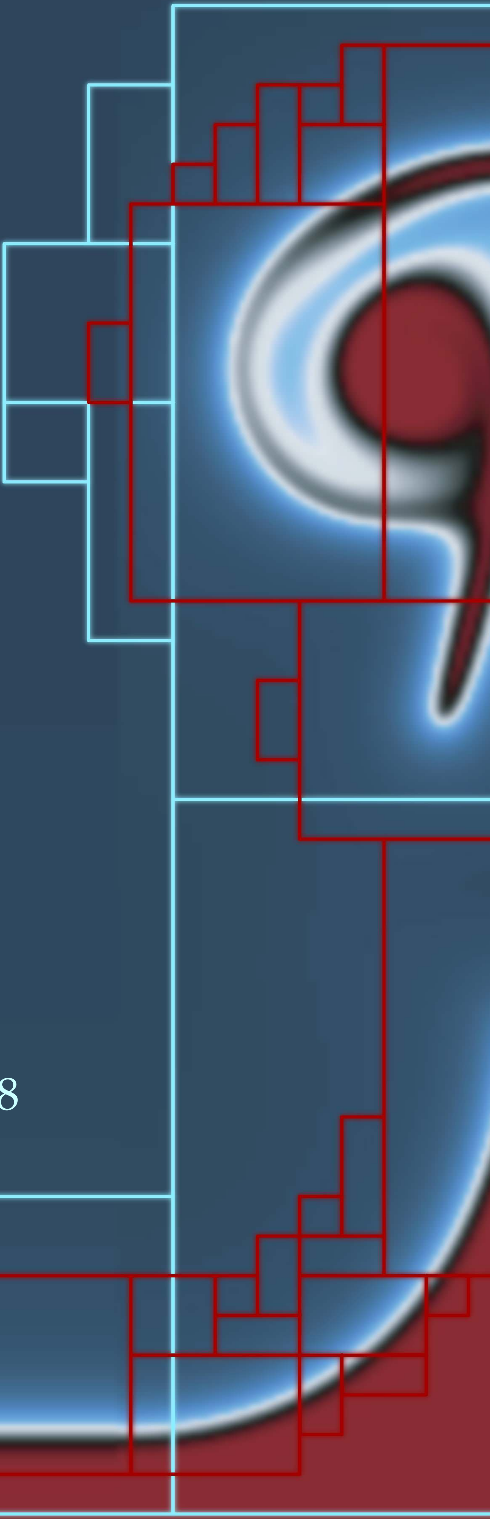
Communications in Applied Mathematics and Computational Science

vol. 13 no. 1 2018



msp

Communications in Applied Mathematics and Computational Science

msp.org/camcos

See inside back cover or msp.org/camcos for submission instructions.

The subscription price for 2018 is US $100/year for the electronic version, and $150/year (+$15, if shipping outside the US) for print and electronic. Subscriptions, requests for back issues from the last three years and changes of subscriber address should be sent to MSP.

Communications in Applied Mathematics and Computational Science (ISSN 2157-5452 electronic, 1559-3940 printed) at Mathematical Sciences Publishers, 798 Evans Hall #3840, c/o University of California, Berkeley, CA 94720-3840, is published continuously online. Periodical rate postage paid at Berkeley, CA 94704, and additional mailing offices.

CAMCoS peer review and production are managed by EditFLOW® from MSP.

PUBLISHED BY

mathematical sciences publishers

nonprofit scientific publishing

http://msp.org/

COMM. APP. MATH. AND COMP. SCI.
Vol. 13, No. 1, 2018
dx.doi.org/10.2140/camcos.2018.13.1

ADAPTIVELY WEIGHTED LEAST SQUARES FINITE ELEMENT METHODS FOR PARTIAL DIFFERENTIAL EQUATIONS WITH SINGULARITIES

BRIAN HAYHURST, MASON KELLER, CHRIS RAI,
XIDIAN SUN AND CHAD R. WESTPHAL

The overall effectiveness of finite element methods may be limited by solutions that lack smoothness on a relatively small subset of the domain. In particular, standard least squares finite element methods applied to problems with singular solutions may exhibit slow convergence or, in some cases, may fail to converge. By enhancing the norm used in the least squares functional with weight functions chosen according to a coarse-scale approximation, it is possible to recover near-optimal convergence rates without relying on exotic finite element spaces or specialized meshing strategies. In this paper we describe an adaptive algorithm where appropriate weight functions are generated from a coarse-scale approximate solution. Several numerical tests, both linear and nonlinear, illustrate the robustness of the adaptively weighted approach compared with the analogous standard L^2 least squares finite element approach.

1. Introduction

In this paper we consider partial differential equations that exhibit singular behavior at isolated locations in the domain. It is well known that problems with smooth data may fail to provide smooth solutions as a consequence of either the domain or the operator. To illustrate the main ideas, consider

$$\begin{cases} \mathcal{K}(u) = f & \text{in } \Omega, \\ u = g & \text{on } \partial\Omega, \end{cases} \tag{1}$$

where $\mathcal{K}$ is a second-order differential operator. If $f \in L^2(\Omega)$ and $g \in H^{3/2}(\Omega)$ is sufficient to guarantee that $u \in H^2(\Omega)$, then we consider the problem to have full regularity. We consider problems without this property to have a low regularity, or (potentially) nonsmooth solutions. For example, Poisson's equation is known to have full regularity when Ω is convex, but can have nonsmooth solutions when

This work was supported by the National Science Foundation under grant DMS-1216297.
MSC2010: primary 65N30; secondary 65N12, 35J20, 76D05.
Keywords: adaptive finite element methods, weighted norm minimization, singularities.

$\partial\Omega$ has corners (or edges) with interior angle greater than π [20]. This lack of smoothness is localized, however. In any subdomain excluding a neighborhood of each corner point, the solution remains smooth. Other elliptic problems have similar behavior as a consequence of the domain; see, e.g., [24; 25]. The operator $\mathcal{K}$ can also induce a loss of smoothness when coefficients are either singular (i.e., $\to\infty$) or degenerate (i.e., $\to 0$) at distinct points in Ω [5].

Invariably, numerical methods tend to suffer as a consequence of a loss of regularity. Finite element convergence rates can be reduced, or in some cases, the method can fail to converge to the solution of the problem. Moreover, in many situations, the loss of optimal rates of convergence is effective globally, even though the nonsmooth behavior of the solution is localized. This global effect from a local component is known as *the pollution effect*.

A wide range of computational approaches has been developed to handle the difficulties induced by such singularities and encompass nearly all aspects of the overall numerical framework. In the finite element context, problems where singularities cause slow convergence can often be effectively treated with graded meshes or an adaptive mesh refinement strategy [30; 17; 1; 12]. In more extreme cases, where standard formulations would not yield discretization convergence, specialized finite element spaces can be employed to better match the low regularity inherent in the problem, for example, using Nédélec or Raviart–Thomas elements as in [7; 10].

In cases where the operator kernels are known analytically, an enhanced finite element basis can be constructed to capture singular solutions better than with standard polynomial bases [4; 3; 2; 35; 32]. Further, there are a number of paradigms that are designed around a weak variational formulation that seek solutions in lower-order Sobolev spaces rather than more traditional approaches. In the context of discontinuous Galerkin (DG) and discontinuous Petrov–Galerkin (DPG) [18] methods, for example, continuity requirements in the trial and test spaces are relaxed and additional degrees of freedom on the element boundaries lead to additional jump conditions in the variational problem. Additionally, in the least squares finite element context, for example, dual space norms induced by the operator adjoint can replace standard L^2 norms to relax regularity requirements [9; 14]. The methodology we propose here has parallels to each of these ideas.

In this work we introduce an adaptively weighted least squares finite element approach for problems with singularities. By generalizing the standard least squares functional with weighted norms, we may essentially redistribute the strength by which the variational problem is enforced across the domain. The use of weighted norms and weighted inner products is, of course, not a new idea. Using weighted norms to generalize L^2 residual minimization problems allows for robust treatment of problems with boundary singularities in weighted $H^1(\Omega)$ or $H(\mathrm{div})$ norms [27; 28; 15]. Though this approach is effective, it requires the explicit construction of a weight function localized to each singular point in the domain. Here we use a sequence of coarse-scale approximations to generate a customized weighted norm in which to minimize the error. This adaptive approach can reproduce the effectiveness of the weighted norm least squares approach, but with the advantage of not requiring the a priori knowledge of either the power or location of any singularity. By analogy, this is similar to the advantage of adaptive mesh refinement in allowing approximate solutions to guide the construction of an optimal mesh. In [34], this adaptive approach is used in a weighted Galerkin formulation for problems with boundary layers.

