
Article:
Speck, R, Ruprecht, D, Emmett, M et al. (2015) A multi-level spectral deferred correction method. BIT Numerical Mathematics, 55 (3). 843 - 867. ISSN: 0006-3835

https://doi.org/10.1007/s10543-014-0517-x

BIT manuscript No.
(will be inserted by the editor)

A multi-level spectral deferred correction method

Robert Speck · Daniel Ruprecht · Matthew Emmett · Michael Minion · Matthias Bolten · Rolf Krause

Received: date / Accepted: date

Abstract The spectral deferred correction (SDC) method is an iterative scheme for computing a higher-order collocation solution to an ODE by performing a series of correction sweeps using a low-order timestepping method. This paper examines a variation of SDC for the temporal integration of PDEs called multi-level spectral deferred corrections (MLSDC), where sweeps are performed on a hierarchy of levels and an FAS correction term, as in nonlinear multigrid methods, couples solutions on different levels. Three different strategies to reduce the computational cost of correction sweeps on the coarser levels are examined: re-

Robert Speck and Daniel Ruprecht acknowledge supported by Swiss National Science Foundation grant 145271 under the lead agency agreement through the project "ExaSolvers" within the Priority Programme 1648 "Software for Exascale Computing" of the Deutsche Forschungsgemeinschaft. Matthias Bolten acknowledges support from DFG through the project "ExaStencils" within SPPEXA. Daniel Ruprecht and Matthew Emmett also thankfully acknowledge support by grant SNF-147597. Matthew Emmett and Michael Minion were supported by the Applied Mathematics Program of the DOE Office of Advanced Scientific Computing Research under the U.S. Department of Energy under contract DE-AC02-05CH11231. Michael Minion was also supported by the U.S. National Science Foundation grant DMS-1217080.

R. Speck
Jülich Supercomputing Centre, Forschungszentrum Jülich, Germany and Institute of Computational Science, Università della Svizzera italiana, Lugano, Switzerland.
E-mail: r.speck@fz-juelich.de

D. Ruprecht
Institute of Computational Science, Università della Svizzera italiana, Lugano, Switzerland.
E-mail: daniel.ruprecht@usi.ch

M. Emmett
Center for Computational Sciences and Engineering, Lawrence Berkeley National Laboratory, USA.
E-mail: mwemmett@lbl.gov

M. Minion
Institute for Computational and Mathematical Engineering, Stanford University, USA.
E-mail: mlminion@stanford.edu

M. Bolten
Department of Mathematics, Bergische Universität Wuppertal, Germany.
E-mail: bolten@math.uni-wuppertal.de

R. Krause
Institute of Computational Science, Università della Svizzera italiana, Lugano, Switzerland.
E-mail: rolf.krause@usi.ch

ducing the degrees of freedom, reducing the order of the spatial discretization, and reducing the accuracy when solving linear systems arising in implicit temporal integration. Several numerical examples demonstrate the effect of multi-level coarsening on the convergence and cost of SDC integration. In particular, MLSDC can provide significant savings in compute time compared to SDC for a three-dimensional problem.

Keywords spectral deferred corrections · multi-level spectral deferred corrections · FAS correction · PFASST

Mathematics Subject Classification (2000) 65M55 · 65M70 · 65Y05

1 Introduction

The numerical approximation of initial value ordinary differential equations is a fundamental problem in computational science, and many integration methods for problems of different character have been developed [2,20,21]. Among different solution strategies, this paper focuses on a class of iterative methods called Spectral Deferred Corrections (SDC) [16], which is a variant of the defect and deferred correction methods developed in the 1960s [3, 15,35,36,42,46]. In SDC methods, high-order temporal approximations are computed over a timestep by discretizing and approximating a series of correction equations on intermediate substeps. These corrections are applied iteratively to a provisional solution computed on the substeps, with each iteration – or *sweep* – improving the solution and raising the formal order of accuracy of the method, see e.g. [11,13,45]. The correction equations are cast in the form of a Picard integral equation containing an explicitly calculated term corresponding to the temporal integration of the function values from the previous iteration. Substeps in SDC methods are chosen to correspond to Gaussian quadrature nodes, and hence the integrals can be stably computed to a very high order of accuracy.

One attractive feature of SDC methods is that the numerical method used to approximate the correction equations can be low-order (even first-order) accurate, while the solution after many iterations can in principal be of arbitrarily high-order of accuracy. This has been exploited to create SDC methods that allow the governing equations to be split into two or more pieces that can be treated either implicitly or explicitly and/or with different timesteps, see e.g. [5,6,29,32].

For high-order SDC methods constructed from low-order propagators, the provisional solution and the solution after the first few correction iterations are of lower-order compared to the final solution. Hence it is possible to reduce the computational work done on these early iterations by reducing the number of substeps (i.e. quadrature nodes) since higher-order integrals are not yet necessary. In [30,32], the number of substeps used in initial iterations of SDC methods is appropriately reduced to match the accuracy of the solution, and the methods there are referred to as *ladder methods*. Ladder methods progress from a low-order coarse solution to a high-order fine solution by performing one or more SDC sweeps on the coarse level and then using an interpolated (in time and possibly space) version of the solution as the provisional solution for the next correction sweep. In both [30,32] the authors conclude that the reduction in work obtained by using ladder methods is essentially offset by a corresponding decrease in accuracy, making ladder methods no more computationally efficient than non-ladder SDC methods. On the other hand, in [28], SDC methods for a method of lines discretizations of PDEs are explored wherein the ladder strategy allows both spatial and temporal coarsening as well as the use of lower-order spatial discretizations in initial iterations. The numerical results in [28] indicate that adding spatial coarsening

to SDC methods for PDEs can increase the overall efficiency of the timestepping scheme, although this evidence is based only on numerical experiments using simple test cases.

This paper significantly extends the idea of using spatial coarsening in SDC when solving PDEs. A general multi-level strategy is analyzed wherein correction sweeps are applied to different levels as in the V-cycles of multigrid methods (e.g. [7,8]). A similar strategy is used in the parallel full approximation scheme in space and time (PFASST), see [18,34] and also [39], to enable concurrency in time by iterating on multiple timesteps simultaneously. As in nonlinear multigrid methods, multi-level SDC applies an FAS-type correction to enhance the accuracy of the solution on coarse levels. Therefore, some of the fine sweeps required by a single-level SDC algorithm can be replaced by coarse sweeps, which are relatively cheaper when spatial coarsening strategies are used. The paper introduces MLSDC and discusses three such spatial coarsening strategies: (1) reducing the number of degrees of freedom, (2) reducing the order of the discretization and (3) reducing the accuracy of implicit solves. To enable the use of a high-order compact stencils for spatial operators, several modifications to SDC and MLSDC are presented that incorporate a weighting matrix. It is shown for example problems in one and two dimensions that the number of MLSDC iterations required to converge to the collocation solution can be fewer than for SDC, even when the problem is poorly resolved in space. Furthermore, results from a three-dimensional benchmark problem demonstrate that MLSDC can significantly reduce time-to-solution compared to single-level SDC.

2 Multi-level spectral deferred corrections

The details of the MLSDC schemes are presented in this section. The original SDC method is first reviewed in §2.1, while MLSDC along with a brief review of FAS corrections, the incorporation of weighting matrices and a discussion of different coarsening strategies is presented in §2.2.

2.1 Spectral deferred corrections

SDC methods for ODEs were first introduced in [16], and were subsequently refined and extended e.g. in [22,24,32,33]. SDC methods iteratively compute the solution to the collocation equation by approximating a series of correction equations at spectral quadrature nodes using low-order substepping methods. The derivation of SDC starts from the Picard integral form of a generic IVP given by

$$u(t) = u_0 + \int_0^t f\big(u(s),s\big)\mathrm{d}s \tag{1}$$

where $t \in [0,T]$, $u_0, u(t) \in \mathbb{R}^N$, and $f : \mathbb{R}^N \times \mathbb{R} \to \mathbb{R}^N$. We now focus on a single timestep $[T_n, T_{n+1}]$, which is divided into substeps by defining a set of quadrature nodes on the interval. Here we consider Lobatto quadrature and denote $M+1$ nodes $\boldsymbol{t} := (t_m)_{m=0,\ldots,M}$ such that $T_n = t_0 < t_1 < \ldots < t_M = T_{n+1}$. We now denote the collocation polynomial on $[T_n, T_{n+1}]$ by $u_p(t)$ and write $U_j = u_p(t_j) \approx u(t_j)$. In order to derive equations for the intermediate solutions U_j, we define quadrature weights

$$q_{m,j} := \frac{1}{\Delta t}\int_{T_n}^{t_m} l_j(s)\, ds,\ m = 0,\ldots,M,\ j = 0,\ldots,M \tag{2}$$

where $(l_j)_{j=0,\dots,M}$ are the Lagrange polynomials defined by the nodes $\boldsymbol{t}$, and $\Delta t = T_{N+1} - T_N$. Inserting $u_p(t)$ into (1) and noting that the quadrature with weights defined in (2) integrates the polynomial $u_p(t)$ exactly, we obtain

$$U_m = u_0 + \Delta t \sum_{j=0}^{M} q_{m,j} f(U_j, t_j),\ m = 0, \dots, M. \tag{3}$$

For a more compact notation, we now define the *integration matrix* $\boldsymbol{q}$ to be the $M+1 \times M+1$ matrix consisting of entries $q_{m,j}$. Note that because we use Gauss-Lobatto nodes, the first row of $\boldsymbol{q}$ is all zeros. Next, we denote

$$\boldsymbol{U} := [U_0, \dots, U_M]^T,$$

and

$$\boldsymbol{F}(\boldsymbol{U}) := [F_0, \dots, F_M]^T := [f(U_0, t_0), \dots, f(U_M, t_M)]^T.$$

In order to multiply the integration matrix $\boldsymbol{q}$ with the vector of the right-hand side values, we define $\boldsymbol{Q} := \boldsymbol{q} \otimes \boldsymbol{I}_N$ where $\boldsymbol{I}_N \in \mathbb{R}^{N \times N}$ is the identity matrix and $\otimes$ is the Kronecker product. With these definitions, the set of equations in (3) can be written more compactly as

$$\boldsymbol{U} = \boldsymbol{U}_0 + \Delta t \, \boldsymbol{Q} \boldsymbol{F}(\boldsymbol{U})$$

where $\boldsymbol{U}_0 := U_0 \otimes \boldsymbol{I}_N$. Eq. (4) is an implicit equation for the unknowns in $\boldsymbol{U}$, and is also referred to as the collocation formulation. Because we use Gauss-Lobatto nodes, the value U_M readily approximates the solution $u(T_{n+1})$.

