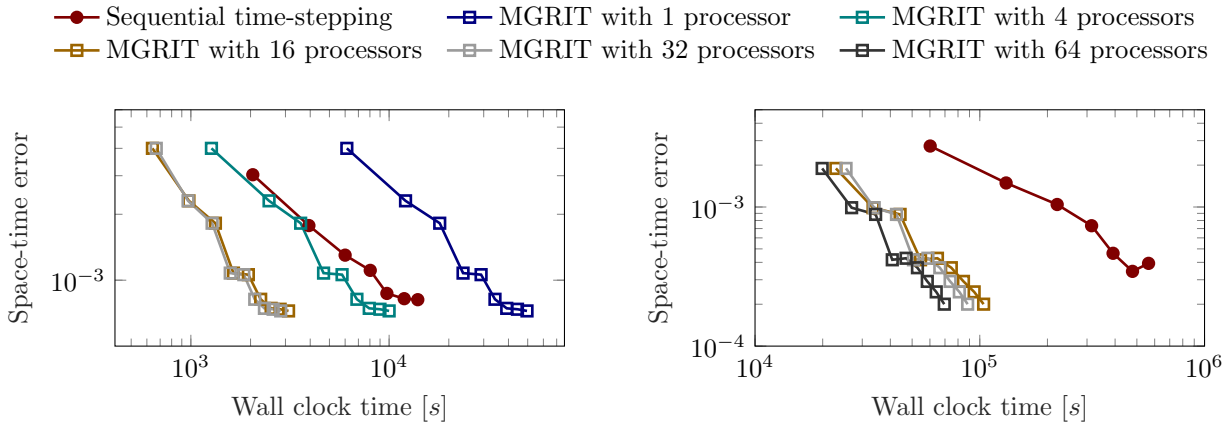


(a) Coarse mesh.

(b) Fine mesh.

Figure 13: Space-time error reduction (velocity) for sequential time-stepping over 7 cycles and time-periodic MGRIT over 9 iterations.

Parallel performance. Comparing wall clock times[17] of sequential time-stepping and MGRIT for the *coarse* mesh (see Figure 14a), it can be seen that achieved speedups are not as large as for the linear FSI case. However, significant speedups over sequential time-stepping are achieved when using 16 processors to parallelize in the temporal domain. Depending on the required accuracy (see Figure 14a and Table 6), the speedup ranges between 2.96x and 6.20x using 16 processors. The largest achieved speedup is 6.64x using 32 processors. Thus, the benefit of adding more parallelism beyond 16 processors is negligible, while in the linear FSI case, a larger additional speedup can be achieved. This is likely due to the nonlinear solver taking more iterations toward the end of the temporal domain, and thus, causing less optimal load balancing. To achieve an even larger speedup at a higher number of processors, a different decomposition of the temporal domain could be considered that takes the cost of each time step (with respect to its position along the temporal domain) into account.



(a) Coarse mesh: The largest achieved speedup is 6.20x using 16 processors and 6.64x using 32 processors. See Table 6.

(b) Fine mesh: The largest achieved speedup is 7.73x using 32 processors and 9.60x using 64 processors. See Table 7.

Figure 14: Space-time error (velocity) compared to wall clock time of sequential time stepping and time-periodic MGRIT for two different mesh sizes when using a zero initial condition for the fluid velocity and the analytic solution for all other variables. Note, that the MGRIT run with 1 processor was omitted for the fine mesh.

Now considering the *fine* mesh, a larger number of processors can be employed. Figure 14b illustrates accuracy (for the velocity approximation) compared to associated cost for sequential time-stepping, and time-periodic MGRIT with 16, 32 and 64 processors in time. Again, it is possible to rapidly reduce the space-time error with time-periodic MGRIT, giving the new algorithm a significant edge over sequential time-stepping in terms of wall clock time. Considering a comparable space-time error after 6 cycles (sequential time-stepping) and 6 iterations (time-periodic MGRIT), observed speedups range from 6.40x using 16 processors to 9.05x using 64 processors, see Table 7. It is further noted, that a small increase of the space-time error is observed between cycle 5 and 6 for sequential time-stepping, which similarly occurs for the time-periodic MGRIT algorithm between iteration 4 and 5. After iteration 5, however, the time-periodic MGRIT algorithm can further reduce the space-time error, while the wall clock time is still smaller than for sequential time-stepping.

4.3. Flow in a left atrium / left ventricle geometry

Figure 15a illustrates the flow in the left atrium and ventricle during diastole and systole for cycle 10, as obtained from sequential time-stepping. During diastole, flow through the pulmonary veins causes a slight spiraling of flow in the left atrium and a filling of the left ventricle through the mitral valve. Peak velocities

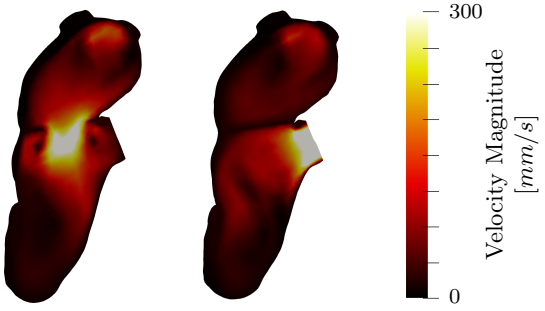
[17] Reported runtimes for ORCA (*coarse* mesh) and TOM (*fine* mesh).

occur at the pulmonary vein boundaries and the mitral valve section. The peak flow through the MV is approximately 46.18 cm/s with respective Reynolds number $Re \approx 3147.74$, which underpins the choice of a stabilized Galerkin scheme.

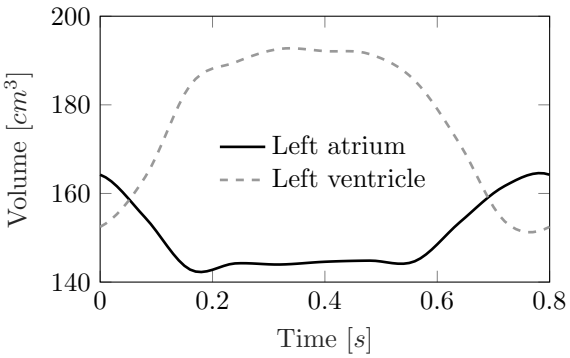
Further, the strong jet-like flow through the mitral valve opening causes a circular flow structure to develop around the jet with reflow regions between the jet and the endocardial wall, which moves towards the apex of the ventricle after the MV closes. During systole, the vortices near the MV decelerate and blood is ejected through the aortic valve boundary.

Considering the volume of the LV over time, see Figure 15b, the end-diastolic volume [59] at $t = 0.37$ s is approximately, $EDV \approx 192.50$ cm^3, and the end-systolic volume at $t = 0.8$ s is approximately, $ESV \approx 152.46$ cm^3. Thus, the stroke volume is, $SV = EDV - ESV \approx 40.04$ cm^3, and the left ventricular ejection fraction is, $LVEF = 100\ SV/EDV \approx 20.80$ %.

Therefore, according to the LVEF categories defined in the *European Society of Cardiology Guidelines 2016* [60, Section 3.2.1], the LVEF is significantly reduced. Further, the SV is reduced and the EDV and ESV are high compared to the normal ranges [61, Table 1] in healthy individuals.



(a) Velocity magnitude during diastole at $t = 0.16$ s and during systole at $t = 0.64$ s at a section cutting the MV surface and AV outflow boundary. Figure from [53, Figure 3].



(b) Fluid volume in left atrium and left ventricle over one cardiac cycle.