The organization of this paper is as follows. In the next section, we formally introduce the idea of a weighted least squares finite element method. In Section 3 we provide details of an adaptive framework for choosing effective weight functions from a sequence of coarse-scale approximations. In Section 4 we provide several numerical examples that illustrate the robustness of the method.

2. Notation and background

Throughout this paper, Ω and $\partial\Omega$ represent the domain and boundary of the PDE, which has a nonsmooth solution at distinct locations in $\overline{\Omega}$. We use standard notation for the $L^2(\Omega)^d$ norm $\|\cdot\|$ and inner product $\langle\,\cdot\,,\cdot\,\rangle$ and use $\|\cdot\|_D$ to denote the L^2 norm on subdomain $D \subseteq \Omega$.

We consider the least squares finite element approach to problems of the form in (1). Let $LU = F$ be a linear, first-order reformulation of (1). For nonlinear problems, L represents a linearization about a current approximation and the solution procedure would involve a sequence of such linearized problems. In either case, we thus require finding a finite element approximation to U in the function space $\mathcal{V}$. The standard L^2 least squares method here is to define the least squares functional

$$\mathcal{F}(U; F) = \|LU - F\|^2 \tag{2}$$

and minimize over $\mathcal{V}$: find $U \in \mathcal{V}$ such that $\mathcal{F}(U; f) \leq \mathcal{F}(V; f)$ for all $V \in \mathcal{V}$. This minimization problem is equivalent to the variational problem: find $U \in \mathcal{V}$ such that

$$\langle LU, LV\rangle = \langle F, LV\rangle \quad \text{for all } V \in \mathcal{V}.$$

In general, we assume a least squares finite element formulation that is well posed and robust for smooth problems. In many cases these formulations contain additional consistent constraints. The weighting procedure here is designed to extend such a formulation to recover optimal (or nearly optimal) behavior in the presence of nonsmooth solutions.

For the general weighted least squares method, let $w : \overline{\Omega} \to [0, 1]$ denote a weight function (possibly different for each equation), and define the weighted least squares functional

$$\mathcal{F}_w(U;\, F) = \|w(LU - F)\|^2. \tag{3}$$

Similar to the standard approach, minimizing $\mathcal{F}_w$ over $U \in \mathcal{V}$ is equivalent to finding $U \in \mathcal{V}$ such that

$$\langle wLU, wLV\rangle = \langle wF, wLV\rangle \quad \text{for all } V \in \mathcal{V}.$$

The weighted least squares approach has been used effectively for problems with singular behavior, essentially seeking to recover optimal finite element convergence rates away from the singular points and rates similar to the interpolant near singularities. In [27; 28; 15] the weighted least squares approach is developed using weight functions based on the asymptotic behavior of the solutions near singularities. In [5] a similar approach is taken for a problem with singular/degenerate coefficients. Adopting this idea in practice has been effective for other applications (e.g., for incompressible fluids [29; 16]) and provides a flexible and straightforward way to modify a least squares finite element method in the presence of singularities. This approach requires a priori knowledge of the location and an estimate of the asymptotic behavior of each point of nonsmoothness to define an appropriate weight function. In the following section, we develop a general adaptive approach that does not require this a priori information, but rather builds an optimal composite weight function based on a coarse-scale approximate solution that requires no explicit user input.

3. The adaptively weighted least squares approach

Let Ω^h represent a triangulation of the domain and $\mathcal{V}^h$ an associated finite element space in which we will approximate the solution. Given a weight function w, the discrete solution U^h is the unique minimizer of $\mathcal{F}_w(U^h;\, F)$ over $\mathcal{V}^h$: find $U^h \in \mathcal{V}^h$ such that

$$\mathcal{F}_w(U^h;\, F) \le \mathcal{F}_w(V^h;\, F) \quad \text{for all } V^h \in \mathcal{V}^h. \tag{4}$$

The adaptive approach is based on defining w from a current approximation to the exact solution. For this, we define an elementwise measure of the approximation gradient

$$\mathcal{G}(\tau) = \frac{1}{\mu(\tau)}\|\nabla U^h\|_\tau, \tag{5}$$

where $\mathcal{G}_i = \mathcal{G}(\tau_i)$ is the value on τ_i, the i-th element of Ω^h. In cases where the elements are of vastly different scales, we take $\mu(\tau) = h_\tau^2$ as a measure of the area of the element, making $\mathcal{G}(\tau)$ a measure of error density. With quasiuniform meshes, $\mu(\tau) = 1$ can be used. We now define $\mathcal{G}$ as a piecewise constant function on Ω^h.

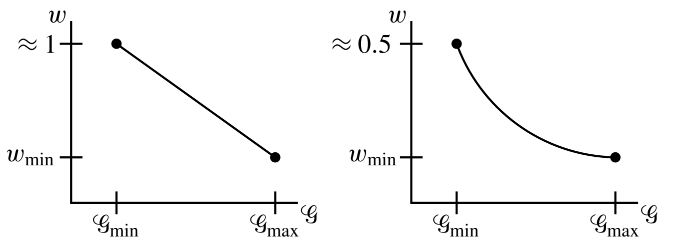


Figure 1. Two shape function options for constructing the weight function. The affine model (left) reflects (6), and the inverse model (right) illustrates (7).

The maximum and minimum values of $\mathcal{G}$ are denoted by

$$\mathcal{G}_{\min} = \min_{\tau_i \in \Omega^h} \mathcal{G}_i \quad \text{and} \quad \mathcal{G}_{\max} = \max_{\tau_i \in \Omega^h} \mathcal{G}_i .$$

Locations with large/small gradients imply that the weight function should be chosen small/large (see, e.g., [27; 28; 15]). By redefining the metric under which the error is minimized in this way, the variational problem is weakened in regions where the solution is most difficult to approximate.

We give two options for constructing w as a piecewise constant function from $\mathcal{G}$:

$$w_i = \frac{\mathcal{G}_{\max} - \mathcal{G}_i}{\mathcal{G}_{\max} - \mathcal{G}_{\min}} + \frac{\mathcal{G}_{\min}}{\mathcal{G}_{\max}}. \tag{6}$$

or

$$w_i = \frac{c}{\mathcal{G}_i + c}, \quad \text{where } c = \frac{\mathcal{G}_{\min}\mathcal{G}_{\max}}{\mathcal{G}_{\max} - \mathcal{G}_{\min}}. \tag{7}$$

In each (6) and (7), $w_i \leq 1$ and $w_i = w_{\min} = \mathcal{G}_{\min}/\mathcal{G}_{\max}$ when $\mathcal{G}_i = \mathcal{G}_{\max}$. Figure 1 illustrates the shape function for each case (affine and inverse) and suggests a range of other empirical options.

In an iterative framework, the basic adaptively weighted least squares method is described in Algorithm 1.

```
**Start**: initially set $w = 1$ uniformly; choose initial mesh $\Omega^h$
**Solve**: obtain initial solution $U^h_{\text{old}}$ by solving (4)
**while** (overall accuracy $<$ goal) {
    **Refine mesh**: (optional) uniformly or adaptively
    **while** (nonlinear error estimate $>$ tolerance) {
        **Relinearize**: about $U^h_{\text{old}}$ (for nonlinear problems)
        **Construct weight**: use $U^h_{\text{old}}$ to define $G_i$ from (5) and $w_i$ from (6) or (7)
        **Resolve**: using $w$, find $U^h$ by solving (4); set $U^h_{\text{old}} \leftarrow U^h$
    }
}
```

Algorithm 1. Adaptively weighted least squares framework.