Here, we consider ODEs that can be split into stiff (f^I) and non-stiff (f^E) pieces so that

$$f(u(t),t) = f^E\big(u(t),t\big) + f^I\big(u(t),t\big).$$

SDC iterations begin by spreading the initial condition U_0 to each of the collocation nodes so that the provisional solution $\boldsymbol{U}^0$ is given by $\boldsymbol{U}^0 = [U_0, \cdots, U_0]$. We define by

$$s_{m,j} := \frac{1}{\Delta t} \int_{t_{m-1}}^{t_m} l_j(s)\, ds,\ m = 1, \dots, M$$

the quadrature weights for node-to-note integration, approximating integrals over $[t_{m-1}, t_m]$, and as $\boldsymbol{s}$ the $M \times M+1$ matrix consisting of the entries $s_{m,j}$. Note that $\boldsymbol{s}$ can be easily constructed from the integration matrix $\boldsymbol{q}$. Furthermore, we denote as before $\boldsymbol{S} := \boldsymbol{s} \otimes \boldsymbol{I}_N$. Then, the semi-implicit update equation corresponding to the forward/backward Euler substepping method for computing $\boldsymbol{U}^{k+1}$ is given by

$$\begin{aligned} U_{m+1}^{k+1} = U_m^{k+1} &+ \Delta t_m \big[f^E(U_m^{k+1}, t_m) - f^E(U_m^k, t_m)\big] \\ &+ \Delta t_m \big[f^I(U_{m+1}^{k+1}, t_{m+1}) - f^I(U_{m+1}^k, t_{m+1})\big] + \Delta t \, S_m^k \end{aligned} \tag{4}$$

where S_m^k is the m^{th} row of $\boldsymbol{S}\boldsymbol{F}(\boldsymbol{U}^k)$ and $\Delta t_m := t_{m+1} - t_m$. The process of solving (4) at each node is referred to as an *SDC sweep* or an *SDC iteration* (see Algorithm 1). SDC with a fixed number of k iterations and first-order sweeps is formally $O(\Delta t^k)$ up to the accuracy of the underlying integration rule [12,45]. When SDC iterations converge, the scheme becomes equivalent to the collocation scheme determined by the quadrature nodes, and hence is of order $2M$ with $M+1$ Lobatto nodes.

It has been shown [24,30] that in certain situations (particularly stiff equations) the convergence of SDC iterates can slow down considerably for large values of Δt. For a fixed

Algorithm 1: IMEX SDC sweep algorithm.

Data: Initial U_0, function evaluations $\boldsymbol{F}(\boldsymbol{U}^k)$ from the previous iteration, and (optionally) FAS corrections $\boldsymbol{\tau}$.
Result: Solution $\boldsymbol{U}^{k+1}$ and function evaluations $\boldsymbol{F}(\boldsymbol{U}^{k+1})$.

 # Compute integrals
for $m = 0 \ldots M-1$ **do**
  $S_m^k \longleftarrow \Delta t \sum_{j=0}^{M} s_{m,j}(F_j^{E,k} + F_j^{I,k})$
end

 # Set initial condition and compute function evaluation
$t \longleftarrow t_0$; $U_0^{k+1} \longleftarrow U_0$
$F_0^{E,k+1} \longleftarrow f^E(U_0, t)$
$F_0^{I,k+1} \longleftarrow f^I(U_0, t)$

 # Forward/backward Euler substepping for correction
for $m = 0 \ldots M-1$ **do**
  $t \longleftarrow t + \Delta t_m$
  $\text{RHS} \longleftarrow U_m^{k+1} + \Delta t_m\big(F_m^{E,k+1} - F_m^{E,k} - F_{m+1}^{I,k}\big) + S_m^k + \tau_m$
  $U_{m+1}^{k+1} \longleftarrow \text{Solve}\big(U - \Delta t_m f^I(U,t) = \text{RHS}\big)$ for U
  $F_{m+1}^{E,k+1} \longleftarrow f^E(U_{m+1}^{k+1}, t)$
  $F_{m+1}^{I,k+1} \longleftarrow f^I(U_{m+1}^{k+1}, t)$
end

The FAS correction, denoted by $\boldsymbol{\tau}$, is included here to ellucidate how FAS corrections derived in §2.2 are incorporated into an SDC sweep – for plain, single level SDC algorithms the FAS correction $\boldsymbol{\tau}$ would be zero.

number of iterations, this lack of convergence is characterized by order reduction. Hence in this study, to allow for a reasonable comparison of SDC and MLSDC, we perform iterations until a specified convergence criterion is met. Convergence is monitored by computing the SDC residual

$$\boldsymbol{r}^k = \boldsymbol{U}_0 + \Delta t \boldsymbol{Q}\boldsymbol{F}(\boldsymbol{U}^k) - \boldsymbol{U}^k, \tag{5}$$

and the iteration is terminated when the norm of the residual drops below a prescribed tolerance. Similary, if SDC or MLSDC are used to solve the collocation problem up to some fixed tolerance, one also observes a significant increase in the number of iterations required to reach a set tolerance. Accelerating the convergence of SDC for stiff problems has been studied in e.g. [25,44].

2.2 Multi-level spectral deferred corrections

In multi-level SDC (MLSDC), SDC sweeps are performed on a hierarchy of discretizations or *levels* to solve the collocation equation (4). This section presents the details of the MLSDC iterations for a generic set of levels, and in Sect. 2.2.4, three different coarsening strategies are explored. For the following, we define levels $\ell = 1 \ldots L$, where $\ell = 1$ is the discretization that is to be solved (referred to generically as the *fine* level), and subsequent levels $\ell = 2 \ldots L$ are defined by successive coarsening of a type to be specified later.

2.2.1 FAS correction

Solutions on different MLSDC levels are coupled in the same manner as used in the full approximation scheme (FAS) for nonlinear multigrid methods (see e.g. [7]). The FAS cor-

rection for coarse SDC iterations is determined by considering SDC as an iterative method for solving the collocation formulation (4), where the operators A_ℓ are given by $A_\ell(\boldsymbol{U}_\ell) \equiv \boldsymbol{U}_\ell - \Delta t \boldsymbol{Q}_\ell \boldsymbol{F}_\ell(\boldsymbol{U}_\ell)$. Note that the approximations A_ℓ of the operator A can differ substantially between levels as will be discussed in §2.2.4. Furthermore, we assume that suitable restriction (denote by R) and interpolation operators between levels are available, see §2.2.5. The FAS correction for coarse-grid sweeps is then given by

$$\boldsymbol{\tau}_{\ell+1} = A_{\ell+1}(R\boldsymbol{U}_\ell) - RA_\ell(\boldsymbol{U}_\ell) = \Delta t\big(R\boldsymbol{Q}_\ell \boldsymbol{F}_\ell(\boldsymbol{U}_\ell) - \boldsymbol{Q}_{\ell+1}\boldsymbol{F}_{\ell+1}(R\boldsymbol{U}_\ell)\big). \tag{6}$$

In particular, if the fine residual is zero (i.e., $\boldsymbol{U}_\ell \equiv \boldsymbol{U}_{0,\ell} + \Delta t \boldsymbol{Q}_\ell \boldsymbol{F}_\ell(\boldsymbol{U}_\ell)$) the FAS-corrected coarse equation becomes

$$\begin{aligned} \boldsymbol{U}_{\ell+1} - \Delta t \boldsymbol{Q}_{\ell+1}\boldsymbol{F}_{\ell+1}(\boldsymbol{U}_{\ell+1}) &= R\boldsymbol{U}_{0,\ell} + \Delta t\big(R\boldsymbol{Q}_\ell \boldsymbol{F}_\ell(\boldsymbol{U}_\ell) - \boldsymbol{Q}_{\ell+1}\boldsymbol{F}_{\ell+1}(R\boldsymbol{U}_\ell)\big) \\ &= R\boldsymbol{U}_\ell - \Delta t \boldsymbol{Q}_{\ell+1}\boldsymbol{F}_{\ell+1}(R\boldsymbol{U}_\ell) \end{aligned}$$

so that the coarse solution is the restriction of the fine solution. Note that for multi-level schemes, FAS-corrections from finer levels need to be restricted and incorporated to coarser levels as well, i.e. if on level ℓ the equation is already corrected by $\boldsymbol{\tau}_\ell$ with

$$A_\ell(\boldsymbol{U}_\ell) = \boldsymbol{U}_\ell - \Delta t \boldsymbol{Q}_\ell \boldsymbol{F}_\ell(\boldsymbol{U}_\ell) - \boldsymbol{\tau}_\ell,$$

the correction $\boldsymbol{\tau}_{\ell+1}$ for level $\ell+1$ is then given by

$$\boldsymbol{\tau}_{\ell+1} = A_{\ell+1}(R\boldsymbol{U}_\ell) - RA_\ell(\boldsymbol{U}_\ell) = \Delta t\big(R\boldsymbol{Q}_\ell \boldsymbol{F}_\ell(\boldsymbol{U}_\ell) - \boldsymbol{Q}_{\ell+1}\boldsymbol{F}_{\ell+1}(R\boldsymbol{U}_\ell)\big) + R\boldsymbol{\tau}_\ell.$$

Coarse levels thus include the FAS corrections of all finer levels.