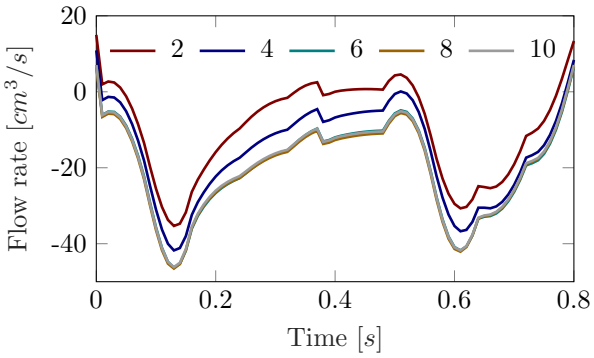
Figure 15: Flow magnitude during diastole and systole (left) and fluid volume over time (right) in the left atrium and left ventricle, respectively.

4.3.1. Time-periodic steady-state

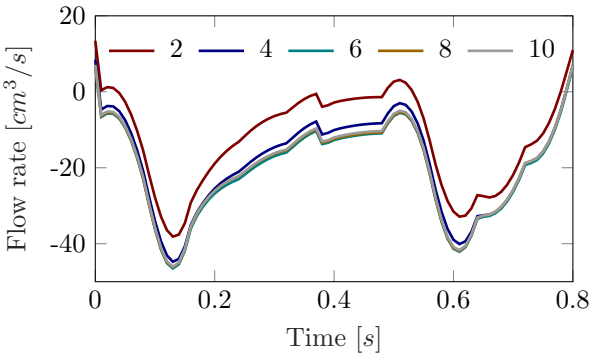
Figure 16a illustrates the flow rate at pulmonary vein 2 over the duration of one cycle for all even numbered cycles, as obtained with sequential time-stepping. The graph highlights that it is necessary to run multiple cycles to minimize cycle-to-cycle variations and obtain a time-periodic steady-state. Furthermore, the flow rate at pulmonary vein 1 converges to a periodic steady-state quicker than the flow rate at pulmonary vein 4; however, a periodic steady-state is obtained at each vein, see Figure SM1a.

On the other hand, Figure 16b highlights a similar convergence behavior when the time-periodic MGRIT algorithm is employed to obtain a periodic steady-state solution. In contrast to the sequential time-stepping solution, the initial error in the flow rates can be reduced slightly more quickly, compare Figure SM1a and Figure SM1b. It seems reasonable to assume that this is achieved by providing a better initial guess of the initial condition by means of XBraid's *skip-first-down* option and by updating the initial condition multiple times during each time-periodic MGRIT iteration (see Section 2.4).

Selecting the space-time solution for cycle 10 as the time-periodic steady-state solution, the error for each cycle and each iteration can be quantified. Figure 17 highlights that a time-periodic steady-state can be achieved by running sequential time-stepping for 7 cycles or time-periodic MGRIT for 6 iterations. Thus, in terms of convergence to a periodic steady-state, one time-periodic MGRIT iteration roughly corresponds to running one cycle using sequential time-stepping similar to Section 4.1 and Section 4.2.

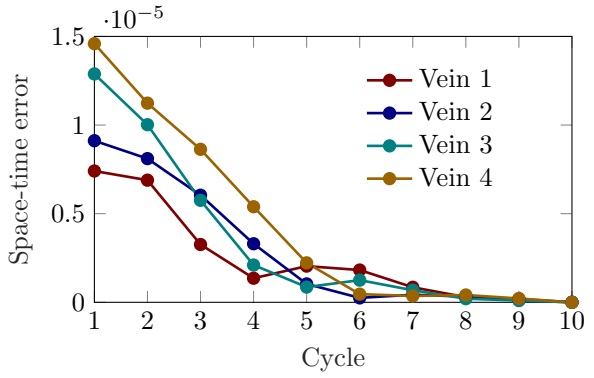


(a) A time-periodic steady-state is achieved by running multiple cycles using sequential time stepping.

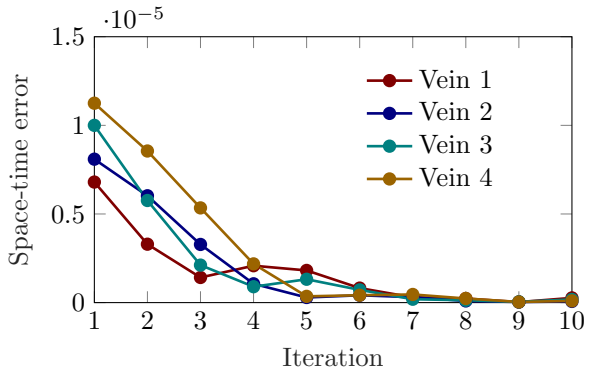


(b) A time-periodic steady-state is achieved by running multiple iterations using time-periodic MGRIT.

Figure 16: Flow rate at pulmonary vein 2 for even-numbered sequential time-stepping cycles and time-periodic MGRIT iterations. Note, a negative flow rate corresponds to inflow.



(a) Per-cycle error compared to cycle 10.



(b) Per-iteration error compared to cycle 10.

Figure 17: Convergence of flow rate errors at veins $1-4$ for sequential time-stepping (left) and time-periodic MGRIT (right). Reference data were selected as the sequential time-stepping solution over cycle 10.

4.3.2. Parallel performance

Here, runtimes and errors for sequential time-stepping and time-periodic MGRIT are compared. Timing results were obtained on TheoSim (see Appendix SM3.1) with 8 or 16 allocated processors per node.[18] Space-time errors are those reported in Figure 17 for vein 2, which are representative for all veins, see [39].

As Figure 18 illustrates, spatial parallelism can reduce the wall clock time for sequential time-stepping. For example, the speedup of switching from 8 processors to 16 processors in space is 1.58x, see Table 2. Spatial parallelism, however, saturates with a subsequent increase of the *time-to-solution* for 32 processors compared to 16 processors in space, which is also related to induced *node-to-node* communication for the employed hardware.

For any required space-time error and the considered distribution of processors for time- or space-time parallelism (see Table 2), time-periodic MGRIT yields a shorter time-to-solution than sequential time-stepping. The results reported in Figure 18 and Table 2 further highlight that it is beneficial to assign more processors for parallelization in the temporal domain instead of the spatial domain. For example, when using a total number of 80 processors, the wall clock time for time-periodic MGRIT can be reduced when

[18] Although each TheoSim node has 20 processors, preliminary experiments showed that performance deteriorates beyond 16 processors per node.

more processors are assigned to the temporal component than the spatial component.

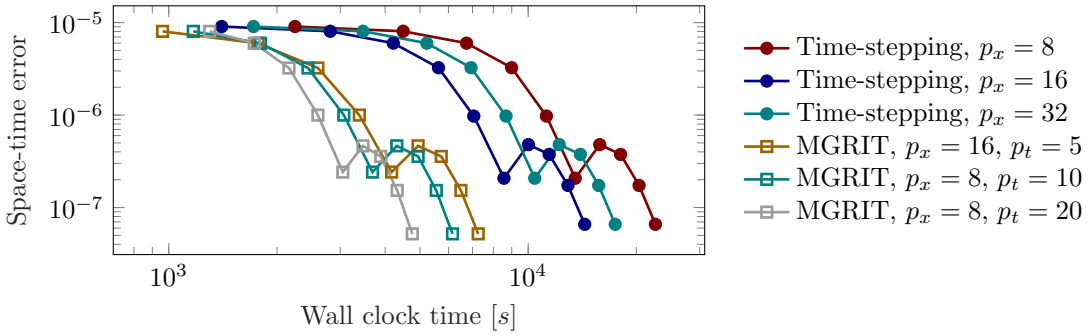


Figure 18: Space-time flow rate error at vein 2 compared to wall clock time for sequential time stepping and time-periodic MGRIT. MGRIT is consistently faster than sequential time-stepping and using a larger number of processors in time (denoted as p_t) instead of space (denoted as p_x) reduces the wall clock time of time-periodic MGRIT.