The framework here is quite flexible and may be thought of analogously to the idea of adaptive mesh refinement, where a sequence of increasingly accurate approximations is found by successively redefining the weight function and resolving a finer-scale and higher-resolution problem. The mesh refinement step allows the weight function to be developed through coarse-scale approximations which are relatively computationally inexpensive. Stopping criteria for the algorithm can be based on a single metric, like the global value of the least squares functional (3) or by the total number of refinement levels desired. For nonlinear problems, an indicator of how well the nonlinear error is resolved can involve a measure of the change between iterates or a comparison between linear and nonlinear functionals. It is also possible to simply take a fixed number of linearization steps on each mesh level, refining the weight function at each opportunity.

In [27; 28; 15], several weighted norm least squares methods are designed around minimizing the approximation error in weighted Sobolev spaces, where the weight functions are chosen according to the asymptotic nature of the solution. For example, in [27], assume $U \sim r^{\alpha-1}$ represents the asymptotic behavior of the solution to $LU = F$ near a boundary singularity, where r is the distance to the singular point and $\alpha \in (0, 1)$ represents the power of the singular solution. A simple calculation indicates that $U \in H^s(\Omega)$ for $s < \alpha \in (0, 1)$. The a priori weight function described in [27] requires choosing $w \sim r^\beta$ such that $wU \in H^2(\Omega)$, which indicates $\beta \gtrsim 2 - \alpha$. With a weight function of this design, it is proved that optimal finite element error convergence in a weighted Sobolev space is expected. This indicates that the pollution effect is eliminated, yielding the same convergence as the L^2 interpolant in a neighborhood of the singular point and optimal convergence in a neighborhood excluding the singularity. For the adaptive approach, we mimic this by choosing the weight construction in (7), where we see that asymptotically

$$w \sim \frac{1}{|\nabla U|} \sim r^{2-\alpha},$$

which matches the a priori construction described in [27]. Analysis of the weighted least squares methods in [27; 28; 15] is done in the context of a hierarchy of Sobolev spaces weighted by powers of r, whereas here we have a set of spaces weighted by an evolving approximate solution.

In the following section we present several numerical tests that illustrate the utility of the adaptively weighted approach described here. The first two examples have a known analytic solution, and the convergence, both near the singularity and away from it, is carefully monitored to show how the adaptive approach improves convergence. The remaining examples provide a variety of other measures to illustrate the effectiveness and flexibility of the adaptive approach.

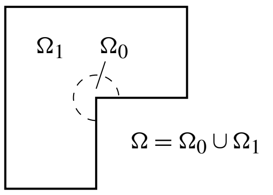


Figure 2. L-shaped domain for Example 1: Ω is partitioned into subdomains Ω_0 and Ω_1 to distinguish global convergence from local convergence near the singular point.

4. Numerical results

In this section we provide several numerical examples to illustrate the effectiveness and robustness of the adaptively weighted least squares approach as described in Algorithm 1. In the first example, we consider a div/curl first-order system induced by the Laplace operator. In this context, regularity dictates that the standard least squares approach using H^1 conforming elements is not applicable for nonconvex domains. Weighted least squares methods can be used to recover optimal convergence in a weighted H^1 norm (see, e.g., [27; 28]), and the results here show that the adaptively weighted approach achieves similar results, but does so with no explicit a priori information provided by the user. The second example applies the adaptively weighted approach to a singularly perturbed elliptic operator that induces a nonsmooth solution at an interior point in the domain. Here, a mixed least squares finite element formulation is examined and the adaptively weighted approach increases slow convergence induced by the loss of smoothness in the solution. In the next example, we consider a div/curl least squares formulation of the incompressible Stokes equations in a nonconvex domain. We show how the adaptively weighted approach ameliorates the pollution effect, yields optimal convergence in the weighted least squares functional norm, and gives asymptotically accurate approximations to the velocity in the neighborhood of a reentrant corner. In addition we show that the adaptively weighted approach improves mass conservation in the example. The next two examples illustrate the algorithm in the framework of a nonlinear problem. In these cases we consider two different formulations of the stationary Navier–Stokes equations applied to standard benchmark problems (the lid-driven cavity and flow over a square obstacle).

All computational results are implemented in FreeFem++ [21].

Example 1 (Poisson on the L-shaped domain). For this example we define $\Omega = \{(x, y) \in (-1, 1)^2 : (x, y) \notin [0, 1) \times (-1, 0]\}$, the L-shaped domain pictured in Figure 2. We also define a partition of the domain to distinguish between a neighborhood of the singular point and the rest of the domain: $\Omega_0 = \{(x, y) \in \Omega : x^2 + y^2 < (0.25)^2\}$ denotes the neighborhood of the origin and $\Omega_1 = \Omega \setminus \Omega_0$ represents the remainder of the domain in which the solution is smooth.

Standard LS ($w = 1$)

N	$\mathcal{F}^{1/2}$	$\|p^* - p^h\|_{\Omega_0}$	$\|p^* - p^h\|_{\Omega_1}$	$\|\boldsymbol{u}^* - \boldsymbol{u}^h\|_{\Omega_0}$	$\|\boldsymbol{u}^* - \boldsymbol{u}^h\|_{\Omega_1}$
1716	1.22	0.0166	0.0454	0.389	0.448
6898	1.21	0.0157	0.0439	0.382	0.439
27742	1.20	0.0152	0.0431	0.377	0.434
rate $\approx$	0	0	0	0	0

Adaptively weighted LS

N	$\mathcal{F}_w^{1/2}$	$\|p^* - p^h\|_{\Omega_0}$	$\|p^* - p^h\|_{\Omega_1}$	$\|\boldsymbol{u}^* - \boldsymbol{u}^h\|_{\Omega_0}$	$\|\boldsymbol{u}^* - \boldsymbol{u}^h\|_{\Omega_1}$
1716	0.136	0.00140	0.000595	0.1427	0.0470
6898	0.0755	0.000313	0.000132	0.0855	0.0151
27742	0.0407	0.000104	0.0000412	0.0524	0.00441
rate $\approx$	0.89	1.58	1.68	0.71	1.78
optimal rate	1	$1.6\bar{6}$	2	$0.6\bar{6}$	2

Table 1. Convergence comparison between the standard least squares approximation ($w = 1$) and the adaptively weighted approach. Convergence rate is estimated from results on the two finest levels, and the optimal rate is based on standard interpolation bounds for the exact solution.

We consider numerically approximating a nonsmooth solution to the problem

$$\begin{cases} \Delta p = f & \text{in } \Omega, \\ p = p^* & \text{on } \partial\Omega, \end{cases} \tag{8}$$

where we take $f = 0$, and the boundary data is chosen so that the exact solution corresponds to $p^* = r^{2/3} \sin(2\theta/3)$ and (r, θ) corresponds to a local polar coordinate system centered at the origin. The exact solution here is in the kernel of the Laplacian and represents the nonsmooth component of a typical Poisson problem on a domain with a reentrant corner of interior angle $3\pi/2$.

We introduce the flux variable $\boldsymbol{u} = \nabla p$ and consider the expanded first-order system

$$\begin{cases} \nabla \cdot \boldsymbol{u} = f & \text{in } \Omega, \\ \nabla \times \boldsymbol{u} = 0 & \text{in } \Omega, \\ \boldsymbol{u} - \nabla p = \boldsymbol{0} & \text{in } \Omega, \\ \hat{\boldsymbol{\tau}} \cdot \boldsymbol{u} = \hat{\boldsymbol{\tau}} \cdot \nabla p^* & \text{on } \partial\Omega, \\ p = p^* & \text{on } \partial\Omega, \end{cases} \tag{9}$$

where $\hat{\boldsymbol{\tau}}$ is the counterclockwise unit tangent vector to $\partial\Omega$. The boundary condition on $\boldsymbol{u}$ is found by differentiating the boundary data on p^*, and though this equation is redundant, including it generally improves the quality of approximations on coarse meshes. In this example, boundary conditions on $\boldsymbol{u}$ and p are imposed strongly, though there are a range of boundary condition treatments possible in the least

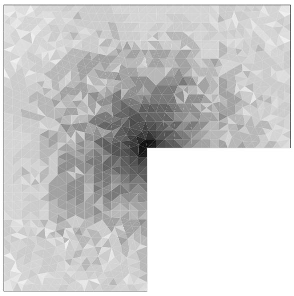


Figure 3. Adaptively generated weight function for domain with $N = 1716$ elements. Larger values are lighter ($w_{\max} = 0.507$); smaller values are darker ($w_{\min} = 0.0281$).

squares context. The associated weighted least squares functional is

$$\mathcal{F}_w(\boldsymbol{u}, p; f) = \|w(\nabla \cdot \boldsymbol{u} - f)\|^2 + \|w\nabla \times \boldsymbol{u}\|^2 + \|w(\boldsymbol{u} - \nabla p)\|^2, \qquad (10)$$

which we minimize over standard continuous P1 elements for each unknown, enforcing boundary conditions on p and $\boldsymbol{u}$ strongly. We follow Algorithm 1 for the iterative approach, and for this problem (5) takes the form

$$\mathcal{G}(\tau) = (\|\nabla p^h\|_\tau^2 + \|\nabla \boldsymbol{u}^h\|_\tau^2)^{1/2}$$

on each element τ_i. The piecewise constant weight function in each step is computed according to (7).