2.2.2 The MLSDC algorithm

The MLSDC scheme introduced here proceeds as follows. The initial condition U_0 and its function evaluation are spread to each of the collocation nodes on the finest level so that the first provisional solution $\boldsymbol{U}_1^0$ is given by

$$\boldsymbol{U}_1^0 = [U_0, \ldots, U_0].$$

A single MLSDC iteration then consists of the following steps:

1. Perform one fine SDC sweep using the values $\boldsymbol{U}_1^k$ and $\boldsymbol{F}_1(\boldsymbol{U}_1^k)$. This will yield provisional updated values $\boldsymbol{U}_1^{k+1}$ and $\boldsymbol{F}_1(\boldsymbol{U}_1^{k+1})$.
2. Sweep from fine to coarse: for each $\ell = 2 \ldots L$:
   (a) Restrict the fine values $\boldsymbol{U}_{\ell-1}^{k+1}$ to the coarse values $\boldsymbol{U}_\ell^k$ and compute $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^k)$.
   (b) Compute the FAS correction $\boldsymbol{\tau}_\ell^k$ using $\boldsymbol{F}_{\ell-1}(\boldsymbol{U}_{\ell-1}^{k+1})$, $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^k)$, and $\boldsymbol{\tau}_{\ell-1}^k$ (if available).
   (c) Perform n_ℓ SDC sweeps with the values on level ℓ beginning with $\boldsymbol{U}_\ell^k$, $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^k)$ and the FAS correction $\boldsymbol{\tau}_\ell^k$. This will yield new values $\boldsymbol{U}_\ell^{k+1}$ and $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^{k+1})$.
3. Sweep from coarse to fine: for each $\ell = L-1 \ldots 1$:
   (a) Interpolate coarse grid correction $\boldsymbol{U}_{\ell+1}^{k+1} - R\boldsymbol{U}_\ell^{k+1}$ and add to $\boldsymbol{U}_\ell^{k+1}$. Recompute new values $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^{k+1})$
   (b) If $\ell > 1$, perform n_ℓ SDC sweeps beginning with values $\boldsymbol{U}_\ell^{k+1}$, $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^{k+1})$ and the FAS correction $\boldsymbol{\tau}_\ell^k$. This will once again yield new values $\boldsymbol{U}_\ell^{k+1}$ and $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^{k+1})$.

Algorithm 2: MLSDC iteration for L levels.

Data: Initial $U_{1,0}^k$ and function evaluations $\boldsymbol{F}_1^k$ from the previous iteration on the fine level.
Result: Solution $\boldsymbol{U}_\ell^{k+1}$ and function evaluations $\boldsymbol{F}_\ell^{k+1}$ on all levels.

 # Perform fine sweep and check convergence criteria
$\boldsymbol{U}_1^{k+1}, \boldsymbol{F}_1^{k+1} \longleftarrow \text{SDCSweep}\big(\boldsymbol{U}_1^k, \boldsymbol{F}_1^k\big)$
if *fine level has converged* **then**
  return
end

 # Cycle from fine to coarse
for $\ell = 1 \dots L-1$ **do**
  # Restrict, re-evaluate, and save restriction (used later during interpolation)
  for $m = 0 \dots M$ **do**
    $U_{\ell+1,m}^k \longleftarrow \text{Restrict}\big(U_{\ell,m}^{k+1}\big)$
    $F_{\ell+1,m}^k \longleftarrow \text{FEval}\big(U_{\ell+1,m}^{k+1}\big)$
    $\tilde{U}_{\ell+1,m}^k \longleftarrow U_{\ell+1,m}^k$
  end
  # Compute FAS correction and sweep
  $\boldsymbol{\tau}_{\ell+1} \longleftarrow \text{FAS}\big(\boldsymbol{F}_\ell^{k+1}, \boldsymbol{F}_{\ell+1}^k, \boldsymbol{\tau}_\ell\big)$
  $\boldsymbol{U}_{\ell+1}^{k+1}, \boldsymbol{F}_{\ell+1}^{k+1} \longleftarrow \text{SDCSweep}\big(\boldsymbol{U}_{\ell+1}^k, \boldsymbol{F}_{\ell+1}^k, \boldsymbol{\tau}_{\ell+1}\big)$
end

 # Cycle from coarse to fine
for $\ell = L-1 \dots 2$ **do**
  # Interpolate coarse correction and re-evaluate
  for $m = 0 \dots M$ **do**
    $U_{\ell,m}^{k+1} \longleftarrow U_{\ell,m}^{k+1} + \text{Interpolate}\big(U_{\ell+1,m}^{k+1} - \tilde{U}_{\ell+1,m}^k\big)$
    $F_{\ell,m}^{k+1} \longleftarrow \text{FEval}\big(U_{\ell,m}^{k+1}\big)$
  end
  $\boldsymbol{U}_\ell^{k+1}, \boldsymbol{F}_\ell^{k+1} \longleftarrow \text{SDCSweep}\big(\boldsymbol{U}_\ell^{k+1}, \boldsymbol{F}_\ell^{k+1}, \boldsymbol{\tau}_\ell\big)$
end

 # Return to finest level before next iteration
for $m = 0 \dots M$ **do**
  $U_{1,m}^{k+1} \longleftarrow U_{1,m}^{k+1} + \text{Interpolate}\big(U_{2,m}^{k+1} - \tilde{U}_{2,m}^k\big)$
  $F_{1,m}^{k+1} \longleftarrow \text{FEval}\big(U_{1,m}^{k+1}\big)$
end

Note that when interpolating from coarse to fine levels the correction $\boldsymbol{U}_{\ell+1}^{k+1} - R\boldsymbol{U}_{\ell+1}^k$ is interpolated and subsequently added to $\boldsymbol{U}_\ell^{k+1}$ instead of simply overwriting the fine values with interpolated coarse values. Also note that instead of interpolating solution values $\boldsymbol{U}_{\ell+1}^{k+1}$ to $\boldsymbol{U}_\ell^{k+1}$ and immediately re-evaluating the function values $\boldsymbol{F}_\ell(\boldsymbol{U}_\ell^{k+1})$, the change in the function values can be interpolated as well. Doing so reduces the cost of the interpolation step, but possibly at the cost of increasing the number of MLSDC iterations required to reach convergence. Since no significant increase could be observed during our tests, we skip the re-evaluation of the right-hand side and use interpolation of the coarse function values throughout this work. The above is summarized by Algorithm 2.

2.2.3 Semi-implicit MLSDC with compact stencils

In order to achieve higher-order accuracy with finite difference discretizations in space, the use of Mehrstellen discretizations is a common technique especially when using multigrid methods [43]. While the straightforward use of larger stencils leads to larger matrix bandwidths and higher communication costs during parallel runs, *high-order compact* schemes allow for high-order accuracy with stencils of minimal extent [41]. The compact stencil for a given discretization is obtained by approximating the leading order error term by a finite difference approximation of the right-hand side, resulting in a weighting matrix. Discretizing e.g. the heat equation $\underline{u}_t = \nabla^2 \underline{u}$ in space[1] yields

$$Wu_t = Au$$

with system matrix A and weighting matrix W. Formally, the discrete Laplacian is given by $W^{-1}A$. Using this approach, a fourth-order approximation of the Laplacian can be achieved using only nearest neighbors (three-point stencil in 1D, nine-point-stencil in 2D, 19-point stencil in 3D). For further reading on compact schemes we refer to [31,41,43].

The presence of a weighting matrix requires some modifications to MLSDC. We start with the semi-implicit SDC update equation (4) given by

$$\begin{aligned} U_{m+1}^{k+1} = U_m^{k+1} &+ \Delta t_m \left[f^E(U_m^{k+1}, t_m) - f^E(U_m^k, t_m) \right] \\ &+ \Delta t_m \left[f^I(U_{m+1}^{k+1}, t_m) - f^I(U_{m+1}^k, t_m) \right] + \Delta t \, S_m^k. \end{aligned} \tag{7}$$

Next, we assume a linear, autonomous implicit part $f^I(U,t) = f^I(U) = W^{-1}AU$ for a spatial vector U with sparse matrices W and A stemming from the discretization of the Laplacian with compact stencils. Furthermore, we define

$$\tilde{f}^I(U) = AU$$

so that

$$\tilde{f}^I(U) = W f^I(U). \tag{8}$$

With these definitions (7) becomes

$$\begin{aligned} \left(I - \Delta t_m W^{-1} A\right) U_{m+1}^{k+1} = U_m^{k+1} &+ \Delta t_m \left[f^E(U_m^{k+1}, t_m) - f^E(U_m^k, t_m) \right] \\ &- \Delta t_m W^{-1} A U_{m+1}^k + \Delta t \, S_m^k. \end{aligned}$$

Since the operator $\left(I - \Delta t_m W^{-1} A\right)$ is not sparse, we avoid computing with it by multiplying the equation above by W, so that

$$\begin{aligned} (W - \Delta t_m A) U_{m+1}^{k+1} = W U_m^{k+1} &+ \Delta t_m W \left[f^E(U_m^{k+1}, t_m) - f^E(U_m^k, t_m) \right] \\ &- \Delta t_m \tilde{f}^I(U_{m+1}^k) + \Delta t \, \tilde{S}_m^k \end{aligned} \tag{9}$$

where $\tilde{S}_m^k$ now represents the m^{th} row of $\boldsymbol{S}\tilde{\boldsymbol{F}}^k(\boldsymbol{U}^k)$, using $W f^E(U_m^k, t_m)$ and $\tilde{f}^I(U_m^k)$ instead of $f^E(U_m^k, t_m)$ and $f^I(U_m^k)$ as integrands, that is $\tilde{S}_m^k = \sum_{j=0}^{M} s_{m,j} \left(W f^E(U_j^k, t_j) + \tilde{f}^I(U_j^k) \right)$.

