The Parallel Full Approximation Scheme in Space and Time for a Parabolic Finite Element Problem

Oliver Sander, Ruth Schöbel, and Robert Speck

1 Introduction

The parallel full approximation scheme in space and time (PFASST, [4]) can integrate multiple time-steps simultaneously by using inner iterations of spectral deferred corrections (SDC, [3]) on a space-time hierarchy. It mimics a full approximation scheme (FAS, [12]) for a sequence of coupled collocation problems. For the simulation of space–time dependent problems, PFASST has been used in combination with finite differences, e.g., in [8, 10], but also in connection with particle simulations [11] and spectral methods [5]. In this work we combine PFASST with a finite element discretization in space. Using a simple, nonlinear reaction–diffusion equation, we will derive the discretized, "composite collocation problem" PFASST aims to solve in parallel and show the correct handling of the mass matrix. There exist two different ways to write down the composite collocation problem with a non-trivial mass matrix and we will demonstrate that we can avoid inversion of the mass matrix with the added benefit of a better order of accuracy in time per PFASST iteration. The choice of restriction and prolongation in space plays a major role and we will show the correct formulation and placement of those. Both mass matrix handling and choice of transfer operators mark key differences to using standard finite differences in space, both in terms of theoretical formulation and simulation results. Using a concrete example, we numerically test the order of accuracy per iteration of PFASST and compare it with SDC.

Oliver Sander
Faculty of Mathematics
TU Dresden, Germany e-mail: oliver.sander@tu-dresden.de

Ruth Schöbel
Jülich Supercomputing Centre
Forschungszentrum Jülich GmbH, Germany, e-mail: r.schoebel@fz-juelich.de

Robert Speck
Jülich Supercomputing Centre
Forschungszentrum Jülich GmbH, Germany, e-mail: r.speck@fz-juelich.de

2 PFASST and finite elements in space

We consider the reaction–diffusion equation

$$\begin{aligned} v_t(x,t) &= \Delta v(x,t) + \mathrm{g}(v(x,t)), && x \in \Omega, t \in [t_0, T], \\ v(x,t) &= 0, && x \in \partial\Omega, \end{aligned} \tag{1}$$

with suitable initial conditions for $t = t_0$ and $g : \mathbb{R} \to \mathbb{R}$ continuously differentiable. Here $\Omega \subset \mathbb{R}$ is a polyhedral domain with boundary $\partial\Omega$, and Δ denotes the Laplace operator.

2.1 Finite element discretization in space

We define test functions φ^h in a finite-dimensional space $\mathbf{V}^h \subset \mathbf{H}_0^1(\Omega)$, multiply (1) by these test functions, and integrate by parts. Thus, $v^h(\cdot, t) \in \mathbf{V}^h$ is given by

$$\int_\Omega \varphi^h v_t^h \, dx = -\int_\Omega \nabla\varphi^h \nabla v^h \, dx + \int_\Omega \varphi^h \mathrm{g}(v^h) \, dx \quad \forall \varphi^h \in \mathbf{V}^h. \tag{2}$$

We choose a basis $\varphi_1, \ldots, \varphi_N$ of $\mathbf{V}^h$ and approximate $\mathrm{g}(v^h)$ by an element of $\mathbf{V}^h$ and express v^h and $\mathrm{g}(v^h)$ as

$$v^h(x,t) = \sum_{i=1}^N v_i(t)\varphi_i(x), \qquad \mathrm{g}(v^h)(x,t) \approx \sum_{i=1}^N \mathrm{g}(v_i(t))\varphi_i(x), \tag{3}$$

where the coefficients $v_i(t)$, $i = 1, \ldots, N$, are time-dependent functions. Inserting (3) into equation (2) yields

$$\mathbf{M}u_t = -\mathbf{A}u + \mathbf{M}g(u) =: f(u). \tag{4}$$

Here, $u := (v_1, \ldots, v_N)$ is a vector holding the coefficients v_i, and $g : \mathbb{R}^N \to \mathbb{R}^N$, $g := (\mathrm{g}(v_1), \ldots, \mathrm{g}(v_N))^T$. The matrix $\mathbf{M} \in \mathbb{R}^{N\times N}$ is the mass matrix and $\mathbf{A} \in \mathbb{R}^{N\times N}$ the stiffness matrix

$$\mathbf{M}_{ij} := \int_{\Omega_i} \varphi_i \varphi_j \, dx, \qquad \mathbf{A}_{ij} := \int_{\Omega_i} \nabla\varphi_i \nabla\varphi_j \, dx.$$