In Table 1 convergence is summarized for the adaptively weighted approach as well as the standard approach (corresponding to $w = 1$). Since the exact solution is known, we report the L^2 error in both p and $\boldsymbol{u}$ and in both Ω_0 and Ω_1. In each case, a quasiuniform mesh is used with N total elements, and for the adaptive approach we take three iterations on each mesh and report the values at the third iteration.

A simple calculation reveals that $p^* \in H^{1+s}(\Omega)$ and $\boldsymbol{u}^* = \nabla p^* \in H^s(\Omega)^2$ for $s < \frac{2}{3}$. Since $\boldsymbol{u}^* \notin H^1(\Omega)$, convergence is not guaranteed for the standard LS approach, and it fails as expected. The adaptive approach performs better, showing near-optimal convergence rates for the L^2 error for both p and $\boldsymbol{u}$ in each subdomain. The least squares functional norm, which is essentially a weighted H^1 seminorm, converges at the optimal rate. This shows that we can retain the convenience of using H^1 conforming finite element spaces, even when regularity indicates that the solution is not in $H^1(\Omega)$ locally.

To illustrate the character of the weight function generated by this approach, Figure 3 shows the weight generated on the coarsest mesh for the results in Table 1. Smaller values of w (in darker color) occur near the reentrant corner. For context, the example in Table 1 produces weight with $\|w^h\|_{L^2(\Omega)} \approx 0.876$, which in absolute magnitude does not substantially differ from the scale under uniform weighting, which gives $\|1\|_{L^2(\Omega)} = \sqrt{3} \approx 1.73$ for this example.

In this example the domain had one singular point, but applying the method to a problem with multiple singularities is analogous and straightforward.

Example 2 (a singularly perturbed elliptic problem). For this example we treat a problem with a singularity in the interior of the domain, induced by the operator rather than the geometry of the boundary. Consider the problem

$$\begin{cases} -\nabla\cdot(r^{2\beta}\nabla u)+r^{2\alpha}u=f & \text{in } \Omega, \\ u=0 & \text{on } \partial\Omega, \end{cases} \tag{11}$$

where $\Omega=(-1,1)^2$ and r is the coordinate distance from $(0,0)$. When the coefficients are degenerate (i.e., go to zero) or singular (i.e., blow up) at an interior point, as is possible here, the solution may be nonsmooth in a neighborhood of the origin. In [5], a weighted norm least squares finite element method is developed for (11) where the weight function is chosen by the expected regularity of the problem. For this example, we choose $\beta=0.5$ and $\alpha=-0.5$, which induces a solution with asymptotic behavior of r^{λ} for $\lambda\approx 0.618034$. The function f is chosen so that the exact solution is given by

$$u=(1-x^2)(1-y^2)r^{\lambda},$$

which exhibits the expected nonsmooth behavior at the origin, but satisfies homogenous Dirichlet boundary conditions. For this example we recall the weighted least squares approach in [5], but apply the adaptive approach in choosing the weight function.

Let $\boldsymbol{\sigma}=-r^{2\beta}\nabla u$, and define the weighted least squares functional

$$\mathscr{F}_w(u,\boldsymbol{\sigma};f)=\|w(\nabla\cdot\boldsymbol{\sigma}+r^{2\alpha}u-f)\|^2+\|w(\boldsymbol{\sigma}+r^{2\beta}\nabla u)\|^2.$$

We use a uniform triangulation of Ω and approximate $\boldsymbol{\sigma}$ in the lowest-order $H(\text{div})$ conforming Raviart–Thomas finite element space, RT0, and use conforming P1 elements for u, with boundary conditions on u enforced strongly. As in Example 1, we define a partition of Ω, where $\Omega_0=(-0.2,0.2)^2$ represents a fixed neighborhood of the origin and $\Omega_1=\Omega\setminus\Omega_0$ is the remainder of the domain.

We follow Algorithm 1 for the iterative approach and use

$$\mathscr{G}(\tau)=(\|\nabla u\|_{\tau}^2+\|\nabla\boldsymbol{\sigma}\|_{\tau}^2)^{1/2}$$

as the elementwise gradient measure and use (7) for the construction of w from $\mathscr{G}$. Table 2 summarizes numerical results on four nested mesh levels. The standard least squares approach shows results typical of a problem with reduced regularity. Even though the functional norm decreases at approximately $\mathbb{O}(h)$, the L^2 error of u shows slow convergence, even in the subdomain away from the origin. The adaptively weighted approach yields similarly slowly decreasing errors near the origin, but faster convergence in the rest of the domain.

	Standard LS ($w=1$)			Adaptively weighted LS		
N	$\mathscr{F}^{1/2}$	$\lVert u-u^h\rVert_{\Omega_0}$	$\lVert u-u^h\rVert_{\Omega_1}$	$\mathscr{F}_w^{1/2}$	$\lVert u-u^h\rVert_{\Omega_0}$	$\lVert u-u^h\rVert_{\Omega_1}$
512	0.591	0.0154	0.0133	0.591	0.0154	0.0133
2048	0.299	0.00817	0.00270	0.0707	0.00668	0.00704
8192	0.150	0.00406	0.00151	0.0336	0.00314	0.00175
32768	0.0756	0.00222	0.000874	0.0148	0.00183	0.000512
rate $\approx$	0.99	0.87	0.79	1.18	0.78	1.77

Table 2. Numerical results for Example 2. Convergence rates are computed relative to the two finest mesh levels.

For the formulation used for this problem, it's important to recognize the challenge here is somewhat different from the previous example. In Example 1, the flux variable fails to be in $H^1(\Omega)$ in a neighborhood of the corner point, but we still use a finite element subspace of H^1 for its approximation. Thus, the standard approach cannot be expected to converge. Here we have $u \in H^1(\Omega)$ and $\boldsymbol{\sigma} \in H(\mathrm{div}) = \{\boldsymbol{v} \in L^2(\Omega)^2 : \nabla\cdot\boldsymbol{v} \in L^2(\Omega)\}$, which is consistent with the approximating spaces, though not smooth enough to achieve optimal L^2 rates. The standard approach converges, albeit slowly, and the adaptively weighted approach serves to weaken the problem enough near the origin to enhance the convergence away from the origin, i.e., mitigating the pollution effect.

Example 3 (Stokes flow). For this example we consider steady incompressible flow in $\Omega \subset \mathbb{R}^2$ modeled by Stokes' equations

$$\begin{cases} -\Delta\boldsymbol{u} + \nabla p = 0 & \text{in } \Omega, \\ \nabla\cdot\boldsymbol{u} = 0 & \text{in } \Omega, \\ \boldsymbol{u} = \boldsymbol{g} & \text{on } \partial\Omega, \end{cases} \tag{12}$$

where $\boldsymbol{u} = (u_1, u_2)$ represents fluid velocity, p is the pressure, and $\boldsymbol{g}$ gives the velocity on the boundary $\partial\Omega$. Figure 4 describes the domain and boundary conditions for this example. By introducing the velocity gradient $\boldsymbol{U} = \nabla\boldsymbol{u}$, system (12) can be reformulated to the first-order system

$$\begin{cases} -\nabla\cdot\boldsymbol{U} + \nabla p = \boldsymbol{0} & \text{in } \Omega, \\ \nabla\times\boldsymbol{U} = \boldsymbol{0} & \text{in } \Omega, \\ \boldsymbol{U} - \nabla\boldsymbol{u} = \boldsymbol{0} & \text{in } \Omega, \\ \nabla\cdot\boldsymbol{u} = 0 & \text{in } \Omega, \\ \boldsymbol{u} = \boldsymbol{g} & \text{on } \partial\Omega, \\ \hat{\boldsymbol{\tau}}\cdot\boldsymbol{U} = \hat{\boldsymbol{\tau}}\cdot\nabla\boldsymbol{g} & \text{on } \partial\Omega, \end{cases} \tag{13}$$

where $\hat{\boldsymbol{\tau}}$ is a unit tangent vector to $\partial\Omega$. Including the curl constraint of $\boldsymbol{U}$ into the system is an additional, yet consistent, constraint from the definition of $\boldsymbol{U}$. Additionally we note that $U_{11} + U_{22} = \nabla\cdot\boldsymbol{u} = 0$, and we directly substitute $U_{22} = U_{11}$ in (13), reducing the total unknowns by one.