Algorithm	Number of cycles	Number of iterations	Number of processors p_x	p_t	$p_x p_t$	Wall clock time $[s]$	Speedup vs. 7 cycles
Time-stepping	7	-	8	-	8	15814 s	
	7	-	16	-	16	10017 s	1.58x
	7	-	32	-	32	12199 s	1.30x
MGRIT	1	6	16	5	80	4940 s	3.20x
	1	6	8	10	80	4310 s	3.67x
	1	6	8	20	160	3470 s	4.56x

Table 2: Runtimes and respective speedups for sequential time-stepping (7 cycles) and time-periodic MGRIT (6 iterations) with p_x processors in space and p_t processors in time.

5. Discussion

In this work, a new time-periodic MGRIT algorithm was proposed as a modification of the existing non-periodic MGRIT algorithm. The key ingredient is that only the fine grid is made periodic, such that the initial condition is continuously updated as an improved approximation of the time-periodic steady-state solution at the initial / cycle time becomes available. This allows the algorithm to naturally solve a time-periodic space-time problem and accelerates convergence to a time-periodic steady-state solution across the whole temporal domain, yielding significant runtime reductions compared to sequential time-stepping. Existing benefits of the MGRIT algorithm (and its implementations) remain intact, e.g., its nonintrusiveness, the ability to solve linear and nonlinear problems, the use of various cycling strategies and relaxation schemes, the ability to parallelize in time, in space and in space-time, etc.

The new time-periodic MGRIT algorithm was applied to a variety of application classes (see Section 3): two-dimensional linear flow in a complex geometry, linear and nonlinear fluid-structure interaction in two and three dimensions and coupled nonlinear flow in a three-dimensional left atrium / left ventricle geometry. For each of the application classes, it was demonstrated that applying the time-periodic MGRIT algorithm to a single time period yields the same time-periodic steady-state solution as sequential time-stepping over multiple time periods, see Section 4. It was repeatedly observed that running one sequential time-stepping cycle is roughly equivalent to performing one time-periodic MGRIT iteration with respect to convergence to a time-periodic steady-state, which yields an approximation of the required MGRIT iterations for the new algorithm. That is, many researchers know from experience how many sequential time-stepping cycles are required to obtain a time-periodic steady-state solution, which directly translates into the number of MGRIT iterations that need to be performed.

The property that one sequential time-stepping cycle and one time-periodic MGRIT iteration are equivalent w.r.t. steady-state convergence is ideal in the sense that each MGRIT iteration can update the initial condition at relatively cheaper cost compared to running one cycle using sequential time-stepping, see Section 4, because the underlying multigrid algorithm achieves convergence in the interior of the temporal domain. This yielded robust and significant speedups over sequential time-stepping irrespective of the various employed computing resources, ultimately enabling more detailed and more complex simulations.

The question why the time-periodic MGRIT algorithm can yield better speedups compared to those reported in previous works for similar PDE models [58, 41] has multiple answers. For example, exploiting the periodicity reduces the size of the temporal domain for time-periodic MGRIT, which generally improves the effectiveness of multigrid reduction within the MGRIT algorithm. Furthermore, (approximate) updates of the initial condition are communicated as early as possible, which accelerates the propagation of information across the whole temporal domain. Additionally, time-periodic MGRIT is jumpstarted by using an initial space-time guess obtained from a coarse-grid solve during the setup phase, which means that the first time-periodic MGRIT iteration starts off from an already improved initial condition at the initial time point.

Another surprising observation is that in some cases, time-periodic MGRIT without parallelism can outperform sequential time-stepping, see Section 4.2.1. Accounting for the explanations given in the previous paragraph and taking into consideration that the time-periodic MGRIT algorithm is applied to a much smaller temporal domain, one can appreciate the leeway for MGRIT to outperform sequential time-stepping. A similar result was reported in [30], where a waveform relaxation-based algorithm was able to outperform the standard algorithm for an initial boundary-value problem when no parallelism was employed.

For all considered applications, the time-periodicity was boundary driven and the period length known a priori. It was, however, not assessed how the time-periodic MGRIT algorithm can be applied to time-periodic problems where the period length is only known approximately or even unknown. This aspect is part of future research.

Similarly, the influence of the choice of (e.g., material) parameters on the convergence and observed speedup of the time-periodic MGRIT algorithm was well beyond the scope of this work, however, should be investigated in the future. Related works are, e.g., [62, 63, 64].

All considered problems were relatively small in space, in part to show speedups without requiring large clusters. Spatial refinement, and larger spatial problems in general, will help improve strong scaling because solving a time step will become relatively more expensive [41]. Therefore, communication overhead due to time parallelism will become relatively smaller for larger spatial problems.

While some explorations with respect to analyzing and estimating the convergence rate of *time-periodic* MGRIT in the nonlinear PDE case were made as part of developing the proposed method [39], no mathematically rigorous a priori convergence estimates are available and should be part of future research. It is, however, highlighted that the available convergence bounds for non-periodic MGRIT [25, 26, 27, 29] are a useful tool for improving the algorithm's performance and informing parameter choices a priori.

Traditional (non-periodic) MGRIT is not restricted to the simple two-level case and neither is the new time-periodic MGRIT variant. In fact, the modification in the MGRIT algorithm is limited to the fine-grid. Therefore, the capability of MGRIT to use one or multiple coarse-grids and thus exploit a greater level of parallelism to further reduce the time-to-solution is maintained. Exploring the true multilevel case, however, was beyond the scope of this work and needs to be investigated in the future.[19]

6. Conclusion

In this work, we have proposed a time-periodic MGRIT algorithm as a means to reduce the time-to-solution of numerical algorithms by exploiting the time periodicity inherent to many applications in science and engineering. The time-periodic MGRIT algorithm was proven to be applicable to a variety of linear and nonlinear single- and multiphysics problems. It was demonstrated that using the new parallel-in-time algorithm yields a significant runtime reduction compared to sequential time-stepping. Robust speedups

[19] Preliminary results are available in [39].

were observed across the various flow and fluid-structure interaction problems in two and three dimensions, for a variety of space-time discretizations and employed computing hardware. It was further demonstrated that a time-periodic steady-state can be obtained with respect to a given tolerance by either running q sequential time-stepping cycles or q time-periodic MGRIT iterations. Thus, an intuition about the required number of sequential time-stepping cycles directly translates into the number of MGRIT iterations that need to be performed to achieve a periodic steady-state.

7. Acknowledgements

D.N. would like to acknowledge funding from Engineering and Physical Sciences Research Council (EP/N011554/1 and EP/R003866/1).

O.R. and A.H. were funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 - 390740016. We acknowledge the support by the Stuttgart Center for Simulation Science (SimTech).

Additional Information

Declarations of interest: none.

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

References

[1] P. J. Roache, Marching methods for elliptic problems: Part 1, Numerical Heat Transfer 1 (1) (1978) 1–25.

[2] P. J. Roache, Elliptic marching methods and domain decomposition, Vol. 5, CRC press, 1995.

[3] A. M. Quarteroni, A. Valli, Domain decomposition methods for partial differential equations, Oxford University Press, 1999.