[1] We adopt here and in the upcoming examples the following notation: Solutions of PDEs are denoted with an underline, e.g. $\underline{u}$, and depend continuously on one or more spatial variables and a time variable. Discretizing a PDE in space by the method of lines results in an IVP with dimension N equal to the degrees of freedom of the spatial discretization. The solution of such an IVP is a vector-valued function denoted by a lower case letter, e.g. u, and depends continuously on time. The numerical approximation of u at some point in time t_m is denoted by a capital letter, e.g. U_m^k, where k corresponds to the iteration number.

While this equation avoids the inversion of W, the computation of the residual does not. By equation (5), the m^{th} component of the residual at iteration k reads either

$$r_m^k = U_0 + \Delta t \left(\boldsymbol{Q}\boldsymbol{F}(\boldsymbol{U}^k) \right)_m - U_m^k,$$

or, after multiplication with W,

$$W r_m^k = W U_0 + \Delta t \left(\boldsymbol{Q}\tilde{\boldsymbol{F}}(\boldsymbol{U}^k) \right)_m - W U_m^k.$$

Both equations require the solution of a linear system with matrix W, either to compute the components of $\boldsymbol{F}(\boldsymbol{U}^k)$ from (8) or to retrieve r_m^k from $W r_m^k$. Note that the subscript m denotes here the m^{th} column. Thus, we either need to obtain r_m^k from $W r_m^k$ (in case $W f^E$ is stored during the SDC sweep) or f^I from $\tilde{f}^I$ (in case f^E is stored). In either case, solving a linear system with the weighting matrix becomes inevitable for the computation of the formally correct residual.

Furthermore, evaluating (6) for the FAS correction also requires the explicit use of f^E and $f^I = W^{-1}\tilde{f}^I$ to compute $R\boldsymbol{Q}_\ell \boldsymbol{F}_\ell(\boldsymbol{U}_\ell)$. Moreover, from (9) we note that weighted SDC sweeps on coarse levels $\ell+1$ require the computation of $W_{\ell+1}\tau_{\ell+1,m}$ on all coarse nodes $\boldsymbol{t}_\ell$ so that $\boldsymbol{Q}_{\ell+1}\boldsymbol{F}_{\ell+1}(R\boldsymbol{U}_\ell)$ can be replaced by $\boldsymbol{Q}_{\ell+1}\tilde{\boldsymbol{F}}_{\ell+1}(R\boldsymbol{U}_\ell)$. For spatial discretizations in which both parts f^E and f^I of the right-hand side make use of weighting matrices W^E and W^I or e.g. for finite element discretizations with a mass matrix, we note that similar modifications to the MLSDC scheme as presented here must be made. The investigation of MLSDC for finite element discretizations is left for future work.

2.2.4 Coarsening strategies

The goal in MLSDC methods is to reduce the total cost of the method by performing SDC sweeps on coarsened levels at reduced computational cost. In this section we describe the three types of spatial coarsening used in the numerical examples:

1. REDUCED RESOLUTION IN SPACE: Use fewer degrees of freedom for the spatial representation (e.g. nodes, cells, points, particles, etc.) on the coarse levels. This directly translates into significant computational savings for evaluations of f, particularly for 3D problems. This approach requires spatial interpolation and restriction operators to transfer the solution between levels.

2. REDUCED ORDER IN SPACE: Use a spatial discretization on the coarse levels that is of reduced order. Lower-order finite difference stencils, for example, are typically cheaper to evaluate than higher-order ones, see [37] for an application of this strategy for the time-parallel Parareal method.

3. REDUCED IMPLICIT SOLVE IN SPACE: Use only a few iterations of a spatial solver in every substep, if an implicit or implicit-explicit method is used in the SDC sweeps. By not solving the linear or nonlinear system in each SDC substep to full accuracy, savings in execution time can be achieved.

We note that a fourth possibility not pursued here is to use a simplified physical representation of the problem on coarse levels. This approach requires a detailed understanding of the problem to derive suitable coarse level models and appropriate coarsening and interpolation operators. Similar ideas have been studied for Parareal in [14, 23].

The spatial coarsening strategies outlined above can significantly reduce the cost of a coarse level SDC substep, but do not affect the number of substeps used. In principle, it is also possible to reduce the number of quadrature nodes on coarser levels as in the ladder schemes mentioned in the introduction. In this paper, no such temporal coarsening is applied and we focus on the application of spatial coarsening strategies which leads to a large reduction of the runtime for coarse level sweeps.

2.2.5 Transfer operators

In order to apply Strategy 1 and reduce the number of spatial degrees of freedom, transfer operators between different levels are required. In the tests presented here that are based on finite difference discretizations on simple cartesian meshes, the spatial degrees of freedom are aligned, so that simple injection can be used for restriction.

We have observed that the order of the used spatial interpolation has a strong impact on the convergence of MLSDC. While global information transfer when using e.g. spectral methods does not influence the convergence properties of MLSDC, the use of local Lagrangian interpolation for finite difference stencils has to be applied with care. In numerical experiments not documented here, MLSDC with simple linear interpolation required twice as many iterations as MLSDC with fifth-order spatial interpolation. Further, low resolutions in space combined with low-order interpolation led to significant degradation of the convergence speed of MLSDC, while high spatial resolutions were much less sensitive. Throughout the paper, Strategy 1 is applied with third-order Lagrangian interpolation, which has proven to be sufficient in all cases studied here.

We note that the transfer operators would be different if e.g. finite elements were used and operators between element spaces of different order and/or on different meshes would be required.

2.2.6 Stability of SDC and MLSDC

Stability domains for SDC are presented in e.g. [16]. The stability of semi-implicit SDC is addressed in [32] and the issue of order reduction for stiff problems is discussed. Split SDC methods are further analyzed theoretically and numerically in [19]. A stability analysis for MLSDC is complicated by the fact that it would need to consider the effects of the different spatial coarsening strategies laid out in 2.2.4. Therefore, it cannot simply use Dahlquist's test equation but has to resort to some well-defined PDE examples in order to assess stability. Hence, for MLSDC the results presented here are experimental but development of a theory for the convergence properties of MLSDC is ongoing work. However, in all examples presented below, stability properties of SDC and MLSDC appeared to be comparable, but a comprehensive analysis is left for future work.

3 Numerical Examples

In this section we investigate the performance of MLSDC for four numerical examples. First, in order to demonstrate that the FAS correction in MLSDC is not unusable for hyperbolic problems per se, performance for the 1D wave equation is studied in §3.1. To investigate performance for a nonlinear problem, MLSDC is then applied to the 1D viscous Burgers' equation in §3.2. A detailed investigation of different error components is given and we

verify that the FAS corrections allow the solutions on coarse levels to converge to the accuracy determined by the discretization on the *finest* level. The 2D Navier-Stokes equations in vorticity-velocity form are solved in §3.3, showing again a reduction of the number of required iterations by MLSDC, although using a coarsened spatial resolution is found to have a negative impact on convergence, if the fine level is already under-resolved. In §3.4, a FORTRAN implementation of MLSDC is applied to the three-dimensional Burgers' equation and it is demonstrated that the reduction in fine level sweeps translates into a significant reduction of computing time. Throughout all examples, we make use of a linear geometric multigrid solver [10,43] with JOR relaxation in 3D and SOR relaxation 1D and 2D as smoothers, to solve the linear problems in the implicit part as well as to solve the linear system with the weighting matrix for the residual and the FAS correction. The parallel implementation of the multigrid solver used for the last example is described in [4].

In the examples below, we compare the number of sweeps on the fine and most expensive level required by SDC or MLSDC to converge up to a set tolerance. For SDC, which sweeps only on the fine level, this number is identical to the number of iterations. For MLSDC, each iteration consists of one cycle through the level hierarchy, starting from the finest level, going up to the coarsest and then down again, with one SDC sweep on each level on the way up and down, cf. Algorithm 2. Except for the last iteration, the final fine sweep is also the first fine sweep of the next iteration, so that for MLSDC the number of fine sweeps is equal to the number of iterations plus one. Note that a factor of two coarsening in the spatial resolution in each dimension yields a factor of eight reduction in degrees of freedom in three dimensions, which makes coarse level sweeps significantly less expensive.

3.1 Wave equation

For spatial multigrid, the FAS formalism is mostly derived and analyzed for stationary elliptic or parabolic problems, although there are examples of applications to hyperbolic problems as well [1,38]. Here, as a first test, we investigate the performance of MLSDC for a simple 1D wave equation to verify that the FAS procedure as used in MLSDC does not break down for a hyperbolic problem per se. The problem considered here, with the wave equation written as a first order system, reads

$$u_t(x,t) + v_x(x,t) = 0$$
$$v_t(x,t) + u_x(x,t) = 0$$

on $x \in [0,1]$ with periodic boundary conditions and

$$u(x,0) = \exp\left(-\frac{1}{2}\left(\frac{x-0.5}{0.1}\right)^2\right), \quad v(x,0) = 0$$

for $0 \leq t \leq T$. For the spatial derivatives, centered differences of 4^{th} order with 128 points are used on the fine level and of 2^{nd} order with 64 points on the coarse. Both SDC and MLSDC perform 40 timesteps of length $\Delta t = 0.025$ to integrate up to $T = 1.0$ and iterations on each step are performed until $\left\|\boldsymbol{r}^k\right\|_\infty \leq 5 \times 10^{-8}$. The average number of fine level sweeps over all steps for SDC and MLSDC is shown in Table 1 for three different values of M. In all cases, MLSDC leads to savings in terms of required fine level sweeps. We note that for a fine level spatial resolution of only 64 points, using spatial coarsening has a significant negative effect on the performance of MLSDC (not documented here): This suggests that for a problem which is spatially under-resolved on the finest level, further coarsening the spatial resolution within MLSDC might hurt performance, see also §3.3.

M	SDC	MLSDC(1,2)
3	18.5	11.1
5	17.6	10.6
7	14.3	8.2

Table 1: Average number of fine level sweeps over all time-steps of SDC and MLSDC for the wave equation example to reach a residual of $\left\|\boldsymbol{r}^k\right\|_\infty \leq 5\times 10^{-8}$. The numbers in parentheses after MLSDC indicate the used coarsening strategies, see §2.2.4.