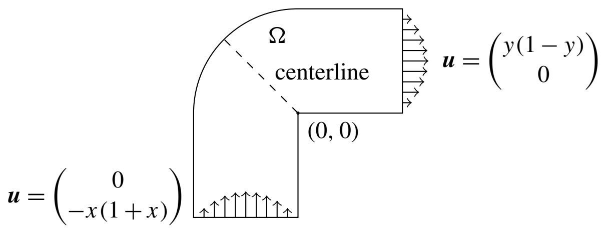

Ω
centerline
$(0,0)$
$\boldsymbol{u} = \begin{pmatrix} y(1-y) \\ 0 \end{pmatrix}$
$\boldsymbol{u} = \begin{pmatrix} 0 \\ -x(1+x) \end{pmatrix}$

Figure 4. Stokes flow domain and boundary conditions. Inflow and outflow boundary values are shown; no-slip conditions apply to other walls. Conservation of mass is measured as the velocity flux across the diagonal line along $y = -x$.

The standard div/curl least squares approach is to minimize the functional

$$\mathcal{F}(\boldsymbol{u}, \boldsymbol{U}, p) = \|\nabla \cdot \boldsymbol{U} - \nabla p\|^2 + \|\nabla \times \boldsymbol{U}\|^2 + \|\boldsymbol{U} - \nabla \boldsymbol{u}\|^2 + \|\nabla \cdot \boldsymbol{u}\|^2$$

over an appropriate space of functions for each unknown. When Ω is sufficiently smooth and convex, the norm induced by $\mathcal{F}$ is equivalent to the $H^1(\Omega)$ norm of each unknown (up to a constant for p) and accurate discrete approximations can be found using standard conforming piecewise polynomial spaces for each unknown. For nonconvex domains, $\boldsymbol{U}$ cannot be guaranteed to remain in $H^1(\Omega)$ and the H^1 equivalence of LS functional norm breaks down. This well known loss of regularity has severe consequences for the standard div/curl LS approach — similar to Examples 1 and 2, singularities at nonconvex corners can cause a loss of convergence and inaccurate solutions globally. System (13) is certainly not the only first-order formulation of (12), and the literature in least squares finite elements reflects a wide range of choices with different advantages and disadvantages (see, e.g., [6; 22; 13]). The div/curl approach does not require exotic finite element spaces, it admits realistic boundary conditions for $\boldsymbol{U}$, and it tends to yield linear systems that can be solved robustly by multigrid methods. However, this system exhibits a loss of regularity (see, e.g., [23; 11; 26]), which is what makes the weighted norm approach a compelling way to deal with problems with singularities.

For the adaptively weighted least squares approach, we directly follow the procedure defined in Algorithm 1, defining the weighted least squares functional by

$$\mathcal{F}_w(\boldsymbol{u}, \boldsymbol{U}, p) = \|w(\nabla \cdot \boldsymbol{U} - \nabla p)\|^2 + \|w\nabla \times \boldsymbol{U}\|^2 + \|w(\boldsymbol{U} - \nabla \boldsymbol{u})\|^2 + \|w\nabla \cdot \boldsymbol{u}\|^2,$$

where w is chosen from a previous approximation according to elementwise values of

$$\mathcal{G}(\tau) = \|\nabla \boldsymbol{u}^h\|_\tau^2 + \|\nabla \boldsymbol{U}^h\|_\tau^2 + \|\nabla p^h\|_\tau^2,$$

and w is constructed according to (7). All unknowns are approximated with continuous P2 elements. We follow the nested iteration approach, where the initial approximation is computed on a coarse quasiuniform mesh, a weight function is generated on this mesh (see Figure 5), then the mesh is refined uniformly by

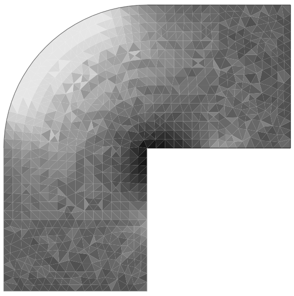


Figure 5. Adaptively generated weight function for Stokes flow example problem. Shown in grayscale is the first adaptive weight function, based on the initial approximation on mesh with $h^{-1} = 16$. (Larger values are lighter; smaller values are darker.)

splitting each element into four elements, and the next iterate is computed on the refined mesh. This is then repeated for a total of four refinement levels.

Since no exact solution is available for this problem, we consider several metrics of convergence. First is the least squares functional norm $\mathcal{F}_w(\boldsymbol{u}^h, \boldsymbol{U}^h, p^h)^{1/2}$, which includes the weight function used in finding the approximate solution. The second metric we use is the unweighted residual norm evaluated on a subdomain that excludes a neighborhood of the singularity:

$$\mathcal{R}^{1/2} = \left(\|\nabla\cdot\boldsymbol{U}^h - \nabla p^h\|^2_{\Omega_1} + \|\nabla\times\boldsymbol{U}^h\|^2_{\Omega_1} + \|\boldsymbol{U}^h - \nabla\boldsymbol{u}^h\|^2_{\Omega_1} + \|\nabla\cdot\boldsymbol{u}^h\|^2_{\Omega_1}\right)^{1/2},$$

where $\Omega_1 = \{(r,\theta)\in\Omega : r > 0.1\}$. Figure 6 shows a comparison of convergence between the standard least squares approach and the adaptively weighted approach. As in Examples 1 and 2, the standard approach stalls, while the adaptive approach converges at nearly optimal rates. Strong convergence in both the functional and

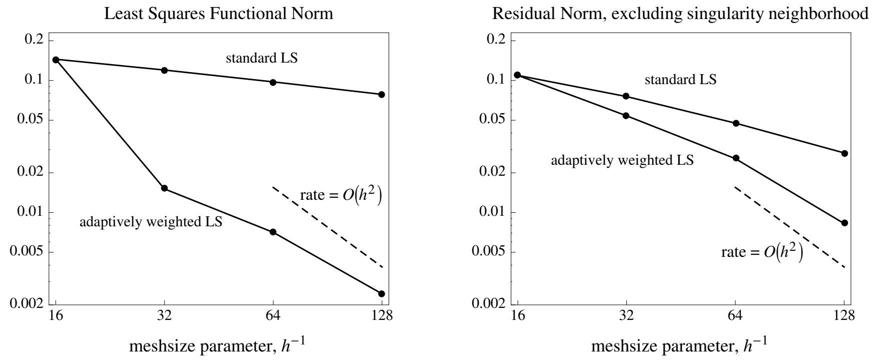


Figure 6. Convergence comparison between standard least squares solution versus the adaptively weighted approach for increasing refinement level. The left shows the least squares functional norm $\mathcal{F}^{1/2}$ for the standard approach and the weighted functional norm $\mathcal{F}_w^{1/2}$. The right shows the L^2 residual norm in a subdomain excluding a neighborhood of the singularity.