[4] J. Nievergelt, Parallel methods for integrating ordinary differential equations, Commun. ACM 7 (12) (1964) 731–733.
URL http://dx.doi.org/10.1145/355588.365137

[5] M. J. Gander, 50 years of Time Parallel Time Integration, in: Multiple Shooting and Time Domain Decomposition, Springer, 2015.
URL http://dx.doi.org/10.1007/978-3-319-23321-5_3

[6] B. W. Ong, J. B. Schroder, Applications of time parallelization, Computing and Visualization in Science 23 (1-4). doi:10.1007/s00791-020-00331-4.
URL https://doi.org/10.1007%2Fs00791-020-00331-4

[7] C. Lubich, A. Ostermann, Multi-grid dynamic iteration for parabolic equations, BIT Numerical Mathematics 27 (2) (1987) 216–234.
URL http://dx.doi.org/10.1007/BF01934186

[8] S. Vandewalle, E. Van de Velde, Space-time concurrent multigrid waveform relaxation, Annals of Numerical Mathematics 1 (1-4) (1994) 347–360.
URL http://dx.doi.org/10.13140/2.1.1146.1761

[9] G. Horton, S. Vandewalle, A Space-Time Multigrid Method for Parabolic Partial Differential Equations, SIAM Journal on Scientific Computing 16 (4) (1995) 848–864.
URL http://dx.doi.org/10.1137/0916050

[10] C. Farhat, M. Chandesris, Time-decomposed parallel time-integrators: theory and feasibility studies for fluid, structure, and fluid-structure applications, International Journal for Numerical Methods in Engineering 58 (9) (2003) 1397–1434.
URL http://dx.doi.org/10.1002/nme.860

[11] C. Farhat, J. Cortial, C. Dastillung, H. Bavestrello, Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses, International Journal for Numerical Methods in Engineering 67 (2006) 697–724.
URL http://dx.doi.org/10.1002/nme.1653

[12] A. J. Christlieb, C. B. Macdonald, B. W. Ong, Parallel high-order integrators, SIAM Journal on Scientific Computing 32 (2) (2010) 818–835.
URL http://dx.doi.org/10.1137/09075740X

[13] R. Speck, D. Ruprecht, R. Krause, M. Emmett, M. L. Minion, M. Winkel, P. Gibbon, A massively space-time parallel N-body solver, in: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12, IEEE Computer Society Press, Los Alamitos, CA, USA, 2012, pp. 92:1–92:11.
URL http://dx.doi.org/10.1109/SC.2012.6

[14] M. Emmett, M. L. Minion, Toward an Efficient Parallel in Time Method for Partial Differential Equations, Communications in Applied Mathematics and Computational Science 7 (2012) 105–132.
URL http://dx.doi.org/10.2140/camcos.2012.7.105

[15] F. P. Hamon, M. Schreiber, M. L. Minion, Multi-level spectral deferred corrections scheme for the shallow water equations on the rotating sphere, Journal of Computational Physics 376 (2019) 435–454.

[16] J.-L. Lions, Y. Maday, G. Turinici, A "parareal" in time discretization of PDE's, Comptes Rendus de l'Académie des Sciences - Series I - Mathematics 332 (2001) 661–668.
URL http://dx.doi.org/10.1016/S0764-4442(00)01793-6

[17] S. Friedhoff, R. D. Falgout, T. V. Kolev, S. MacLachlan, J. B. Schroder, A multigrid-in-time algorithm for solving evolution equations in parallel, Tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States) (2012).

[18] R. D. Falgout, S. Friedhoff, T. V. Kolev, S. P. MacLachlan, J. B. Schroder, Parallel time integration with multigrid, SIAM Journal on Scientific Computing 36 (6) (2014) C635–C661.

[19] M. J. Gander, Y.-L. Jiang, B. Song, H. Zhang, Analysis of two parareal algorithms for time-periodic problems, SIAM Journal on Scientific Computing 35 (5) (2013) A2393–A2415.

[20] P. Benedusi, D. Hupp, P. Arbenz, R. Krause, A Parallel Multigrid Solver for Time–periodic Incompressible Navier–Stokes Equations in 3D, in: Numerical Mathematics and Advanced Applications ENUMATH 2015, Springer, 2016, pp. 265–273. doi:10.1007/978-3-319-39929-4_26.
URL https://doi.org/10.1007/978-3-319-39929-4_26

[21] S. Friedhoff, S. P. MacLachlan, A generalized predictive analysis tool for multigrid methods, Numerical Linear Algebra with Applications 22 (4) (2015) 618–647.

[22] S. Friedhoff, B. S. Southworth, On "Optimal" h-Independent Convergence of Parareal and MGRIT using Runge-Kutta Time Integration, arXiv preprint arXiv:1906.06672.

[23] O. A. Krzysik, H. De Sterck, S. P. MacLachlan, S. Friedhoff, On selecting coarse-grid operators for Parareal and MGRIT applied to linear advection, arXiv preprint arXiv:1902.07757.

[24] H. De Sterck, S. Friedhoff, A. J. M. Howse, S. P. MacLachlan, Convergence analysis for parallel-in-time solution of hyperbolic systems, arXiv preprint arXiv:1903.08928.

[25] V. A. Dobrev, T. V. Kolev, N. A. Petersson, J. B. Schroder, Two-level convergence theory for multigrid reduction in time (MGRIT), SIAM Journal on Scientific Computing 39 (5) (2017) S501–S527.

[26] B. S. Southworth, Necessary conditions and tight two-level convergence bounds for Parareal and multigrid reduction in time, SIAM J. on Matrix Analysis and Applications.

[27] A. Hessenthaler, B. S. Southworth, D. Nordsletten, O. Röhrle, R. D. Falgout, J. B. Schroder, Multilevel convergence analysis of multigrid-reduction-in-time, SIAM Journal on Scientific Computing 42 (2) (2020) A771–A796. arXiv:https://doi.org/10.1137/19M1238812, doi:10.1137/19M1238812.
URL https://doi.org/10.1137/19M1238812

[28] M. J. Gander, T. Lunet, Toward error estimates for general space-time discretizations of the advection equation, Computing and Visualization in Science 23 (1) (2020) 1–14.

[29] B. S. Southworth, W. Mitchell, A. Hessenthaler, F. Danieli, Tight two-level convergence of Linear Parareal and MGRIT: Extensions and Implications in Practice.

[30] S. Vandewalle, R. Piessens, Efficient Parallel Algorithms for Solving Initial-Boundary Value and Time-Periodic Parabolic Partial Differential Equations, SIAM Journal on Scientific and Statistical Computing 13 (6) (1992) 1330–1346. doi:10.1137/0913075.
URL http://dx.doi.org/10.1137/0913075

[31] M. J. Gander, I. Kulchytska-Ruchka, S. Schöps, A new parareal algorithm for time-periodic problems with discontinuous inputs, arXiv preprint arXiv:1810.12372.

[32] J. Christopher, R. D. Falgout, J. B. Schroder, S. M. Guzik, X. Gao, A space-time parallel algorithm with adaptive mesh refinement for computational fluid dynamics, Computing and Visualization in Science 23 (1-4) (2020) 13, lLNL-JRNL-798697. doi:10.1007/s00791-020-00334-1.

[33] J. Christopher, X. Gao, S. M. Guzik, R. D. Falgout, J. B. Schroder, Parallel in time for a fully space-time adaptive mesh refinement algorithm, in: AIAA SciTech 2020 Forum, 2020, lLNL-CONF-800089. arXiv:https://arc.aiaa.org/doi/pdf/10.2514/6.2020-0340, doi:10.2514/6.2020-0340.
URL https://arc.aiaa.org/doi/abs/10.2514/6.2020-0340

[34] B. Song, Y.-L. Jiang, Analysis of a new parareal algorithm based on waveform relaxation method for time-periodic problems, Numerical Algorithms 67 (3) (2014) 599–622. doi:10.1007/s11075-013-9810-z.
URL http://dx.doi.org/10.1007/s11075-013-9810-z

[35] M. J. Gander, I. Kulchytska-Ruchka, S. Schöps, A new parareal algorithm for time-periodic problems with discontinuous inputs, in: Lecture Notes in Computational Science and Engineering, Springer International Publishing, 2020, pp. 243–250. doi:10.1007/978-3-030-56750-7_27.
URL https://doi.org/10.1007%2F978-3-030-56750-7_27

[36] A. Brandt, S. F. McCormick, J. Ruge, Algebraic multigrid (amg) for sparse matrix equations, Sparsity and its Applications 257.