2.2 The collocation problem and SDC

For the temporal discretization, we decompose the interval $[t_0, T]$ into time-steps $t_0 < t_1 < \cdots < t_L = T$, $L \in \mathbb{N}$. For one time-step $[t_l, t_{l+1}]$, the Picard formulation of (4) is

$$\mathbf{M}u(t) = \mathbf{M}u_{l,0} + \int_{t_l}^{t} f(u(s))\, ds, \qquad t \in [t_l, t_{l+1}], \tag{5}$$

where $u_{l,0} \coloneqq u(t_l)$. To approximate the integral we use a spectral quadrature rule on $[t_l, t_{l+1}]$ with M quadrature nodes $\tau_{l,1}, ..., \tau_{l,M}$ such that $t_l < \tau_{l,1} < ... < \tau_{l,M} = t_{l+1}$.

For each of the M nodes we introduce a set of M quadrature weights $q_{m,j} \coloneqq \int_{t_l}^{\tau_{l,m}} L_j(s)\, ds$, $m, j = 1, \ldots, M$, where $L_1, \ldots, L_M$ are the Lagrange polynomials for the nodes $\tau_{l,1}, ..., \tau_{l,M}$. We can then approximate the integral in (5) from t_l to $\tau_{l,m}$ by

$$\Delta t \sum_{j=1}^{M} q_{m,j} f(u_{l,j}) \approx \int_{t_l}^{\tau_{l,m}} f(u(s))\, ds, \quad m = 1, \ldots, M,$$

where $\Delta t \coloneqq t_{l+1} - t_l$ denotes the time-step size. Using this in Equation (5) the unknown values $u(\tau_{l,1}), \ldots, u(\tau_{l,M})$ can be approximated by a solution $u_{l,1}, \ldots, u_{l,M} \in \mathbb{R}^N$ of the nonlinear system of equations

$$\mathbf{M}u_{l,m} = \mathbf{M}u_{l,0} + \Delta t \sum_{j=1}^{M} q_{m,j} f(u_{l,j}) \quad \text{for } m = 1, \ldots, M.$$

This is the so called "collocation problem", which we can rewrite as

$$\mathbf{C}_{\boldsymbol{f}}^{\text{coll}}(\boldsymbol{u}_l) \coloneqq (\mathbf{I}_M \otimes \mathbf{M} - \Delta t(\mathbf{Q} \otimes \mathbf{I}_N)\boldsymbol{f})\,(\boldsymbol{u}_l) = (\mathbf{I}_M \otimes \mathbf{M})\boldsymbol{u}_{l,0}, \tag{6}$$

where $\mathbf{I}_X \in \mathbb{R}^{X \times X}$, $X \in \mathbb{N}$ is the identity matrix, $\otimes$ denotes the Kronecker product, $\boldsymbol{u}_l \coloneqq (u_{l,1}, ..., u_{l,M})^T \in \mathbb{R}^{MN}$, $\boldsymbol{u}_{l,0} \coloneqq (u_{l,0}, ..., u_{l,0})^T \in \mathbb{R}^{MN}$, $\mathbf{Q} \coloneqq (q_{ij}) \in \mathbb{R}^{M \times M}$, and the vector function $\boldsymbol{f} : \mathbb{R}^{MN} \to \mathbb{R}^{MN}$ is given by $\boldsymbol{f}(\boldsymbol{u}_l) \coloneqq (f(u_{l,1}), ..., f(u_{l,M}))^T$.