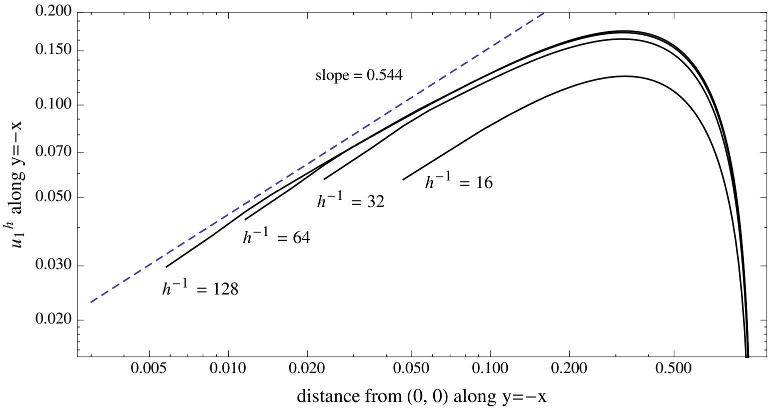


Figure 7. Trace of u_1^h along the line $y = -x$ through Ω for increasing refinement level. The log-log scale shows the asymptotic behavior $u_1^h \sim r^{0.544}$.

residual norms shows that no significant pollution effect is present in the weighted norm approximations.

To examine the quality of the solution near the singularity, we consider the velocity approximates near the reentrant corner. Through asymptotic analysis, it can be shown that $\boldsymbol{u} \sim r^{0.544}$ near the origin for this problem. Figure 7 gives a log-log plot of the trace of u_1 along the line $y = -x$ in Ω, which matches the asymptotic rate well, giving confidence that the method is reproducing a locally accurate solution.

As a final consideration, we measure the mass flux along the centerline of the domain (see Figure 4) relative to the inflow. Least squares finite element methods typically enforce conservation of mass by minimizing the least squares functional which includes $\nabla \cdot \boldsymbol{u} = 0$ as one term. Thus, the error in this term is balanced with the other equations in the system, giving conservation of mass errors on the order of the total discretization error. Rebalancing terms in the functional can improve approximation accuracy in one term at the expense of the others, and it is common to rescale the mass term by a large constant to reduce mass loss. We note that this

% of mass loss at center line

h^{-1}	Adaptive	Standard LS
16	30.6%	30.6%
32	7.04%	21.3%
64	2.05%	14.2%
128	0.694%	9.17%

Table 3. Mass loss at center line of symmetry for the adaptively weighted approach versus the standard approach. Both approaches used the same sequence of triangulations of Ω with mesh size parameter h.

should be done with caution, since introducing a large constant will result in lower accuracy in other terms and can degrade the conditioning of the resulting linear system drastically. In Table 3 we show the relative mass loss in the adaptively weighted approach versus the standard least squares approach, showing a significant improvement and further evidence that the weighted approach eliminates pollution effects induced by the singularity at the corner.

Example 4 (Navier–Stokes, lid-driven cavity). Here we consider the div/curl formulation of steady incompressible flow in $\Omega \subset \mathbb{R}^2$ as modeled by the first-order system

$$\begin{cases} -\nabla \cdot \boldsymbol{U} + \mathrm{Re}\boldsymbol{U}\boldsymbol{u} + \nabla p = \boldsymbol{0} & \text{in } \Omega, \\ \boldsymbol{U} - \nabla \boldsymbol{u} = \boldsymbol{0} & \text{in } \Omega, \\ \nabla \cdot \boldsymbol{u} = 0 & \text{in } \Omega, \\ \nabla \times \boldsymbol{U} = \boldsymbol{0} & \text{in } \Omega, \\ \nabla(\mathrm{tr}(\boldsymbol{U})) = \boldsymbol{0} & \text{in } \Omega, \\ \boldsymbol{u} = \boldsymbol{g} & \text{on } \partial\Omega, \\ \hat{\boldsymbol{\tau}} \cdot \boldsymbol{U} = \hat{\boldsymbol{\tau}} \cdot \nabla g & \text{on } \partial\Omega, \end{cases} \tag{14}$$

where $\mathrm{tr}(\boldsymbol{U})$ is the trace of $\boldsymbol{U}$, $\hat{\boldsymbol{\tau}}$ is a unit tangent vector to $\partial\Omega$, and Re is the dimensionless Reynolds number, defined to be the ratio of inertial forces to viscous forces. At low Re, flow is essentially laminar; as Re increases, flow becomes more turbulent. The nonlinearity induced by the $\mathrm{Re}\boldsymbol{U}\boldsymbol{u}$ term makes (14) a natural candidate for the adaptive weighting procedure since iteration will already be necessary to resolve the nonlinearity. We present results for Stokes flow (Re $= 0$) and turbulent flow at Re $= 100$ in the lid-driven cavity (LDC) domain shown in Figure 8. Despite the nonphysical nature of the problem, lid-driven cavity flow remains a well studied standard test problem for fluid dynamics codes. Our standard for accuracy is the data presented in [8].

The discontinuous boundary conditions on $\boldsymbol{u}$ in LDC flow induce strong singularities in p and in some components of $\boldsymbol{U}$ which exclude them from $L^2(\Omega)$ in the neighborhood of the two upper corners (see [19] for details). This poses a different, seemingly more extreme regularity issue than those induced by nonconvex domains. While this loss of smoothness would seem to preclude the use of $H^1(\Omega)$ conforming elements, we recall that each unknown is sufficiently smooth

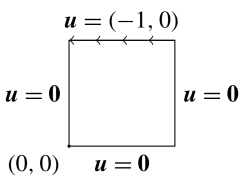

$\boldsymbol{u} = (-1, 0)$
$\boldsymbol{u} = \boldsymbol{0}$
$\boldsymbol{u} = \boldsymbol{0}$
$(0, 0)$
$\boldsymbol{u} = \boldsymbol{0}$

Figure 8. Domain and boundary conditions for the lid-driven cavity test problem.

in any subdomain excluding the upper corners and that the use of an appropriately weighted least squares functional can remove any pollution effect due to the loss of smoothness in the upper corners. Thus, for this problem, we approximate all unknowns using $H^1(\Omega)$ conforming P2 elements (piecewise continuous quadratics), and Ω is discretized using a uniform triangulation. We seek a solution method that converges robustly to the solution away from the singularities.

We implement Newton's method within a nonlinear iteration, defining $\boldsymbol{u}_{\text{old}}$ and $\boldsymbol{U}_{\text{old}}$ as current approximations of $\boldsymbol{u}$ and $\boldsymbol{U}$. The nonlinear inertial term is thus replaced according to $\text{Re}\boldsymbol{U}\boldsymbol{u} \to \text{Re}(\boldsymbol{U}_{\text{old}}\boldsymbol{u} + \boldsymbol{U}\boldsymbol{u}_{\text{old}} - \boldsymbol{U}_{\text{old}}\boldsymbol{u}_{\text{old}})$. When the initial approximations are taken as $\boldsymbol{U}_{\text{old}} = \boldsymbol{0}$ and $\boldsymbol{u}_{\text{old}} = \boldsymbol{0}$, the first Newton step corresponds to a Stokes solve. Since the computation of the weight function is essentially free relative to the PDE solve, we choose to compute a new weight function during each subsequent Newton step according to Algorithm 1. We find that for $\text{Re} = 100$ using a fixed number ($n = 5$) of Newton steps is sufficient to resolve the nonlinearity.

The procedure for each nonlinear step is to minimize the weighted functional

$$\begin{aligned}\mathcal{F}_w(\boldsymbol{u}, \boldsymbol{U}, p; \boldsymbol{u}_{\text{old}}, \boldsymbol{U}_{\text{old}}) = \|w(-\nabla\cdot\boldsymbol{U} + \text{Re}(\boldsymbol{U}_{\text{old}}\boldsymbol{u} + \boldsymbol{U}\boldsymbol{u}_{\text{old}} - \boldsymbol{U}_{\text{old}}\boldsymbol{u}_{\text{old}}) + \nabla p)\|^2 \\ + \|w(\boldsymbol{U} - \nabla\boldsymbol{u})\|^2 + \|w\nabla\cdot\boldsymbol{u}\|^2 + \|w\nabla\times\boldsymbol{U}\|^2 + \|w\nabla(\text{tr}(\boldsymbol{U}))\|^2,\end{aligned}$$

where the weight function is computed according to Algorithm 1 and (7) with elementwise gradient values

$$\mathcal{G}(\tau) = (\|\nabla\boldsymbol{u}\|_\tau^2 + \|\nabla\boldsymbol{U}\|_\tau^2 + \|\nabla p\|_\tau^2)^{1/2}.$$