[37] W. L. Briggs, V. E. Henson, S. F. McCormick, A multigrid tutorial, 2nd Edition, SIAM, Philadelphia, PA, USA, 2000.

[38] W. Hackbusch, U. Trottenberg, Multigrid methods: proceedings of the conference held at Köln-Porz, November 23-27, 1981, Vol. 960, Springer, 2006.
[39] A. Hessenthaler, Multilevel Convergence Analysis: Parallel-in-Time Integration for Fluid-Structure Interaction Problems with Applications in Cardiac Flow Modeling, Ph.D. thesis, Institute for Modelling and Simulation of Biomechanical Systems, University of Stuttgart (2020). doi:http://dx.doi.org/10.18419/opus-11260.
[40] XBraid: Parallel multigrid in time, http://llnl.gov/casc/xbraid.
[41] A. Hessenthaler, D. Nordsletten, O. Röhrle, J. B. Schroder, R. D. Falgout, Convergence of the multigrid reduction in time algorithm for the linear elasticity equations, Numerical Linear Algebra with Applications 25 (3) (2018) e2155, e2155 nla.2155. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/nla.2155, doi:10.1002/nla.2155.
URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nla.2155
[42] E. H. van Brummelen, K. G. van der Zee, R. de Borst, Space/time multigrid for a fluid–structure-interaction problem, Applied Numerical Mathematics 58 (12) (2008) 1951–1971.
[43] T. Richter, A monolithic geometric multigrid solver for fluid-structure interactions in ale formulation, International journal for numerical methods in engineering 104 (5) (2015) 372–390.
[44] U. Trottenberg, C. W. Oosterlee, A. Schüller, Multigrid, Academic Press, 2001.
[45] J. Lee, A. Cookson, I. Roy, E. Kerfoot, L. Asner, G. Vigueras, T. Sochi, S. Deparis, C. Michler, N. P. Smith, et al., Multiphysics computational modeling in CHeart, SIAM Journal on Scientific Computing 38 (3) (2016) C150–C178.
[46] CHeart: A scientific software application for simulating the physics of the human heart, http://cheart.co.uk.
[47] A. Hessenthaler, O. Röhrle, D. Nordsletten, Validation of a non-conforming monolithic fluid-structure interaction method using phase-contrast MRI, International Journal for Numerical Methods in Biomedical Engineering 33 (8).
[48] P. R. Amestoy, I. S. Duff, J.-Y. L'excellent, Multifrontal parallel distributed symmetric and unsymmetric solvers, Computer Methods in Applied Mechanics and Engineering 184 (2-4) (2000) 501–520.
[49] A. Hessenthaler, M. Balmus, O. Röhrle, D. Nordsletten, A Class of Analytic Solutions for Verification and Convergence Analysis of Linear and Nonlinear Fluid-Structure Interaction Algorithms, Computational Methods in Mathematics and Engineering.
[50] D. Nordsletten, D. Kay, N. Smith, A non-conforming monolithic finite element method for problems of coupled mechanics, Journal of Computational Physics 229 (20) (2010) 7571–7593.
[51] V. Shamanskii, A modification of Newton's method, Ukrainian Mathematical Journal 19 (1) (1967) 118–122.
[52] R. A. Nishimura, F. A. Miller Jr, M. J. Callahan, R. C. Benassi, J. B. Seward, A. J. Tajik, Doppler echocardiography: theory, instrumentation, technique, and application, in: Mayo Clinic Proceedings, Vol. 60, Elsevier, 1985, pp. 321–343.
[53] D. Marlevi, M. Balmus, A. Hessenthaler, F. Viola, D. Fovargue, A. de Vecchi, P. Lamata, N. S. Burris, F. D. Pagani, J. Engvall, E. R. Edelman, T. Ebbers, D. A. Nordsletten, Non-invasive estimation of relative pressure for intracardiac flows using virtual work-energy, Medical Image Analysis 68 (2021) 101948. doi:https://doi.org/10.1016/j.media.2020.101948.
URL http://www.sciencedirect.com/science/article/pii/S1361841520303121
[54] M. Balmus, A. Massing, J. Hoffman, R. Razavi, D. A. Nordsletten, A partition of unity approach to fluid mechanics and fluid–structure interaction, Computer methods in applied mechanics and engineering 362 (2020) 112842.
[55] Y. Bazilevs, J. R. Gohean, T. J. R. Hughes, R. D. Moser, Y. Zhang, Patient-specific isogeometric fluid-structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik 2000 left ventricular assist device, Computer Methods in Applied Mechanics and Engineering 198 (45-46) (2009) 3534–3550.
[56] M. E. Moghadam, Y. Bazilevs, T.-Y. Hsia, I. E. Vignon-Clementel, A. L. Marsden, et al., A comparison of outlet boundary treatments for prevention of backflow divergence with relevance to blood flow simulations, Computational Mechanics 48 (3) (2011) 277–291.
[57] J. Hoffman, J. Jansson, R. V. De Abreu, Adaptive modeling of turbulent flow with residual based turbulent kinetic energy dissipation, Computer Methods in Applied Mechanics and Engineering 200 (37-40) (2011) 2758–2767.
[58] R. D. Falgout, A. Katz, T. V. Kolev, J. B. Schroder, A. Wissink, U. M. Yang, Parallel time integration with multigrid reduction for a compressible fluid dynamics application, Tech. rep., Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States) (2015).
[59] J. Holt, Estimation of the residual volume of the ventricle of the dog's heart by two indicator dilution technics, Circulation Research 4 (2) (1956) 187–195.
[60] P. Ponikowski, A. A. Voors, S. D. Anker, H. Bueno, J. G. F. Cleland, A. J. S. Coats, V. Falk, J. R. Gonzlez-Juanatey, V.-P. Harjola, E. A. Jankowska, M. Jessup, C. Linde, P. Nihoyannopoulos, J. T. Parissis, B. Pieske, J. P. Riley, G. M. C. Rosano, L. M. Ruilope, F. Ruschitzka, F. H. Rutten, P. van der Meer, E. S. D. Group, 2016 ESC guidelines for the diagnosis and treatment of acute and chronic heart failure, European Heart Journal 37 (27) (2016) 2129–2200. arXiv:http://oup.prod.sis.lan/eurheartj/article-pdf/37/27/2129/23748755/ehw128.pdf, doi:10.1093/eurheartj/ehw128.
URL https://doi.org/10.1093/eurheartj/ehw128
[61] M. Lomsky, L. Johansson, P. Gjertsson, J. Björk, L. Edenbrandt, Normal limits for left ventricular ejection fraction and volumes determined by gated single photon emission computed tomography - a comparison between two quantification methods, Clinical Physiology and Functional Imaging 28 (3) (2008) 169–173.
[62] J. Steiner, D. Ruprecht, R. Speck, R. Krause, Convergence of parareal for the navier-stokes equations depending on the reynolds number, in: Numerical Mathematics and Advanced Applications-ENUMATH 2013, Springer, 2015, pp. 195–202.
[63] M. J. Gander, Five decades of time parallel time integration, and a note on the degradation of the performance of the parareal algorithm as a function of the reynolds number, Oberwolfach Report.
[64] M. J. Gander, T. Lunet, A reynolds number dependent convergence estimate for the parareal algorithm, in: International Conference on Domain Decomposition Methods, Springer, 2018, pp. 277–284.