3.2 1D viscous Burgers' equation

In this section we investigate the effect of coarsening in MLSDC by considering the nonlinear viscous Burgers' equation

$$\begin{aligned} \underline{u}_t + \underline{u}\cdot\underline{u}_x &= \nu \underline{u}_{xx},\ x\in[-1,1],\ t\in[0,t_{\text{end}}] \\ \underline{u}(x,0) &= u^0(x) \\ \underline{u}(-1,t) &= \underline{u}(1,t), \end{aligned} \tag{10}$$

with $\nu > 0$ and initial condition

$$u^0(x) = \exp\left(-\frac{x^2}{\sigma^2}\right), \quad \sigma = 0.1$$

corresponding to a Gaussian peak strongly localized around $x = 0$. We denote the evaluation of the continuous function $\underline{u}$ on a given spatial mesh with points $(x_i)_{i=1,\dots,N}$ with a subscript N, so that

$$\underline{u}_N(t) := (\underline{u}(x_i,t))_{i=1,\dots,N} \in \mathbb{R}^N.$$

Discretization of (10) in space then yields an initial value problem

$$\begin{aligned} u_t(t) &= f_N(u(t)), \quad u(t)\in\mathbb{R}^N, \quad t\in[0,t_{\text{end}}] \\ u(0) &= \underline{u}_N^0 \end{aligned} \tag{11}$$

with solution u. Finally, we denote by $U_{N,M,\Delta t,k} \in \mathbb{R}^N$ the result of solving (11) with k iterations of MLSDC using a timestep of Δt, M substeps (or $M+1$ Lobatto collocation nodes), and an N-point spatial mesh on the finest level over one time step.

Two runs are performed here, solving (10) with $\nu = 1.0$ and $\nu = 0.1$ with a single MLSDC timestep $t_{\text{end}} = \Delta t = 0.01$. MLSDC with two levels with 7 Gauss-Lobatto collocation points is used with a spatial mesh of $N = 256$ points on the fine level, and $N = 128$ on the coarse level (Strategy 1). The advective term is discretized using a 5^{th}-order WENO finite difference method [27] on the fine level and a simple 1^{st}-order upwind scheme on the coarse level. For the Laplacian, a 4^{th}-order compact stencil is used on the fine level and a 2^{nd}-order stencil is used on the coarse level (Strategy 2). The advective term is treated explicitly while the diffusion term is treated implicitly. The resulting linear system is solved using a linear multigrid solver with a tolerance of 5×10^{-14} on the fine level but solved only approximately using a single V-cycle on the coarse level (Strategy 3). A fixed number of $K = 80$ MLSDC iterations is performed here without setting a tolerance for the MLSDC residual.

In order to assess the different error components, a reference PDE solution $\underline{u}_N(\Delta t)$ is computed with a single-level SDC scheme on a mesh with $N = 1,024$ points using $M+1 = 9$ and $\Delta t = 10^{-4}$. An ODE solution $u(\Delta t)$ is computed by running single-level SDC using

$M+1=9$, $\Delta t = 10^{-4}$ and the same spatial discretization as on the fine level of the MLSDC run. Finally, the collocation solution $u^{\text{coll}}(\Delta t)$ is computed by performing 100 iterations of single-level SDC with $M+1=7$ and again the same spatial discretization as the MLSDC fine level. Reference ODE and collocation solutions are computed for the coarse level using the same parameters and the MLSDC coarse level spatial discretization.

3.2.1 Error components in MLSDC

The relative error of the fully discrete MLSDC solution to the analytical solution $\underline{u}$ of the PDE (10) after a single timestep of length Δt is given by

$$\varepsilon^{\text{PDE}} := \frac{\left\|\underline{u}_N(\Delta t) - U_{N,M,\Delta t,k}\right\|}{\|\underline{u}_N(\Delta t)\|}, \tag{12}$$

where $\|\cdot\|$ denotes some norm on $\mathbb{R}^N$. All errors are hereafter reported using the maximum norm $\|\cdot\|_\infty$. The error ε^{PDE} includes contributions from three sources

$$\begin{aligned}
\varepsilon_N &:= \frac{\|\underline{u}_N(\Delta t) - u(\Delta t)\|}{\|\underline{u}_N(\Delta t)\|} \approx \text{(i) – relative spatial error,} \\
\varepsilon_{\Delta t} &:= \frac{\left\|u(\Delta t) - u^{\text{coll}}(\Delta t)\right\|}{\|\underline{u}_N(\Delta t)\|} \approx \text{(ii) – relative temporal error,} \\
\varepsilon^{\text{coll}} &:= \frac{\left\|u^{\text{coll}}(\Delta t) - U_{N,M,\Delta t,k}\right\|}{\|\underline{u}_N(\Delta t)\|} \approx \text{(iii) – iteration error,}
\end{aligned}$$

with u^{coll} denoting the exact solution of the collocation equation (4). Here, (i) is the spatial discretization error; (ii) is the temporal discretization error, which is the error from replacing the analytical Picard formulation (1) with the discrete collocation problem (4); and (iii) is the error from solving the collocation equation approximately using the MLSDC iteration. The PDE error (12) can be estimated using the triangle inequality according to

$$\varepsilon^{\text{PDE}} \leq \varepsilon^N + \varepsilon_{\Delta t} + \varepsilon^{\text{coll}}.$$

In addition to the PDE error, we define the error between the MLSDC solution and the analytical solution of the semi-discrete ODE (11) as

$$\varepsilon^{\text{ODE}} := \frac{\left\|u(\Delta t) - U_{N,M,\Delta t,k}\right\|}{\|\underline{u}_N(\Delta t)\|} \leq \varepsilon_{\Delta t} + \varepsilon^{\text{coll}}. \tag{13}$$

Note that ε^{ODE} contains contributions from (ii) and (iii), and once the MLSDC iteration has converged, error (13) reduces to the error arising from replacing the exact Picard integral (1) by the collocation formula (4).

The three different error components of MLSDC, ε^{PDE}, ε^{ODE} and $\varepsilon^{\text{coll}}$ are expected to saturate at different levels as $k \to \infty$ according to

$$\begin{aligned}
\varepsilon^{\text{PDE}} &\to \max\{\varepsilon_N, \varepsilon_{\Delta t}\}, \\
\varepsilon^{\text{ODE}} &\to \varepsilon_{\Delta t}, \text{ and} \\
\varepsilon^{\text{coll}} &\to 0.
\end{aligned}$$

The crucial point here is that due to the presence of the FAS correction included in MLSDC, we expect ε^{PDE}, ε^{ODE} and $\varepsilon^{\text{coll}}$ on *all* levels to saturate at values of ε_N and $\varepsilon_{\Delta t}$ determined by the discretization used on the *finest* level. That is, the FAS correction should allow MLSDC to represent the solution on all coarse levels to the same accuracy as on the finest level. This is verified in §3.2.2.

$\nu = 0.1$		$\nu = 1.0$	
Method	# Fine sweeps	Method	# Fine sweeps
SDC	4	SDC	12
MLSDC	3	MLSDC	7

Table 2: Number of fine level sweeps required to reach a residual of $\|\boldsymbol{r}^k\|_\infty \leq 10^{-5}$ for SDC and multi-level SDC for Burgers' equation with $\nu = 0.1$ and $\nu = 1.0$.

3.2.2 Convergence of MLSDC on all levels

Figure 1 shows the three error components ε^{PDE} (green squares), ε^{ODE} (blue diamonds) and $\varepsilon^{\text{coll}}$ (red circles) for $\nu = 0.1$ (upper) and $\nu = 1.0$ (lower) plotted against the iteration number k. The errors on the fine level are shown on the left in Figures 1a and 1c, while errors on the coarse mesh are shown on the right. Furthermore, the estimated spatial discretization error ε_N (dashed) and temporal discretization error $\varepsilon_{\Delta t}$ (dash-dotted) are indicated by black lines.

For $\nu = 0.1$, we note that the PDE error ε^{PDE} on the fine level (Figures 1a and 1c) saturates – as expected – at a level determined by the spatial discretization error ε_N; and the ODE error ε^{ODE} saturates at the level of the temporal discretization error $\varepsilon_{\Delta t}$. The collocation error $\varepsilon^{\text{coll}}$ saturates at near machine accuracy. Increasing the viscosity to $\nu = 1.0$, the spatial error remains at about 10^{-7} on the fine level but the time discretization error significantly increases compared to $\nu = 0.1$. Thus in Figure 1c, both the PDE and the ODE error saturate at the value indicated by $\varepsilon_{\Delta t}$. Once again, the collocation error goes down to machine accuracy, although the rate of convergence is somewhat slower compared to $\nu = 0.1$.