With this matrix notation, spectral deferred corrections can simply be seen as a preconditioned Picard iteration [6, 9]. More precisely, for a lower triangular matrix $\mathbf{Q}_\Delta \in \mathbb{R}^{M \times M}$ we define the preconditioner

$$\mathbf{P}_{\boldsymbol{f}}^{\text{sdc}}(\boldsymbol{u}_l) \coloneqq (\mathbf{I}_M \otimes \mathbf{M} - \Delta t(\mathbf{Q}_\Delta \otimes \mathbf{I}_N)\boldsymbol{f})\,(\boldsymbol{u}_l).$$

Then the preconditioned iteration reads

$$\mathbf{P}_{\boldsymbol{f}}^{\text{sdc}}(\boldsymbol{u}_l^{k+1}) = (\mathbf{P}_{\boldsymbol{f}}^{\text{sdc}} - \mathbf{C}_{\boldsymbol{f}}^{\text{coll}})(\boldsymbol{u}_l^k) + (\mathbf{I}_M \otimes \mathbf{M})\boldsymbol{u}_{l,0}, \quad k = 1, ..., K. \tag{7}$$

The properties of $\mathbf{P}_{\boldsymbol{f}}^{\text{sdc}}$ depend first and foremost on the choice of the matrix $\mathbf{Q}_\Delta$. For this work, we use the backward Euler approach. We refer to [6, 13, 9] for more details on the notation and its relationship to the original description of SDC as in [3]. The key difference is the appearance of the mass matrix $\mathbf{M}$, which for finite differences is just the identity matrix.

2.3 The composite collocation problem and PFASST

For L time-steps, the composite collocation problem is

$$\begin{pmatrix} \mathbf{C}_f^{\text{coll}} & & & \\ -\mathbf{H} & \mathbf{C}_f^{\text{coll}} & & \\ & \ddots & \ddots & \\ & & -\mathbf{H} & \mathbf{C}_f^{\text{coll}} \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \boldsymbol{u}_2 \\ \vdots \\ \boldsymbol{u}_L \end{pmatrix} = \begin{pmatrix} (\mathbf{I}_M \otimes \mathbf{M})\boldsymbol{u}_{0,0} \\ \mathbf{0} \\ \vdots \\ \mathbf{0} \end{pmatrix}, \tag{8}$$

where in the simplest case $\mathbf{N} \in \mathbb{R}^{M\times M}$ just holds ones in the last column and zeros elsewhere. Then, $\mathbf{H} \coloneqq \mathbf{N} \otimes \mathbf{M}$ provides the value at the last quadrature node $\tau_{l,M}$ of a time-step $[t_l, t_{l+1}]$ as initial value for the following time-step. Defining the global state vector $\boldsymbol{u} \coloneqq (\boldsymbol{u}_1, ..., \boldsymbol{u}_L)^T \in \mathbb{R}^{LMN}$, the vector $\boldsymbol{b} \coloneqq ((\mathbf{I}_M \otimes \mathbf{M})\boldsymbol{u}_{0,0}, \mathbf{0}, ..., \mathbf{0})^T \in \mathbb{R}^{LMN}$, and $\boldsymbol{F} : \mathbb{R}^{LMN} \to \mathbb{R}^{LMN}$ with $\boldsymbol{F}(\boldsymbol{u}) \coloneqq (\boldsymbol{f}(\boldsymbol{u}_1), \ldots, \boldsymbol{f}(\boldsymbol{u}_L))^T$, we can write this in the more compact form as $\mathbf{C}_{\boldsymbol{F}}(\boldsymbol{u}) = \boldsymbol{b}$, where $\mathbf{C}_{\boldsymbol{F}}$ is the lower block-bidiagonal, nonlinear operator on the left of (8). Using the definition (6) of $\mathbf{C}_f^{\text{coll}}$ we write (8) as

$$(\mathbf{I}_{LM} \otimes \mathbf{M} - \Delta t(\mathbf{I}_L \otimes \mathbf{Q} \otimes \mathbf{I}_N)\boldsymbol{F} - \mathbf{E} \otimes \mathbf{H})(\boldsymbol{u}) = \boldsymbol{b} \tag{9}$$

where the matrix $\mathbf{E} \in \mathbb{R}^{L\times L}$ has ones on the lower off-diagonal and zeros elsewhere, accounting for the transfer of the solution from one step to the next.