Supplementary Material 1:
Time-periodic steady-state solution of fluid-structure interaction and cardiac flow problems through multigrid-reduction-in-time

Andreas Hessenthaler[a,b], Robert D. Falgout[c], Jacob B. Schroder[d], Adelaide de Vecchi[e], David Nordsletten[e,f], Oliver Röhrle[a,b]

[a] *Institute for Modelling and Simulation of Biomechanical Systems, University of Stuttgart, Pfaffenwaldring 5a, 70569 Stuttgart, Germany*
[b] *Stuttgart Center for Simulation Technology, University of Stuttgart, Pfaffenwaldring 5a, 70569 Stuttgart, Germany*
[c] *Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, Livermore, CA 94551*
[d] *Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM 87131*
[e] *School of Biomedical Engineering and Imaging Sciences, King's College London, 4th FL Rayne Institute, St Thomas Hospital, London, SE1 7EH*
[f] *Department of Biomedical Engineering and Cardiac Surgery, University of Michigan, NCRC B20, 2800 Plymouth Rd, Ann Arbor, 48109*

SM1. Multigrid-reduction-in-time (MGRIT)

$$
\begin{aligned}
RA_0P &= R\begin{bmatrix} I & & & & \\ -\Phi_0 & I & & & \\ & -\Phi_0 & I & & \\ & & \ddots & \ddots & \\ & & & -\Phi_0 & I \end{bmatrix}\begin{bmatrix} [I,\Phi_0,\cdots,\Phi_0^{m-1}]^T & & & \\ & [I,\Phi_0,\cdots,\Phi_0^{m-1}]^T & & \\ & & \ddots & \\ & & & I \end{bmatrix} \\
&= \begin{bmatrix} I & & & & & & & & \\ & \Phi_0^{m-1} & \Phi_0^{m-2} & \cdots & I & & & & \\ & & & & & \ddots & & & \\ & & & & & & \Phi_0^{m-1} & \Phi_0^{m-2} & \cdots & I \end{bmatrix}\begin{bmatrix} I & & & & \\ 0 & & & & \\ \vdots & & & & \\ 0 & & & & \\ -\Phi_0^m & I & & & \\ & 0 & & & \\ & \vdots & & & \\ & 0 & & & \\ & -\Phi_0^m & I & & \\ & & \ddots & \ddots & \\ & & & -\Phi_0^m & I \end{bmatrix} \\
&= \begin{bmatrix} I & & & & \\ -\Phi_0^m & I & & & \\ & -\Phi_0^m & I & & \\ & & \ddots & \ddots & \\ & & & -\Phi_0^m & I \end{bmatrix} = R_I A_0 P = \begin{bmatrix} I & & & & & & & & & \\ & 0 & 0 & \cdots & I & & & & & \\ & & & & & \ddots & & & & \\ & & & & & & 0 & 0 & \cdots & 0 & I \end{bmatrix} A_0 P
\end{aligned}
\tag{SM1}
$$

SM2. Flow in a left atrium / left ventricle geometry: Weak formulation

Finite element discretizations were constructed using $\mathbb{P}^1 - \mathbb{P}^1$ elements for fluid velocity and pressure and $\mathbb{P}^1$ elements for the Lagrange multipliers on the coupling domain, resulting in 55842 degrees-of-freedom (DOFs). The discrete solution at each time step n can then be written as follows:

Find $\boldsymbol{s}^n := (\boldsymbol{v}_A^n, \boldsymbol{v}_V^n, \boldsymbol{\lambda}_A^n, \boldsymbol{\lambda}_V^n, p_A^n, p_V^n) \in \boldsymbol{\mathcal{S}}_D^h := \boldsymbol{\mathcal{V}}_D^h \times \boldsymbol{\mathcal{U}}_D^h \times \boldsymbol{\mathcal{M}}_0^h \times \boldsymbol{\mathcal{N}}_0^h \times \mathcal{W}_A^h \times \mathcal{W}_V^h$, such that for every $d := (\boldsymbol{y}_A, \boldsymbol{y}_V, \boldsymbol{\varphi}_A, \boldsymbol{\varphi}_V, q_A, q_V) \in \boldsymbol{\mathcal{S}}_0^h := \boldsymbol{\mathcal{V}}_0^h \times \boldsymbol{\mathcal{U}}_0^h \times \boldsymbol{\mathcal{M}}_0^h \times \boldsymbol{\mathcal{N}}_0^h \times \mathcal{W}_A^h \times \mathcal{W}_V^h$:

$$
\begin{aligned}
&R\left(\boldsymbol{s}^n, \boldsymbol{s}^{n-1}, \boldsymbol{w}_A^n, \boldsymbol{w}_V^n; \boldsymbol{d}\right) \\
&\quad := \int_{\Omega_A^n} \left[\frac{\boldsymbol{v}_A^n - \boldsymbol{v}_A^n}{\delta_0} + (\boldsymbol{v}_A^n - \boldsymbol{w}_A^n) \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}_A^n\right] \cdot \boldsymbol{y}_A \; d\boldsymbol{x} \\
&\qquad + \int_{\Omega_A^n} \boldsymbol{\sigma}_A^n : \nabla_{\boldsymbol{x}} \boldsymbol{y}_A + \boldsymbol{\varphi}_A \nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}_A^n \; d\boldsymbol{x} \\
&\qquad - \int_{\Gamma_{Vn}^n} \boldsymbol{t}_A^n \cdot \boldsymbol{y}_A \; d\boldsymbol{x} \\
&\qquad + \int_{\Omega_V^n} \left[\frac{\boldsymbol{v}_V^n - \boldsymbol{v}_V^n}{\delta_0} + (\boldsymbol{v}_V^n - \boldsymbol{w}_V^n) \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}_V^n\right] \cdot \boldsymbol{y}_V \; d\boldsymbol{x} \\
&\qquad + \int_{\Omega_V^n} \boldsymbol{\sigma}_V^n : \nabla_{\boldsymbol{x}} \boldsymbol{y}_V + \boldsymbol{\varphi}_V \nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}_A^n \; d\boldsymbol{x} \\
&\qquad - \alpha \int_{\Gamma_{AV}^n} \boldsymbol{t}_V^n \cdot \boldsymbol{y}_V \; d\boldsymbol{x} \\
&\qquad + \int_{\Omega_{LM}^0} \boldsymbol{\varphi}_A \cdot (\boldsymbol{v}_A^n - \alpha \boldsymbol{w}_A^n - (1-\alpha) \boldsymbol{v}_V^n) \; d\boldsymbol{x} \\
&\qquad + \int_{\Omega_{LM}^0} \alpha \boldsymbol{\varphi}_V \left(\boldsymbol{v}_A^n - \boldsymbol{v}_V^n\right) \; d\boldsymbol{x} \\
&\qquad + \int_{\Omega_{LM}^0} (1-\alpha) \left(\boldsymbol{\lambda}_A \cdot \boldsymbol{y}_A - \boldsymbol{\lambda}_V \cdot \boldsymbol{y}_V\right) \; d\boldsymbol{x} \\
&\qquad + \zeta_1 \int_{\Omega_A^n} \left[(\boldsymbol{v}_A^n - \boldsymbol{w}_A^n) \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}_A^n\right] \cdot \left[\boldsymbol{v}_A^n \cdot \nabla_{\boldsymbol{x}} \boldsymbol{y}_A + \nabla_{\boldsymbol{x}} q_A\right] \; d\boldsymbol{x} \\
&\qquad + \zeta_2 \int_{\Omega_A^n} (\nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}_A) \cdot (\nabla_{\boldsymbol{x}} \cdot \boldsymbol{y}_A) \; d\boldsymbol{x} \\
&\qquad + \zeta_3 \int_{\Omega_A^n} (\nabla_{\boldsymbol{x}} p_A^n) \cdot (\boldsymbol{v}_A^n \cdot \nabla_{\boldsymbol{x}} \boldsymbol{y}_A + \nabla_{\boldsymbol{x}} q_A) \; d\boldsymbol{x} \\
&\qquad + \zeta_1 \int_{\Omega_V^n} \left[(\boldsymbol{v}_V^n - \boldsymbol{w}_V^n) \cdot \nabla_{\boldsymbol{x}} \boldsymbol{v}_V^n\right] \cdot \left[\boldsymbol{v}_V^n \cdot \nabla_{\boldsymbol{x}} \boldsymbol{y}_V + \nabla_{\boldsymbol{x}} q_V\right] \; d\boldsymbol{x} \\
&\qquad + \zeta_2 \int_{\Omega_V^n} (\nabla_{\boldsymbol{x}} \cdot \boldsymbol{v}_V) \cdot (\nabla_{\boldsymbol{x}} \cdot \boldsymbol{y}_V) \; d\boldsymbol{x} \\
&\qquad + \zeta_3 \int_{\Omega_V^n} (\nabla_{\boldsymbol{x}} p_V^n) \cdot (\boldsymbol{v}_V^n \cdot \nabla_{\boldsymbol{x}} \boldsymbol{y}_V + \nabla_{\boldsymbol{x}} q_V) \; d\boldsymbol{x}
\end{aligned}
\tag{SM2}
$$