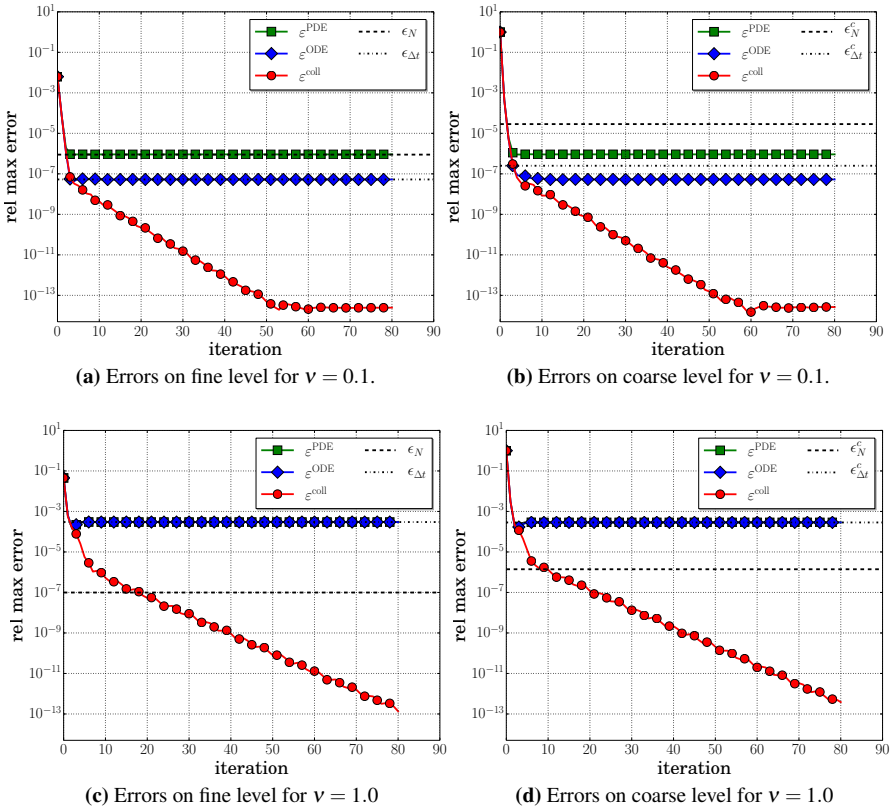
On the coarse level (Figures 1b and 1d), the estimated spatial error ε_N is noticeably higher because the values of N are smaller and the order of the spatial discretization is lower. However, as expected, the coarse level error of MLSDC saturates at values determined by the accuracy of the *finest* level. The saturation of ε^{PDE} and ε^{ODE} are identical in the left and right figures, despite the difference in ε_N and $\varepsilon_{\Delta t}$. This demonstrates that the FAS correction in MLSDC allows the solutions on coarse levels to obtain the accuracy of the finest level as long as sufficiently many iterations are performed.

3.2.3 Required iterations

Table 2 shows the number of fine level sweeps required by SDC and MLSDC to reduce the infinity norm of the residual $\boldsymbol{r}^k$, see (5), below 10^{-5}. For both setups, $\nu = 0.1$ as well as $\nu = 1.0$, MLSDC reduces the number of required fine sweeps compared to single-level SDC. In turn, however, MLSDC adds some overhead from coarse level sweeps. If these are cheap enough, the reduced iteration number will result in reduced computing time, cf. §3.4.

3.2.4 Stopping criteria

Note that the overall PDE error of the solution is not reduced further by additional iterations once $\varepsilon^{\text{coll}} \leq \max\{\varepsilon_N, \varepsilon_{\Delta t}\}$. In Figures 1a–1d, this corresponds to the point where the line with red circles (iteration error) drops below the dot-dashed line (indicating $\varepsilon_{\Delta t}$) or dashed line (indicating ε_N). The MLSDC solution, however, continues to converge to the collocation solution. In a scenario where the PDE error is the main criterion for the quality of a solution, iterating beyond ε^{PDE} no longer improves the solution. This suggests adaptively setting the tolerance for the residual of the MLSDC iteration in accordance with error estimators for ε_{N} and $\varepsilon_{\Delta t}$ to avoid unnecessary further iterations.



(a) Errors on fine level for $\nu = 0.1$.

(b) Errors on coarse level for $\nu = 0.1$.

(c) Errors on fine level for $\nu = 1.0$

(d) Errors on coarse level for $\nu = 1.0$

Fig. 1: Errors on fine and coarse level of MLSDC vs. iteration count. The dashed line indicates the spatial error ε_N while the dot-dashed line indicates the temporal error $\varepsilon_{\Delta t}$. The red circles indicate the difference $\varepsilon^{\mathrm{coll}}$ between MLSDC and the collocation solution, the blue diamonds indicate the difference $\varepsilon^{\mathrm{ODE}}$ between MLSDC and the ODE solution and the green squares indicate the difference $\varepsilon^{\mathrm{PDE}}$ between MLSDC and the PDE solution. In (c) and (d), $\varepsilon^{\mathrm{ODE}}$ is nearly identical to $\varepsilon^{\mathrm{PDE}}$. Note how the FAS correction in MLSDC allows the coarse level to attain the same accuracy as the fine level solution: the saturation limits on the fine and coarse mesh are identical.

3.3 Shear layer instability

In this example, we study the behavior of MLSDC in the case where the exact solution is not well resolved. We consider a shear layer instability in a 2D doubly periodic domain governed by the vorticity-velocity formulation of the 2D Navier-Stokes equations given by

$$\underline{\omega}_t + \underline{u} \cdot \nabla \underline{\omega} = \nu \nabla^2 \underline{\omega}$$

with velocity $\underline{u} \in \mathbb{R}^2 \times [0,\infty)$, vorticity $\underline{\omega} = \nabla \times \underline{u} \in \mathbb{R}^{\times}[0,\infty)$ and viscosity $\nu \in \mathbb{R}^+$. We consider the spatial domain $[0,1]^2$ with periodic boundary conditions in all directions and the initial conditions

$$\underline{u}_1^0(x,y) = -1.0 + \tanh(\rho(0.5-y)) + \tanh(\rho(y-0.25))$$
$$\underline{u}_2^0(x,y) = -\delta \sin(2\pi(x+0.25)).$$

These initial conditions correspond to two horizontal shear layers, of "thickness" $\rho = 50$, at $y = 0.75$ and $y = 0.25$, with a disturbance of magnitude $\delta = 0.05$ in the vertical velocity $\underline{u}_2$. As in §3.2, the system is split into implicit/explicit parts according to

$$\underline{\omega}_t = f^E(\underline{\omega}) + f^I(\underline{\omega})$$

where

$$f^E(\underline{\omega}) = -\underline{u} \cdot \nabla \underline{\omega}$$
$$f^I(\underline{\omega}) = \nu \nabla^2 \underline{\omega}.$$

While the implicit term f^I is discretized and solved as before, we apply a streamfunction approach for the explicit term f^E: for periodic boundary conditions, we can assume $\underline{u} = \nabla \times \underline{\psi}$ for a solenoidal streamfunction $\underline{\psi}$. Thus,

$$\underline{\omega} = \nabla \times (\nabla \times \underline{\psi}) = -\nabla^2 \underline{\psi}.$$

We refer to [9] for more details. To compute $f^E_{p,N}(\omega)$ with order-p operators on an $N \times N$ mesh, we therefore solve the Poisson problem

$$-\nabla^2 \underline{\psi} = \underline{\omega}$$

for $\underline{\psi}$ using the linear multigrid method described previously, calculate the discretized version of $\underline{u} = \nabla \times \underline{\psi}$ and finally compute the discretization of $\underline{u} \cdot \nabla \underline{\omega}$, both with order-$p$ operators.

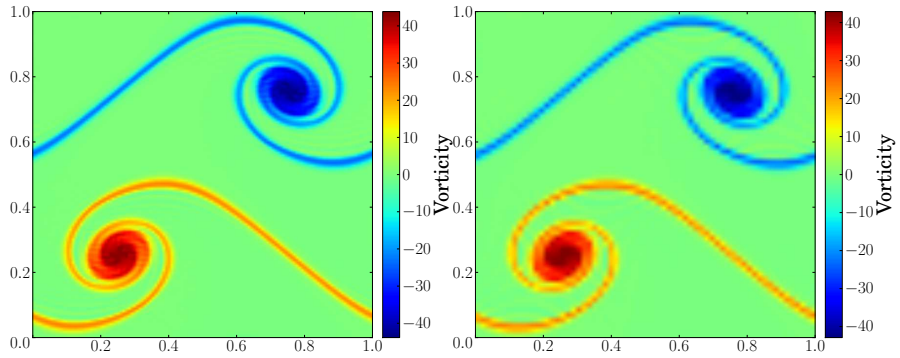
Two levels with $M + 1 = 9$ collocation nodes are used with a 128×128 point spatial mesh and a fourth order stencil on the fine level. Different combinations of coarsening are tested (the numbers in parentheses correspond to the strategies as listed in §2.2.4):

1. MLSDC(1,2) uses a coarsened 64×64 point mesh on the coarse level and second-order stencils.
2. MLSDC(1,2,3(1)) as MLSDC(1,2) but also solves the implicit linear systems in the coarse SDC sweep only approximately with a single V-cycle.
3. MLSDC(1,2,3(2)) as MLSDC(1,2,3(1)) but with two V-cycles.
4. MLSDC(2,3(1)) uses also a 128×128 point mesh on the coarse level, but second-order stencils and approximate linear solves using a single V-cycle.

The simulation computes 256 timesteps of MLSDC up to a final time $t = 1.0$. As reference, a classical SDC solution is computed using 1024 timesteps with $M + 1 = 13$ collocation nodes and the fine level spatial discretization. Both SDC and MLSDC iterate until the residual satisfies $\left\|\boldsymbol{r}^k\right\|_\infty \leq 10^{-12}$.

3.3.1 Vorticity field on all levels

Figure 2 shows the vorticity field at the end of the simulation on the fine and the coarse level. The relative maximum error ε^{ODE} at time $t = 1$ is approximately 10^{-12} (which corresponds to the spatial and temporal residual thresholds that were used for all runs in this example). We note that simply running SDC with the coarse level spatial discretization from MLSDC(1,2) gives completely unsatisfactory results (not shown): spurious vortices exist in addition to the two correct vortices and strong spurious oscillations are present in the vorticity field. In contrast, the coarse level solution from MLSDC shown in Figure 2b looks reasonable, again because of the FAS correction.



(a) MLSDC, fine level: 128×128, $p = 4$ **(b)** MLSDC, coarse level: 64×64, $p = 2$

Fig. 2: Vorticity of the solution of the shear layer instability at $t = 1.0$ on the fine level (left) and coarse level (right) using MLSDC(1,2,3(1)).

3.3.2 Required iterations

Table 3 shows the average number of fine level sweeps over all timesteps required by SDC and MLSDC to converge. The configurations MLSDC(1,2), MLSDC(1,2,3(1)) and MLSDC(1,2,3(2)) do not reduce the number of sweeps, but instead lead to a small increase. Avoiding a coarsened spatial mesh in MLSDC(2,3(1)), however, saves a small amount of fine sweeps compared to SDC. Note that here, in contrast to the example presented in §3.4, Strategy 1 has a significant negative impact on the performance of MLSDC. This illustrates that coarsening in MLSDC cannot be used in the same way for every problem: a careful adaption of the employed strategies to the problem at hand is necessary.