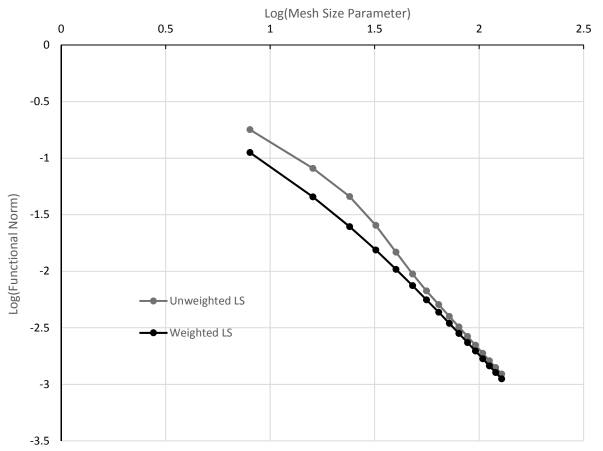
We first show convergence in the following unweighted residual norm on a subdomain that excludes the singularities: $\Omega_1 = \{(x, y) \in \Omega : y \leq 0.75\}$ and

$$\begin{aligned}\mathcal{R}^{1/2} = \big(\|-\nabla\cdot\boldsymbol{U} + \text{Re}\boldsymbol{U}\boldsymbol{u} + \nabla p\|_{\Omega_1}^2 + \|\boldsymbol{U} - \nabla\boldsymbol{u}\|_{\Omega_1}^2 \\ + \|\nabla\cdot\boldsymbol{u}\|_{\Omega_1}^2 + \|\nabla\times\boldsymbol{U}\|_{\Omega_1}^2 + \|\nabla(\text{tr}(\boldsymbol{U}))\|_{\Omega_1}^2\big)^{1/2}.\end{aligned}$$

Figure 9 shows convergence of $\mathcal{R}^{1/2}$ versus the size of each element. For Stokes flow, the adaptively weighted and standard least squares approaches are comparable, but for Navier–Stokes flow at $\text{Re} = 100$, the adaptively weighted approach shows improved error reduction at all resolutions and a nearly optimal $\mathcal{O}(h^2)$ rate.

To further confirm that the adaptively weighted method converges to the exact solution we compare results with benchmark solutions for Stokes flow in [19] and for Navier–Stokes flow in [8].

Figure 10 shows plots of the maximum value of the stream function (left) and the value of the vorticity at $(0, 0.95)$ at increasing resolution (see [19] for a description of these physical quantities). As the mesh is refined we see that the adaptively weighted approach and standard approaches both reproduce the benchmark values asymptotically.



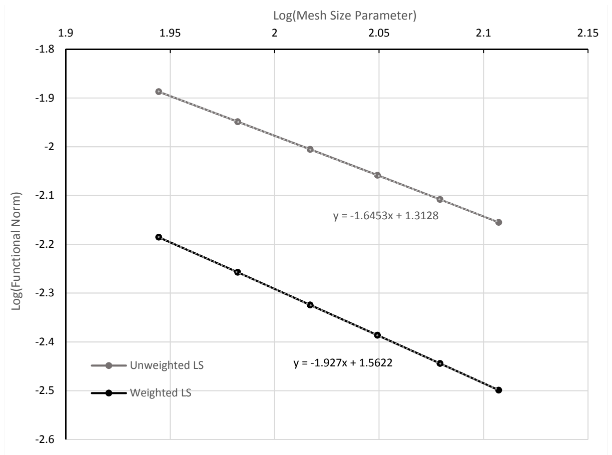
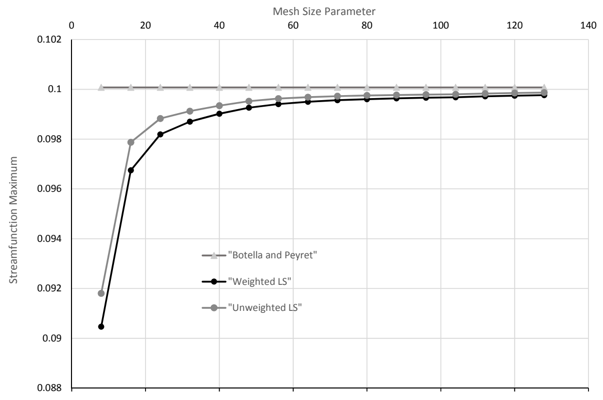


Figure 9. Unweighted residual norm $\mathcal{R}^{1/2}$ versus mesh size parameter h for Stokes flow (Re $= 0$, top) and Navier–Stokes (Re $= 100$, bottom).

Figure 11 shows components of the velocity along horizontal (left) and vertical (right) lines through the center of the domain, compared with a benchmark solution for Re $= 100$ in [8]. The adaptively weighted approach seems to reproduce the benchmark solution well, even though the problem has severe regularity issues from a discontinuous boundary condition.

Example 5 (Navier–Stokes, flow over a square obstacle). In this section, we analyze the steady state flow around a square obstacle using a stress, velocity, pressure formulation of the Navier–Stokes equations:

$$\begin{cases} \nabla\cdot\boldsymbol{\sigma} - \rho\boldsymbol{u}\cdot\nabla\boldsymbol{u} = \mathbf{0}, \\ \boldsymbol{\sigma} = \mu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T) - pI, \\ \nabla\cdot\boldsymbol{u} = 0, \end{cases} \tag{15}$$



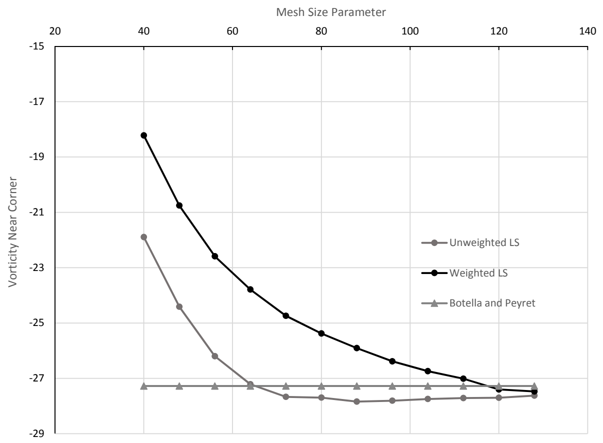
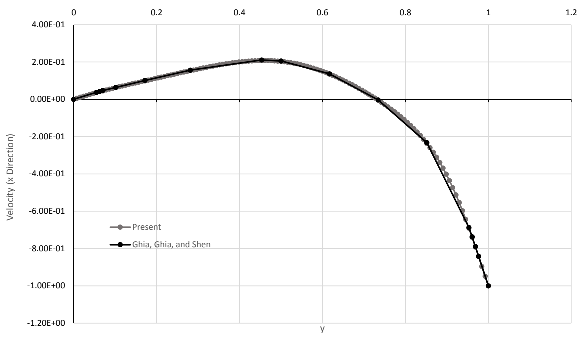


Figure 10. Stream function extrema over various mesh sizes (top) and vorticity values near corner (bottom).

where $\boldsymbol{\sigma} = \left(\begin{smallmatrix}\sigma_{11} & \sigma_{12}\\ \sigma_{21} & \sigma_{22}\end{smallmatrix}\right)$ is the total stress tensor, ρ is the density, $\boldsymbol{u} = (u_1, u_2)$ is the velocity, $\mu = 1$ is the kinematic viscosity, p is the pressure, and I is the 2×2 identity tensor. We define the Reynolds number to be $\mathrm{Re} = \rho v d/\mu$ where $v = 1$ is the characteristic velocity and $d = 1$ is the characteristic length. Thus, ρ is chosen to correspond to the Reynolds number.