where the stabilization terms with parameters $\zeta_1 = \rho \delta_x^{max} / v_f^{max}$, $\zeta_2 = \rho \delta_x^{max} v_f^{max}$ and $\zeta_3 = \delta_x^{max} / v_f^{max}$ with $v_f^{max} = 100 \; cm/s$ are added according to the cG(1)cG(1) scheme [SM57].

Note, that the domain velocity $\boldsymbol{w}_A^n$ and $\boldsymbol{w}_V^n$ is provided from the CT data, see Section 3.3.

The definitions of the function spaces are:

$$
S^1\left(\Omega_i^0\right) = \{f : \Omega_i^0 \to \mathbb{R} \mid f \in \mathcal{C}^0(\bar{\Omega}_i^0), \; f|_{\tau_e} \in \mathbb{P}^1(\tau_e), \; \forall \; \tau_e \subset \mathcal{T}_i^h\}, \tag{SM3}
$$

which represent the first-order piecewise continuous polynomial spaces defined on Ω_i^0. Consequently, we can

define:

$$\begin{aligned}
\boldsymbol{\mathcal{V}}^h &= \left[S^1(\Omega_A^0)\right]^3, & \boldsymbol{\mathcal{U}}^h &= \left[S^1(\Omega_V^0)\right]^3,\\
\boldsymbol{\mathcal{M}}^h &= \left[S^1(\Gamma_{CA})\right]^3, & \boldsymbol{\mathcal{N}}^h &= \left[S^1(\Gamma_{VA})\right]^3,\\
\mathcal{W}_A^h &= S^1(\Omega_A^0), & \mathcal{W}_V^h &= S^1(\Omega_V^0).
\end{aligned}$$

Further restrictions are applied on the respective spaces for the atrium and ventricle in order to incorporate the Dirichlet and homogeneous boundary conditions:

$$\boldsymbol{\mathcal{V}}_D^h = \{\boldsymbol{v} \in \boldsymbol{\mathcal{V}}^h \mid \boldsymbol{v} = \boldsymbol{w}_A \text{ on } \Gamma_{WA}^h\}, \tag{SM4}$$

$$\boldsymbol{\mathcal{V}}_0^h = \{\boldsymbol{v} \in \boldsymbol{\mathcal{V}}^h \mid \boldsymbol{v} = \boldsymbol{0} \text{ on } \Gamma_{WA}^h\}, \tag{SM5}$$

$$\begin{aligned}
\boldsymbol{\mathcal{U}}_D^h = \{\boldsymbol{v} \in \boldsymbol{\mathcal{U}}^h \mid\ & \boldsymbol{v} = \boldsymbol{w}_V \text{ on } \Gamma_{WV}^h,\\
& \boldsymbol{v} = \boldsymbol{w}_V \text{ on } \Gamma_{AV}^h \text{ for } t \in (0, 0.37],\\
& \boldsymbol{v} = \boldsymbol{w}_V \text{ on } \Gamma_{CV}^h \setminus \Gamma_{CVo}^h \text{ for } t \in (0.37, 0.8]\},
\end{aligned} \tag{SM6}$$

$$\begin{aligned}
\boldsymbol{\mathcal{U}}_0^h = \{\boldsymbol{v} \in \boldsymbol{\mathcal{U}}^h \mid\ & \boldsymbol{v} = \boldsymbol{0} \text{ on } \Gamma_{WV}^h,\\
& \boldsymbol{v} = \boldsymbol{0} \text{ on } \Gamma_{AV}^h \text{ for } t \in (0, 0.37],\\
& \boldsymbol{v} = \boldsymbol{0} \text{ on } \Gamma_{CV}^h \setminus \Gamma_{CVo}^h \text{ for } t \in (0.37, 0.8]\},
\end{aligned} \tag{SM7}$$

and similarly for the coupling domains,

$$\boldsymbol{\mathcal{M}}_0^h = \{\boldsymbol{\lambda} \in \gamma_{\Gamma_{CA}} \boldsymbol{\mathcal{V}}_0^h\}, \tag{SM8}$$

$$\boldsymbol{\mathcal{N}}_0^h = \{\boldsymbol{\lambda} \in \gamma_{\Gamma_{CV}} \boldsymbol{\mathcal{U}}_0^h\}, \tag{SM9}$$

where $\gamma_{\Gamma_{CA}}$ and $\gamma_{\Gamma_{CV}}$ are the trace operators on Γ_{CA} and Γ_{CV}, respectively.

SM3. Tables

SM3.1. Parallel machines

Name	LEAD
Operating System	Scientific Linux 6.4 (Carbon)
Compute Nodes	1x AMD Opteron 6373 G34 2.3 GHz, 4x 16 cores, Abu Dhabi 7x AMD Opteron 6328 G34 3.2 GHz, 4x 8 cores, Abu Dhabi
Memory per Node	256 GB
Disks per Node	4 x 128 GB Samsung SSD
Network	2 x 10 GBit Intel 82599ES Ethernet Dual SFP+ 1 x Melanox MCX353A-QCBT, Infiniband single-port QSFP, IB 40 GB/s 2 x 1 GBit
Total Number of Processors	288

Table 1: Hardware and software specifications for LEAD cluster.

Name	TheoSim
Operating System	Scientific Linux 6.10 (Carbon)
Compute Nodes	52x Intel(R) Xeon(R) E5-2650 v2 2.8 GHz, 2x 10 cores, Ivy Bridge Available for project: 18 nodes
Memory per Node	256 GB
Disks per Node	4 x 500 GB Samsung SSD
Network	1 x QDR-Infiniband Interconnect (40 GBit) 1 x 1 GBit Ethernet
Total Number of Processors	1040 Available for project: 360

Table 2: Hardware and software specifications for TheoSim cluster.

Name	ORCA
Operating System	Ubuntu 16.04
Compute Nodes	1x AMD Ryzen Threadripper 2990WX 3.0 GHz, 32 cores
Memory per Node	128 GB
Disks per Node	3 x 1 TB Samsung NVME SSD
Total Number of Processors	32

Table 3: Hardware and software specifications for ORCA cluster.