Method	# Fine sweeps on average
SDC	6.46
MLSDC(1,2)	6.64
MLSDC(1,2,3(1))	6.62
MLSDC(1,2,3(2))	6.64
MLSDC(2,3(1))	5.26

Table 3: Average number of fine level sweeps required to converge for SDC and MLSDC for the shear layer instability. The numbers indicate the different coarsening strategies.

3.4 Three-dimensional viscous Burgers' equation

To demonstrate that MLSDC can not only reduce iterations but also runtime, we consider viscous Burgers' equation in three dimensions

$$\underline{u}_t(\mathbf{x},t) + \underline{u}(\mathbf{x},t) \cdot \nabla \underline{u}(\mathbf{x},t) = \nu \nabla^2 \underline{u}(\mathbf{x},t), \quad \mathbf{x} \in [0,1]^3, \quad 0 \leq t \leq 1$$

with $\mathbf{x} = (x,y,z)$, initial value

$$\underline{u}(\mathbf{x},t) = \exp\left(-\frac{(x-0.5)^2 + (y-0.5)^2 + (z-0.5)^2}{\sigma^2}\right), \quad \sigma = 0.1,$$

homogeneous Dirichlet boundary condition and diffusion coefficients $\nu = 0.1$ and $\nu = 1.0$. The problem is solved using a FORTRAN implementation of MLSDC combined with a C implementation of a parallel multigrid solver (PMG) in space [4]. A single timestep of length $\Delta t = 0.01$ is performed with MLSDC, corresponding to CFL numbers from the diffusive term on the fine level, that is

$$C_{\text{diff}} := \frac{\nu \Delta t}{\Delta x^2},$$

of about $C_{\text{diff}} = 66$ (for $\nu = 0.1$) and $C_{\text{diff}} = 656$ (for $\nu = 1.0$). The diffusion term is integrated implicitly using PMG to solve the corresponding linear system and the advection term is treated explicitly. Simulations are run on 512 cores on the IBM BlueGene/Q JUQUEEN at the Jülich Supercomputing Centre.

MLSDC is run with $M+1 = 3$, $M+1 = 5$ and $M+1 = 7$ Gauss-Lobatto nodes with a tolerance for the residual of 10^{-5}. Two MLSDC levels are used with all three types of coarsening applied:

1. The fine level uses a 255^3 point mesh and the coarse level 127^3.
2. A 4^{th}-order compact difference stencil for the Laplacian and a 5^{th}-order WENO [27] for the advection term are used on the fine level; a 2^{nd}-order stencil for the Laplacian and a 1^{st}-order upwind scheme for advection on the coarse.
3. The accuracy of the implicit solve on the coarse level is varied by fixing the number of V-cycles of PMG on this level.

Three runs are performed, each with a different number of V-cycles on the coarse level. In the first run, the coarse level linear systems are solved to full accuracy, whereas the second and third runs use one and two V-cycles of PMG on the coarse level, respectively, instead of solving to full accuracy. These cases are referred to as MLSDC(1,2), MLSDC(1,2,3(1)), and MLSDC(1,2,3(2)). On the fine level, implicit systems are always solved to full accuracy (the PMG multigrid iteration reaches a tolerance of reach a tolerance of 10^{-12} or stalls).

Required iterations and runtimes. Table 4 shows both the required fine level sweeps for SDC and MLSDC as well as the total runtimes in seconds for $\nu = 0.1$ and $\nu = 1.0$ for three different values of M. MLSDC(1,2) and MLSDC(1,2,3(2)) in all cases manage to significantly reduce the number of fine sweeps required for convergence in comparison to single-level SDC, typically by about a factor of two. These savings in fine level sweeps translate into runtime savings on the order of $30-40\%$. For 3 and 5 quadrature nodes, there is no negative impact in terms of additional fine sweeps by using a reduced implicit solve on the coarse level and MLSDC(1,2,3(2)) is therefore faster than MLSDC(1,2). However, since coarse level V-cycles are very cheap due to spatial coarsening, the additional savings in runtime are small. For 7 quadrature nodes, using a reduced implicit solve on the coarse level in MLSDC(1,2,3(2)) comes at the price of an additional MLSDC iteration and therefore, MLSDC(1,2) is the fastest variant in this case.

Using only a single V-cycle for implicit solves on the coarse grid in MLSDC(1,2,3(1)) results in a modest to significant increase in the number of required MLSDC iterations compared to MLSDC(1,2,3(2)) in almost all cases. The only exception is the run with 3 nodes and $\nu = 0.1$. Therefore, MLSDC(1,2,3(1)) is typically significantly slower than MLSDC(1,2) or MLSDC(1,2,3(2)) and not recommended for use in three dimensions. For 7 quadrature nodes, using only a single V-cycle leads to a dramatic increase in the number of required fine sweeps and MLSDC becomes much slower than single level SDC, indicating that the inaccurate coarse level has a negative impact on convergence.

$M+1=3$ **Gauss-Lobatto nodes**

$\nu = 0.1$			$\nu = 1.0$		
Method	F-Sweeps	Runtime (sec)	Method	F-Sweeps	Runtime (sec)
SDC	9	39.4	SDC	16	74.1
MLSDC(1,2)	4	26.2	MLSDC(1,2)	8	49.1
MLSDC(1,2,3(2))	4	25.6	MLSDC(1,2,3(2))	8	47.0
MLSDC(1,2,3(1))	5	29.7	MLSDC(1,2,3(1))	8	46.7

$M+1=5$ **Gauss-Lobatto nodes**

$\nu = 0.1$			$\nu = 1.0$		
Method	F-Sweeps	Runtime (sec)	Method	F-Sweeps	Runtime (sec)
SDC	7	59.5	SDC	18	162.7
MLSDC(1,2)	3	40.8	MLSDC(1,2)	9	105.6
MLSDC(1,2,3(2))	3	39.8	MLSDC(1,2,3(2))	9	101.5
MLSDC(1,2,3(1))	8	79.7	MLSDC(1,2,3(1))	14	142.8

$M+1=7$ **Gauss-Lobatto nodes**

$\nu = 0.1$			$\nu = 1.0$		
Method	F-Sweeps	Runtime (sec)	Method	F-Sweeps	Runtime (sec)
SDC	5	82.4	SDC	17	224.7
MLSDC(1,2)	2	46.1	MLSDC(1,2)	8	139.5
MLSDC(1,2,3(2))	3	57.2	MLSDC(1,2,3(2))	9	148.1
MLSDC(1,2,3(1))	11	147.2	MLSDC(1,2,3(1))	44	560.4

Table 4: Number of required fine level sweeps and resulting runtimes in seconds by SDC and MLSDC for 3D viscous Burgers' equation. The numbers in parentheses after MLSDC indicate the employed coarsening strategies, see §2.2.4. Reduced implicit solves are indicated by $3(n)$ where n indicates the fixed number of multigrid V-cycles. Otherwise, PMG iterates until a residual of 10^{-12} is reached or the iteration stalls. The tolerance for the SDC/MLSDC iteration is 10^{-5}.

4 Discussion

The paper analyzes the multi-level spectral deferred correction method (MLSDC), an extension to the original single-level spectral deferred corrections (SDC) as well as ladder SDC methods. In contrast to SDC, MLSDC performs correction sweeps in time on a hierarchy of discretization levels, similar to V-cycles in classical multigrid. An FAS correction is used to increase the accuracy on coarse levels. The paper also presents a new procedure to incorporate weighting matrices arising in higher-order compact finite difference stencils into the SDC method. The advantage of MLSDC is that it shifts computational work from the fine level to coarse levels, thereby reducing the number of fine SDC sweeps and, therefore, the time-to-solution.

For MLSDC to be efficient, a reduced representation of the problem on the coarse levels has to be used in order to make coarse level sweeps cheap in terms of computing time. Three strategies are investigated numerically, namely (1) using fewer degrees of freedom, (2) reducing the order of the discretization, and (3) reducing the accuracy of the linear solver in implicit substeps on the coarse level. Numerical results are presented for the wave equation, viscous Burgers' equation in 1D and 3D and for the 2D Navier-Stokes equation in vorticity-velocity formulation. It is demonstrated that because of the FAS correction, the solutions on all levels converge up to the accuracy determined by the discretization on the finest level. More significantly, in all four examples, MLSDC can reduce the number of fine level sweeps required to converge compared to single level SDC. For the 3D example this translates directly into significantly reduced computing times in comparison to single-level SDC.

One potential continuation of this work is to investigate reducing the accuracy of implicit solves on the fine level in MLSDC as well. In [40], so called *inexact* spectral deferred corrections (ISDC) methods are considered, where implicit solves at each SDC node are replaced by a small number of multigrid V-cycles. As with MLSDC, the reduced cost ofx implicit solves are somewhat offset by an increase in the number of SDC iterations required for convergence. Nevertheless, numerical results in [40] demonstrate an overall reduction of cost for ISDC methods versus SDC for certain test cases. The optimal combination of coarsening and reducing V-cycles for SDC methods using multigrid for implicit solves appears to be problem-dependent, and an analysis of this topic is in preparation.

The MLSDC algorithm has also been applied to Adaptive Mesh Refinement (AMR) methods popular in finite-volume methods for conservative systems. In the AMR + MLSDC algorithm, each AMR level is associated with its own MLSDC level, resulting in a hierarchy of hybrid space/time discretizations with increasing space/time resolution. When a new (high resolution) level is added to the AMR hierarchy, a new MLSDC level is created. The resulting scheme differs from traditional sub-cycling AMR time-stepping schemes in a few notable aspects: fine level sub-cycling is achieved through increased temporal resolution of the MLSDC nodes; flux corrections across coarse/fine AMR grid boundaries are naturally incorporated into the MLSDC FAS correction; fine AMR ghost cells eventually become high-order accurate through the iterative nature of MLSDC V-cycling; and finally, the cost of implicit solves on all levels decreases with each MLSDC V-cycle as initial guesses improve. Preliminary results suggest that the AMR+MLSDC algorithm can be successfully applied to the compressible Navier-Stokes equations with stiff chemistry for the direct numerical simulation of combustion problems. A detailed description of the AMR+MLSDC algorithm with applications is currently in preparation.