There are two fundamentally different ways to solve this system iteratively with SDC. We can choose either (note the $\mathbf{Q}_\Delta$ instead of the $\mathbf{Q}$)

$$\mathbf{P}_{\boldsymbol{F}}^{\text{par}}(\boldsymbol{u}) \coloneqq (\mathbf{I}_{LM} \otimes \mathbf{M} - \Delta t(\mathbf{I}_L \otimes \mathbf{Q}_\Delta \otimes \mathbf{I}_N)\boldsymbol{F})(\boldsymbol{u})$$

or

$$\mathbf{P}_{\boldsymbol{F}}^{\text{seq}}(\boldsymbol{u}) \coloneqq (\mathbf{I}_{LM} \otimes \mathbf{M} - \Delta t(\mathbf{I}_L \otimes \mathbf{Q}_\Delta \otimes \mathbf{I}_N)\boldsymbol{F} - \mathbf{E} \otimes \mathbf{H})(\boldsymbol{u}),$$

where the latter still has the $\mathbf{H}$ matrices in the lower off-diagonal. $\mathbf{P}_{\boldsymbol{F}}^{\text{par}}$ is a parallel preconditioner, which performs SDC iterations on each step simultaneously, while $\mathbf{P}_{\boldsymbol{F}}^{\text{seq}}$ propagates a single SDC iteration sequentially forward in time.

The idea of PFASST now is to couple both preconditioners in a two-level space-time full approximations scheme: the parallel $\mathbf{P}_{\boldsymbol{F}}^{\text{par}}$ is used on the original problem in space and time (the "fine" level), while the sequential $\mathbf{P}_{\boldsymbol{F}}^{\text{seq}}$ with better convergence properties is used on a coarser, cheaper level with reduced accuracy in space and/or time to reduce the impact of its sequential nature. To create the coarse level, we reduce the number of degrees of freedom in space and choose a finite element subspace $\tilde{\mathbf{V}}^h \subset \mathbf{V}^h$. Three different transfer operations are then needed for PFASST:

1. Restriction of a coefficient vector $\boldsymbol{u}_{l,m}$, representing an object in $\mathbf{V}^h$, to the representation of an object in $\tilde{\mathbf{V}}^h$,
2. Restriction of the residual $\mathbf{C}_{\boldsymbol{F}}(\boldsymbol{u}) - \boldsymbol{b}$,

3. Prolongation of a coefficient vector $\tilde{\boldsymbol{u}}_{l,m}$, representing an object in $\tilde{\mathbf{V}}_h$, to the representation of an object in $\mathbf{V}_h$.

Using Lagrange polynomials, operations 2 and 3 can be done using the canonical injection $\mathbf{T}^N \in \mathbb{R}^{N\times\tilde{N}}$ for prolongation of the coefficient vector and its transpose $(\mathbf{T}^N)^T \in \mathbb{R}^{\tilde{N}\times N}$ for restriction of the residual. For Operation 1, we use the matrix $\mathbf{R}^N \in \mathbb{R}^{\tilde{N}\times N}$ that represents the Lagrange interpolation of functions from $\mathbf{V}^h$ in $\tilde{\mathbf{V}}^h$. By $\mathbf{T} := \mathbf{I}_{LM} \otimes \mathbf{T}^N$ and $\mathbf{R} := \mathbf{I}_{LM} \otimes \mathbf{R}^N$ we define global transfer operators. Using the tilde symbols to indicate entities on the coarse level, one iteration of PFASST reads:

1. Restrict current iterate to the coarse level: $\tilde{\boldsymbol{u}}^k = \mathbf{R}\boldsymbol{u}^k$.
2. Compute FAS correction: $\boldsymbol{\tau} = \tilde{\mathbf{C}}_{\boldsymbol{F}}(\tilde{\boldsymbol{u}}^k) - \mathbf{T}^T\mathbf{C}_{\boldsymbol{F}}(\boldsymbol{u}^k)$
3. Compute $\tilde{\boldsymbol{u}}^{k+1}$ by solving: $\tilde{\mathbf{P}}^{\text{seq}}_{\boldsymbol{F}}(\tilde{\boldsymbol{u}}^{k+1}) = (\tilde{\mathbf{P}}^{\text{seq}}_{\boldsymbol{F}} - \tilde{\mathbf{C}}_{\boldsymbol{F}})(\tilde{\boldsymbol{u}}^k) + \tilde{\boldsymbol{b}} + \boldsymbol{\tau}$.
4. Apply coarse grid correction: $\boldsymbol{u}^{k+\frac{1}{2}} = \boldsymbol{u}^k + \mathbf{T}(\tilde{\boldsymbol{u}}^{k+1} - \mathbf{R}\boldsymbol{u}^k)$.
5. Compute $\boldsymbol{u}^{k+1}$ by solving: $\mathbf{P}^{\text{par}}_{\boldsymbol{F}}(\boldsymbol{u}^{k+1}) = (\mathbf{P}^{\text{par}}_{\boldsymbol{F}} - \mathbf{C}_{\boldsymbol{F}})(\boldsymbol{u}^{k+\frac{1}{2}}) + \boldsymbol{b}$.