Name	TOM
Operating System	Suse SLES 11 SP1
Compute Nodes	76x Intel(R) Xeon(R) E7-8837 2.66GHz, 8 cores, Westmere EX
Memory per Node	64 GB
Network	NUMAlink 5 Interconnect
Total Number of Processors	608

Table 4: Hardware and software specifications for TOM cluster.

Algorithm	p_x	p_t	$p_x p_t$	TheoSim Wall clock time $[s]$	TheoSim Speedup	LEAD Wall clock time $[s]$	LEAD Speedup
Sequential time-stepping	1	1	1	332	1.00x	523	
	8	1	8	64	5.19x	83	6.30x
	16	1	16	43	7.72x	56	9.34x
	32	1	32	260	1.28x		
MGRIT, F-relaxation, $m = 2$	16	1	16	323	1.03x	446	1.17x
MGRIT, F-relaxation, $m = 32$	4	1	4	202	1.64x	306	1.71x
MGRIT, F-relaxation, $m = 32$	16	1	16	78	4.26x	104	5.03x
MGRIT, F-relaxation, $m = 32$	32	1	32	76	4.37x		
MGRIT, F-relaxation, $m = 32$	16	16	256	9	36.89x		
MGRIT, FCF-relaxation, $m = 2$	16	1	16	233	1.42x	315	1.66x
MGRIT, FCF-relaxation, $m = 32$	4	1	4	225	1.48x	339	1.54x
MGRIT, FCF-relaxation, $m = 32$	16	1	16	81	4.10x	101	5.18x
MGRIT, FCF-relaxation, $m = 32$	32	1	32	48	6.92x		
MGRIT, FCF-relaxation, $m = 32$	16	16	256	10	33.20x		

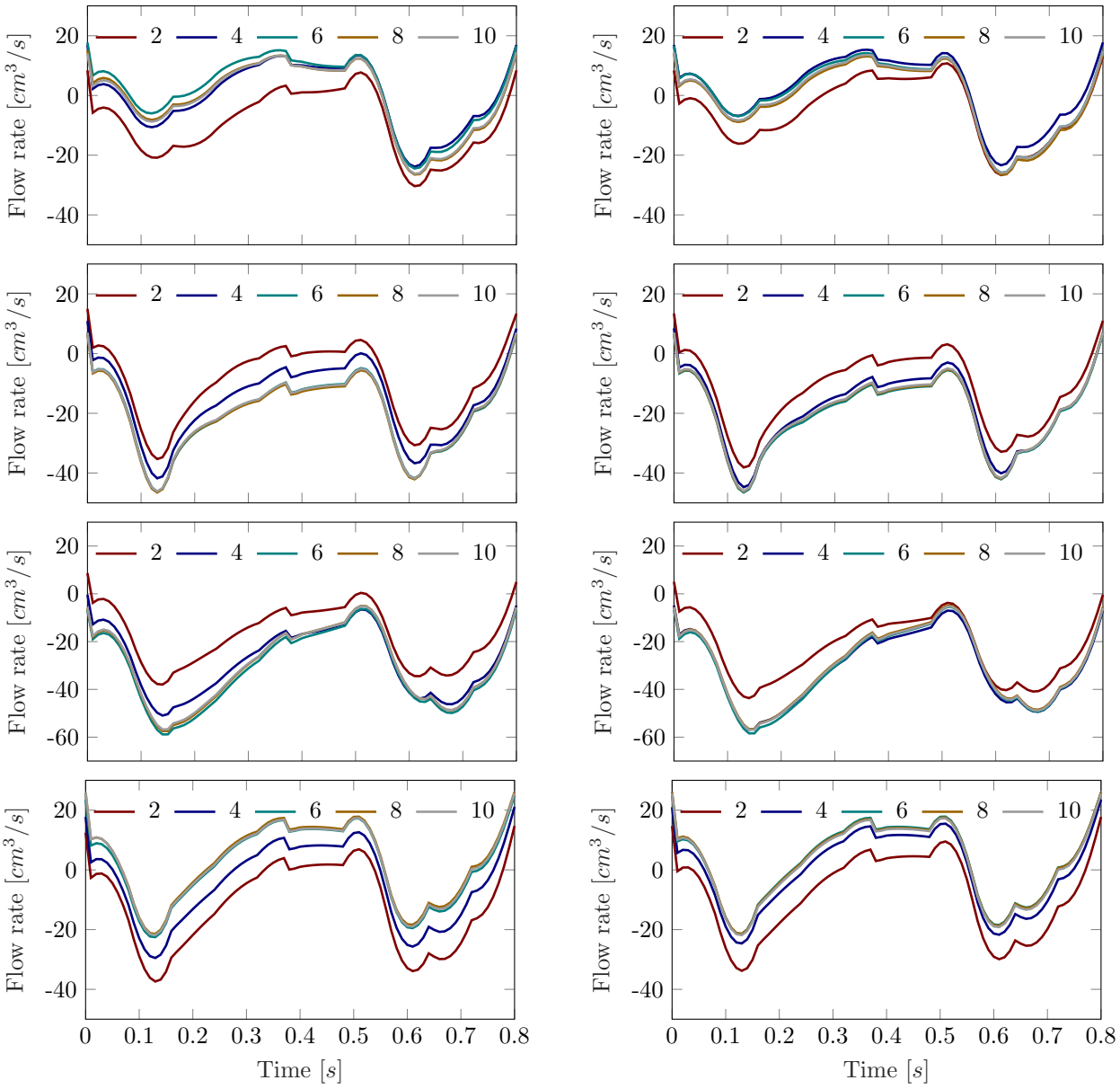
Table 5: Runtimes and respective speedups for sequential time-stepping (10 cycles) and time-periodic MGRIT (q iterations, such that IC tol 10^{-1} is satisfied) with p_x processors in space and p_t processors in time.

Sequential time-stepping		MGRIT		Speedup			
#cycles	Error	#iterations	Error	1 proc	4 procs	16 procs	32 procs
2	$1.78 \cdot 10^{-3}$	3	$1.82 \cdot 10^{-3}$	0.22x	1.10x	2.96x	3.08x
4	$1.11 \cdot 10^{-3}$	4	$1.08 \cdot 10^{-3}$	0.34x	1.72x	4.90x	5.08x
7	$8.15 \cdot 10^{-4}$	6	$8.18 \cdot 10^{-4}$	0.41x	2.03x	6.20x	6.64x
		9	$7.24 \cdot 10^{-4}$	0.28x	1.40x	4.49x	4.91x

Table 6: Coarse mesh: Speedup depending on required accuracy and number of processors for two-level MGRIT with FCF-relaxation and temporal coarsening factor $m = 8$. For a graphical presentation, see Figure 14a.

Sequential time-stepping		MGRIT		Speedup		
#cycles	Error	#iterations	Error	16 procs	32 procs	64 procs
3	$1.04 \cdot 10^{-3}$	2	$9.98 \cdot 10^{-4}$	6.60x	6.52x	8.22x
5	$4.65 \cdot 10^{-4}$	4	$4.17 \cdot 10^{-4}$	7.19x	7.73x	9.60x
6	$3.93 \cdot 10^{-4}$	6	$3.66 \cdot 10^{-4}$	6.40x	7.20x	9.05x
		9	$2.00 \cdot 10^{-4}$	4.60x	5.41x	6.88x

Table 7: Fine mesh: Speedup depending on required accuracy and number of processors for two-level MGRIT with FCF-relaxation and temporal coarsening factor $m = 8$.



(a) A time-periodic steady-state is achieved by running multiple cycles using sequential time stepping.

(b) A time-periodic steady-state is achieved by running multiple iterations using time-periodic MGRIT.

Figure SM1: Flow rates at pulmonary veins $1-4$ (top to bottom) for sequential time-stepping (left) and time-periodic MGRIT (right). Note, a negative flow rate corresponds to inflow.