Finally, the impact and performance of the coarsening strategies presented here are also of relevance to the parallel full approximation scheme in space and time (PFASST) [17, 18, 34, 39] algorithm, which is a time-parallel scheme for ODEs and PDEs. Like MLSDC, PFASST employs a hierarchy of levels but performs SDC sweeps on multiple time intervals concurrently with corrections to initial conditions being communicated forward in time during the iterations. Parallel efficiency in PFASST can be achieved because fine SDC sweeps are done in parallel while sweeps on the coarsest level are in essence done serially. In the PFASST algorithm, there is a trade-off between decreasing the cost on coarse levels to improve parallel efficiency and retaining good accuracy on the coarse level to minimize the number of parallel iterations required to converge. In [18] it was shown that, for mesh-based PDE discretizations, using a spatial mesh with fewer points on the coarse level in conjunction with a reduced number of quadrature nodes, led to a method with significant parallel speed up. Incorporating the additional coarsening strategies presented here for MLSDC into PFASST would further reduce the cost of coarse levels, but it is unclear how this might translate into an increase in the number of parallel PFASST iterations required.

Acknowledgements The plots were generated with the Python Matplotlib [26] package. The final publication is available at springerlink.com, see http://dx.doi.org/10.1007/s10543-014-0517-x.

References

1. Alam, J.M., Kevlahan, N.K.R., Vasilyev, O.V.: Simultaneous spacetime adaptive wavelet solution of nonlinear parabolic differential equations. Journal of Computational Physics **214**(2), 829 – 857 (2006).
2. Ascher, U.M., Petzold, L.R.: Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations. SIAM, Philadelphia, PA (2000)

3. Böhmer, K., Hemker, P., Stetter, H.J.: The defect correction approach. In: K. Böhmer, H.J. Stetter (eds.) Defect Correction Methods. Theory and Applications, pp. 1–32. Springer-Verlag (1984)
4. Bolten, M.: Evaluation of a multigrid solver for 3-level Toeplitz and circulant matrices on Blue Gene/Q. In: K. Binder, G. Münster, M. Kremer (eds.) NIC Symposium 2014, pp. 345–352. John von Neumann Institute for Computing (2014). (to appear)
5. Bourlioux, A., Layton, A.T., Minion, M.L.: High-order multi-implicit spectral deferred correction methods for problems of reactive flow. Journal of Computational Physics **189**(2), 651–675 (2003)
6. Bouzarth, E.L., Minion, M.L.: A multirate time integrator for regularized stokeslets. Journal of Computational Physics **229**(11), 4208–4224 (2010)
7. Brandt, A.: Multi-level adaptive solutions to boundary-value problems. Math. Comp. **31**(138), 333–390 (1977)
8. Briggs, W.L.: A Multigrid Tutorial. SIAM, Philadelphia, PA (1987)
9. Chorin, A.J., Marsden, J.E.: A mathematical introduction to fluid mechanics, 2nd edn. Springer-Verlag (1990)
10. Chow, E., Falgout, R.D., Hu, J.J., Tuminaro, R.S., Yang, U.M.: A survey of parallelization techniques for multigrid solvers. In: Parallel Processing for Scientific Computing, SIAM Series of Software, Environements and Tools. SIAM (2006)
11. Christlieb, A., Morton, M., Ong, B., Qiu, J.M.: Semi-implicit integral deferred correction constructed with additive Runge–Kutta methods. Communications in Mathematical Science **9**(3), 879–902 (2011).
12. Christlieb, A., Ong, B., Qiu, J.M.: Comments on high-order integrators embedded within integral deferred correction methods. Communications in Applied Mathematics and Computational Science **4**(1), 27–56 (2009).
13. Christlieb, A., Ong, B.W., Qiu, J.M.: Integral deferred correction methods constructed with high order Runge-Kutta integrators. Mathematics of Computation **79**, 761–783 (2010).
14. Dai, X., Le Bris, C., Legoll, F., Maday, Y.: Symmetric parareal algorithms for hamiltonian systems. ESAIM: Mathematical Modelling and Numerical Analysis **47**, 717–742 (2013).
15. Daniel, J.W., Pereyra, V., Schumaker, L.L.: Iterated deferred corrections for initial value problems. Acta Cient. Venezolana **19**, 128–135 (1968)
16. Dutt, A., Greengard, L., Rokhlin, V.: Spectral deferred correction methods for ordinary differential equations. BIT Numerical Mathematics **40**(2), 241–266 (2000)
17. Emmett, M., Minion, M.L.: Efficient implementation of a multi-level parallel in time algorithm. In: Proceedings of the 21st International Conference on Domain Decomposition Methods, Lecture Notes in Computational Science and Engineering (2012). (In press)
18. Emmett, M., Minion, M.L.: Toward an efficient parallel in time method for partial differential equations. Communications in Applied Mathematics and Computational Science **7**, 105–132 (2012).
19. Hagstrom, T., Zhou, R.: On the spectral deferred correction of splitting methods for initial value problems. Communications in Applied Mathematics and Computational Science **1**(1), 169–205 (2006).
20. Hairer, E., Norsett, S.P., Wanner, G.: Solving Ordinary Differential Equations I, Nonstiff Problems. Springer-Verlag, Berlin (1987)
21. Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems. Springer-Verlag, Berlin (1991)
22. Hansen, A.C., Strain, J.: Convergence theory for spectral deferred correction. Preprint (2006)
23. Haut, T., Wingate, B.: An asymptotic parallel-in-time method for highly oscillatory PDEs. SIAM Journal on Scientific Computing (2014). In press
24. Huang, J., Jia, J., Minion, M.: Accelerating the convergence of spectral deferred correction methods. Journal of Computational Physics **214**(2), 633–656 (2006)
25. Huang, J., Jia, J., Minion, M.: Accelerating the convergence of spectral deferred correction methods. Journal of Computational Physics **214**(2), 633 – 656 (2006).
26. Hunter, J.D.: Matplotlib: A 2D graphics environment. Computing In Science & Engineering **9**(3), 90–95 (2007)
27. Jiang, G.S., Shu, C.W.: Efficient implementation of weighted ENO schemes. Journal of Computational Physics **126**, 202–228 (1996)
28. Layton, A.T.: On the efficiency of spectral deferred correction methods for time-dependent partial differential equations. Applied Numerical Mathematics **59**(7), 1629 – 1643 (2009).
29. Layton, A.T., Minion, M.L.: Conservative multi-implicit spectral deferred correction methods for reacting gas dynamics. Journal of Computational Physics **194**(2), 697–715 (2004)
30. Layton, A.T., Minion, M.L.: Implications of the choice of quadrature nodes for Picard integral deferred corrections methods for ordinary differential equations. BIT Numerical Mathematics **45**, 341–373 (2005)
31. Lele, S.K.: Compact finite difference schemes with spectral-like resolution. Journal of Computational Physics **103**(1), 16–42 (1992)

32. Minion, M.L.: Semi-implicit spectral deferred correction methods for ordinary differential equations. Communications in Mathematical Sciences **1**(3), 471–500 (2003)
33. Minion, M.L.: Semi-implicit projection methods for incompressible flow based on spectral deferred corrections. Applied numerical mathematics **48**(3), 369–387 (2004)
34. Minion, M.L.: A hybrid parareal spectral deferred corrections method. Communications in Applied Mathematics and Computational Science **5**(2), 265–301 (2010).
35. Pereyra, V.: Iterated deferred corrections for nonlinear operator equations. Numerische Mathematik **10**, 316–323 (1966)
36. Pereyra, V.: On improving an approximate solution of a functional equation by deferred corrections. Numerische Mathematik **8**, 376–391 (1966)
37. Ruprecht, D., Krause, R.: Explicit parallel-in-time integration of a linear acoustic-advection system. Computers & Fluids **59**(0), 72 – 83 (2012).
38. South, J.C., Brandt, A.: Application of a multi-level grid method to transonic flow calculations. In: Transonic flow problems in turbomachinery, pp. 180–206. Hemisphere (1977)
39. Speck, R., Ruprecht, D., Krause, R., Emmett, M., Minion, M., Winkel, M., Gibbon, P.: A massively space-time parallel N-body solver. In: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis, SC '12, pp. 92:1–92:11. IEEE Computer Society Press, Los Alamitos, CA, USA (2012).
40. Speck, R., Ruprecht, D., Minion, M., Emmett, M., Krause, R.: Inexact spectral deferred corrections using single-cycle multigrid (2014). ArXiv:1401.7824 [math.NA]
41. Spotz, W.F., Carey, G.F.: A high-order compact formulation for the 3D poisson equation. Numerical Methods for Partial Differential Equations **12**(2), 235–243 (1996)
42. Stetter, H.J.: Economical global error estimation. In: R.A. Willoughby (ed.) Stiff Differential Systems, pp. 245–258 (1974)
43. Trottenberg, U., Oosterlee, C.W.: Multigrid: Basics, Parallelism and Adaptivity. Academic Press (2000)
44. Weiser, M.: Faster SDC convergence on non-equidistant grids with DIRK sweeps (2013). ZIB Report 13–30
45. Xia, Y., Xu, Y., Shu, C.W.: Efficient time discretization for local discontinuous galerkin methods. Discrete and Continuous Dynamical Systems – Series B **8**(3), 677 – 693 (2007).
46. Zadunaisky, P.: A method for the estimation of errors propagated in the numerical solution of a system of ordinary differential equations. In: G. Contopoulus (ed.) The Theory of Orbits in the Solar System and in Stellar Systems. Proceedings of International Astronomical Union, Symposium 25 (1964)