In contrast to the description in [1, 2], the mass matrices are now included in $\mathbf{P}^{\text{par}}_{\boldsymbol{F}}(\boldsymbol{u})$, $\tilde{\mathbf{P}}^{\text{seq}}_{\boldsymbol{F}}(\tilde{\boldsymbol{u}})$ as well as in $\mathbf{C}_{\boldsymbol{F}}$ and $\tilde{\mathbf{C}}_{\boldsymbol{F}}$. This approach is preferable to others, including the naive one where the collocation problem (6) is multiplied by $\mathbf{M}^{-1}$ from the left. The collocation problem (6) then reads

$$\bar{\mathbf{C}}^{\text{coll}}_{\bar{f}}(\boldsymbol{u}_l) := (\mathbf{I}_M \otimes \mathbf{I}_N - \Delta t(\mathbf{Q} \otimes \mathbf{I}_N)\bar{\boldsymbol{f}})(\boldsymbol{u}_l) = \boldsymbol{u}_{l,0}, \tag{11}$$

for $\bar{\boldsymbol{f}}(\boldsymbol{u}_l) := (\bar{f}(u_{l,1}), ..., \bar{f}(u_{l,M}))^T$ and $\bar{f}(u_{l,m}) = \mathbf{M}^{-1} f(u_{l,m})$. Similarly, the composite collocation problem then is

$$\bar{\mathbf{C}}_{\bar{\boldsymbol{F}}}(\boldsymbol{u}) := (\mathbf{I}_{LMN} - \Delta t(\mathbf{I}_L \otimes \mathbf{Q} \otimes \mathbf{I}_N)\bar{\boldsymbol{F}} - \mathbf{E} \otimes \mathbf{N} \otimes \mathbf{I}_N)(\boldsymbol{u}) = \bar{\boldsymbol{b}}, \tag{12}$$

where $\bar{\boldsymbol{b}} := (\boldsymbol{u}_{0,0}, \mathbf{0}, ..., \mathbf{0})^T$ and $\bar{\boldsymbol{F}} := \left(\bar{\boldsymbol{f}}(\boldsymbol{u}_1), \ldots, \bar{\boldsymbol{f}}(\boldsymbol{u}_L)\right)^T$. SDC and PFASST can then be derived precisely as in the literature, using the modified right-hand side $\bar{f}$. Note that the inversion of the mass matrix is only necessary, if the actual residual of (11) or (12) needs to be computed, which, e.g., is necessary for the FAS correction. There, $\bar{\boldsymbol{F}}$ has to be evaluated on the fine and the coarse level, both containing the inverse of the respective mass matrix. While seemingly attractive in terms of writing a generic code, inversion of the mass matrix can be costly and, as we will see later, convergence of PFASST is way worse in this case. Note that the components of the residual of (11) and (12) in contrast to (6) and (9) are not elements of the dual space of $\mathbf{V}^h$ and therefore cannot be restricted exactly. In this case, the obvious choice is to use $\mathbf{R}^N$ to transfer both residual and coefficient vectors to the coarse level.

3 Numerical results

We now investigate numerically the convergence behavior of PFASST with finite elements in space. In [7] it was shown that for a discretization with finite differences, the single-step version of PFASST (i.e. multilevel SDC) can gain two orders of accuracy per iteration, provided very high-order transfer operators in space are used [1, 7]. We will now show numerically that with finite elements in space, this is no longer necessary.

We use the following nonlinear differential equation

$$u_t = \Delta u + u^2(1-u) \quad \text{on } [0,2] \times [-20,20]. \tag{13}$$

In all simulations, we use 4 Gauss–Raudau nodes to discretize a single time-step. In the following we use SDC for serial time-step calculations and PFASST to calculate 4 time-steps simultaneously. The spatial domain $[-20,20]$ is discretized using Lagrange finite elements of either order 1 or order 3. We use the initial value $u(x,0) = (1+(\sqrt{2}-1)e^{-\sqrt{6}/6x})^{-2}$ as the initial guess for the iteration.

For the first test case we use a third-order Lagrange basis to approximate the solution. We coarsen the problem in space by restricting to a second-order Lagrange basis. Figure 1 shows the results for SDC and PFASST. They show the absolute error of the method in the infinity norm for different time-step sizes, in relation to a reference solution calculated with a much smaller Δt and SDC. While SDC gains one order per iteration as expected, PFASST can gain up to two orders per iteration, at least after some initial iterations have been performed. This "burn-in" phase causes a loss of parallel efficiency when actual speedup is measured. After this phase, however, PFASST shows ideal convergence behavior gaining two orders of accuracy per iteration. There is not yet a theoretical explanation for neither the "burn-in" nor the "ideal" phase.

For a second test case we use a first-order Lagrange basis and coarsen the problem in space by doubling the element size. Figure 2 shows the results for SDC and PFASST. In the same way as in the high-order example before, SDC gains one order or accuracy per iteration, while PFASST can gain up to two orders after a few initial iterations. Note that in the case of a finite difference discretization, the order of the interpolation is crucial to obtain two orders per iteration [1, 7]. The usage of $\mathbf{T}^N$ as exact interpolation for nested finite element spaces removes this, so far, persistent and irritating limitation.

Finally, Figure 3 shows SDC and PFASST applied to the (composite) collocation problem (12) with inverted mass matrix and a first-order Lagrange basis. SDC behaves exactly as before, while PFASST fails to show any reasonable convergence. In particular, increasing the number of iterations does not increase the order of accuracy beyond 1.

The advantage of using finite elements together with PFASST in the way we demonstrated here is not yet analyzed analytically. We intend to address this in a follow-up work. Also, the important fact that two orders of accuracy per iteration

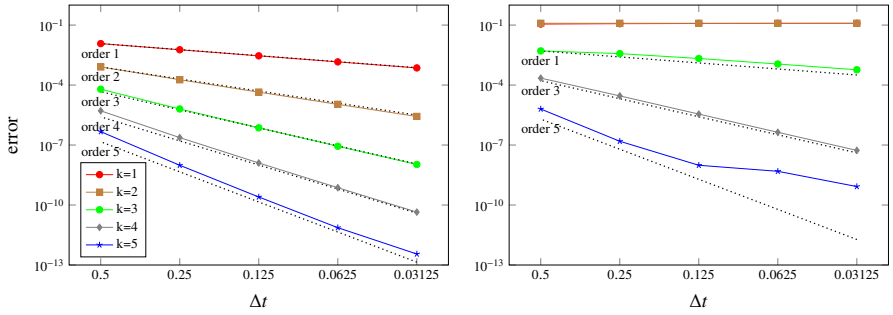

Fig. 1: SDC (left) and PFASST (right) errors for different Δt and number of iterations k, 128 spatial elements, order 3. The dashed lines indicate the expected order of accuracy in time.

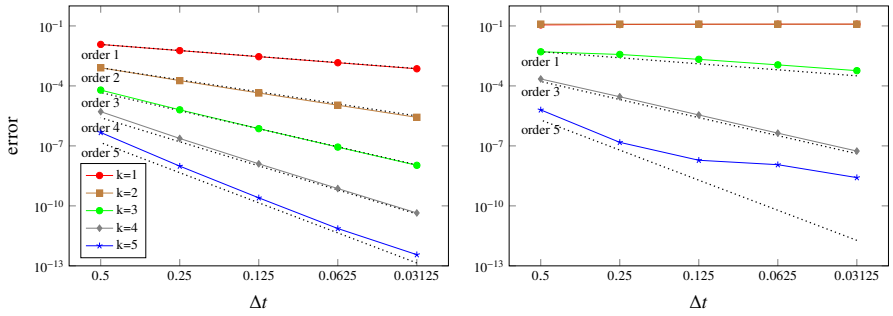

Fig. 2: SDC (left) and PFASST (right) errors for different Δt and number of iterations k, 512 spatial elements, order 1. The dashed lines indicate the expected order of accuracy in time.

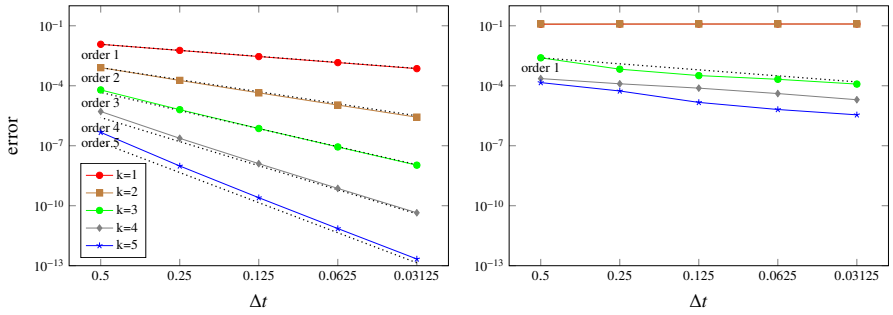

Fig. 3: Naive approach with inverted mass matrix: SDC (left) and PFASST (right) errors for different Δt and number of iterations k, 512 spatial elements, order 1

is possible even with a low-order spatial interpolation does not have a theoretical explanation. A corresponding analysis is work in progress.

References

1. M. Bolten, D. Moser, and R. Speck. A multigrid perspective on the parallel full approximation scheme in space and time. *Numerical Linear Algebra with Applications*, 24(6):e2110, 2017.
2. M. Bolten, D. Moser, and R. Speck. Asymptotic convergence of the parallel full approximation scheme in space and time for linear problems. *Numerical Linear Algebra with Applications*, 25(6):e2208, 2018.
3. A. Dutt, L. Greengard, and V. Rokhlin. Spectral deferred correction methods for ordinary differential equations. *BIT Numerical Mathematics*, 40(2):241–266, 2000.
4. M. Emmett and M. L. Minion. Toward an efficient parallel in time method for partial differential equations. *Communications in Applied Mathematics and Computational Science*, 7:105–132, 2012.
5. S. Götschel and M. L. Minion. Parallel-in-time for parabolic optimal control problems using PFASST. In *International Conference on Domain Decomposition Methods*, pages 363–371. Springer, 2017.
6. J. Huang, J. Jia, and M. Minion. Accelerating the convergence of spectral deferred correction methods. *Journal of Computational Physics*, 214(2):633 – 656, 2006.
7. G. Kremling and R. Speck. Convergence analysis of multi-level spectral deferred corrections. *Communications in Applied Mathematics and Computational Science*, 16.2:227–265, 2021.
8. M. L. Minion, R. Speck, M. Bolten, M. Emmett, and D. Ruprecht. Interweaving PFASST and parallel multigrid. *SIAM journal on scientific computing*, 37(5):S244–S263, 2015.
9. D. Ruprecht and R. Speck. Spectral deferred corrections with fast-wave slow-wave splitting. *SIAM Journal on Scientific Computing*, 38(4):A2535–A2557, 2016.
10. R. Schöbel and R. Speck. PFASST-ER: Combining the parallel full approximation scheme in space and time with parallelization across the method. *Computing and Visualization in Science*, 23(1):1–12, 2020.
11. R. Speck, D. Ruprecht, R. Krause, M. Emmett, M. Minion, M. Winkel, and P. Gibbon. A massively space-time parallel N-body solver. In *SC'12: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–11. IEEE, 2012.
12. U. Trottenberg, C. Oosterlee, and A. Schuller. *Multigrid*. Academic Press, 2000.
13. M. Weiser. Faster SDC convergence on non-equidistant grids by DIRK sweeps. *BIT Numerical Mathematics*, 55(4):1219–1241, 2014.