Figure 12 shows the domain and boundary conditions used for this test. The full domain is 200 units long and 100 units high with a 1×1 square located in the center. Because the solution is symmetric across $y = 0$, a half domain is used for computation. The north and west edges of the domain have boundary conditions of $u_1 = 1$ and $u_2 = 0$. No-slip boundary conditions are employed on the inner square. The symmetry line along the south edge has boundary conditions setting the y derivatives of u_1 and p to be zero. The sheer stresses, σ_{12} and σ_{21}, along with u_2 are also set to zero along the south edge. The east edge is set to be consistent



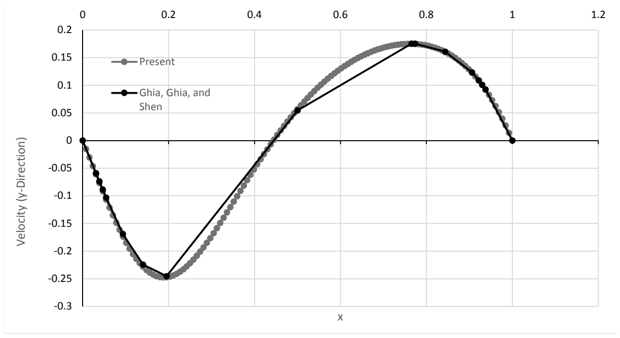
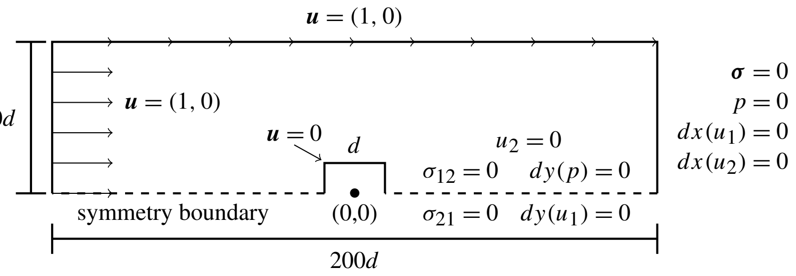


Figure 11. Horizontal (top) and vertical (bottom) velocity profiles through a horizontal center line for Navier–Stokes flow at Re = 100 compared with a benchmark solution.



$\boldsymbol{u} = (1, 0)$
$\boldsymbol{u} = (1, 0)$
50d
$\boldsymbol{u} = 0$
d
$u_2 = 0$
$\sigma_{12} = 0$ $dy(p) = 0$
$\sigma_{21} = 0$ $dy(u_1) = 0$
symmetry boundary
(0,0)
200d
$\boldsymbol{\sigma} = 0$
$p = 0$
$dx(u_1) = 0$
$dx(u_2) = 0$

Figure 12. Domain and boundary conditions for Stokes flow example.

with a fully developed constant flow. It employs a zero normal velocity gradient and zero pressure, which implies each component of $\boldsymbol{\sigma}$ will be zero as well.

Letting $\boldsymbol{u}_{\text{old}}$ represent a current approximation (initially starting with $\boldsymbol{u}_{\text{old}} = \mathbf{0}$), the linearized, weighted least squares functional is given by

$$\mathcal{F}_w(\boldsymbol{\sigma}, \boldsymbol{u}, p; \boldsymbol{u}_{\text{old}}) = \|w(\nabla \cdot \boldsymbol{\sigma} - \rho(\boldsymbol{u}_{\text{old}} \cdot \nabla \boldsymbol{u} + \boldsymbol{u} \cdot \nabla \boldsymbol{u}_{\text{old}} - \boldsymbol{u}_{\text{old}} \cdot \nabla \boldsymbol{u}_{\text{old}}))\|^2 \\ + \|w(\boldsymbol{\sigma} - (\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^T) + pI)\|^2 + \|w(\nabla \cdot \boldsymbol{u})\|^2. \quad (16)$$

For the adaptively weighted method, w is chosen from a previous approximation

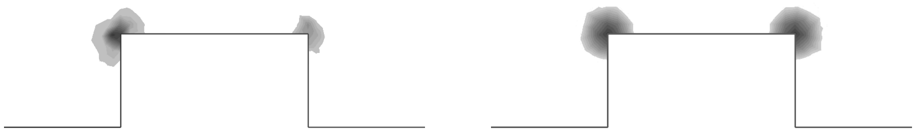


Figure 13. Weight functions on reentrant corners for the adaptive weight (left) and a priori weight (right). White regions represent $w = 1$ while darker regions are closer to $w = 0$.

according to elementwise gradient values,

$$\mathcal{G}(\tau) = \frac{1}{h_\tau^2}(\|\nabla\boldsymbol{\sigma}^h\|_\tau^2 + \|\nabla\boldsymbol{u}^h\|_\tau^2 + \|\nabla p^h\|_\tau^2)^{1/2} \tag{17}$$

and (6) for the weight. For comparison, we also define an a priori weighting approach, which uses a predefined weight function with $w \sim r^\beta$ (Figure 13) near each reentrant corner and $w = 1$ away from the neighborhood of each corner. Based on the known regularity of (15) we may use $\beta = 1.5$ to accelerate convergence. For the standard approach, $w = 1$ over the entire domain. Figure 13 shows a comparison of one adaptively generated weight function and the a priori weight function used.

The computational domain is discretized into N total elements, where we define n as the number of elements on each side of the square obstacle. Figure 14 shows a representative mesh (with $n = 10$) over the computational domain and detail of the local mesh around the square. Numerical results in Figures 16 and 17 use $n = 30$. Computational meshes M1–M4 use $n = 4, 8, 16, 32$, respectively.

We choose the FE spaces based upon the structure of the equations in the system, with $\boldsymbol{\sigma}^h \in \mathrm{RT}_1$ (the next to lowest space of $H(\mathrm{div})$ conforming Raviart–Thomas elements), $\boldsymbol{u}^h \in P_2$ (continuous piecewise quadratic elements), and $p^h \in P_{1\mathrm{dc}}$ (discontinuous piecewise linear elements).

Table 4 summarizes convergence in the functional norm for the three approaches: standard ($w = 1$), adaptive, and a priori. We define a composite global mesh size parameter $h = N^{-1/2}$ where N is the number of elements in the domain. We estimate the rate of convergence to be $\mathcal{O}(h^r)$, with the weighted functional norm $\mathcal{F}_w^{1/2} = (\mathcal{F}_w(\boldsymbol{\sigma}^h, \boldsymbol{u}^h, p^h))^{1/2}$, where $r = \log(\mathcal{F}_{w1}^{1/2}/\mathcal{F}_{w2}^{1/2})/\log(h_1/h_2)$.

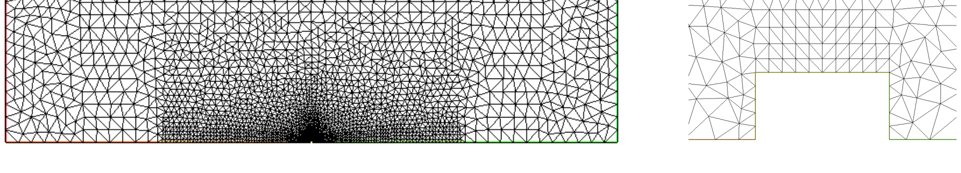


Figure 14. Low-resolution mesh ($n = 10$) over the computational domain (left) and detail around the obstacle (right).

Mesh	N	$\mathcal{F}^{1/2}$ standard	$\mathcal{F}_w^{1/2}$ adaptive	$\mathcal{F}_w^{1/2}$ a priori
M1	720	0.5682	0.4647	0.5246
M2	2880	0.3915	0.2422	0.3117
M3	11520	0.2873	0.1211	0.1253
M4	46080	0.2196	0.0680	0.0493
rate $\approx$		0.39	0.83	1.35

Table 4. Functional norm convergence comparison (at Re $=$ 20) between the standard least squares approximation ($w = 1$), the adaptively weighted approach, and the a priori weighted approach. The meshes are generated in a nested refinement pattern with the structure shown in Figure 14. Convergence rate is estimated from data on the two finest mesh levels.

Convergence is slow for the standard approach while each of the weighted approaches has better convergence.

To further examine computational results, we measure the size of the downstream recirculation eddy and the drag coefficient for a range of Reynolds numbers, comparing values to benchmark solutions published in [31] (noted below as the work of Sen et al.).

We define the reattachment length to be the horizontal distance from the downstream edge of the square to the transition point between recirculation and flow as shown in Figure 15. Computational results summarized in Figure 16 show that both weighted approaches match the values in the reference solution well, while the standard approach significantly under predicts the size of the downstream vortex size.

We define the general coefficient of drag to be

$$C_D = \frac{2}{\text{Re}} \int_s (\sigma \hat{n} \cdot \hat{\imath})\, ds \tag{18}$$

where $\hat{n}$ is a unit vector normal to the surface of the obstacle and $\hat{\imath}$ is a unit vector in the horizontal direction [33]. We can decompose the general formula to this

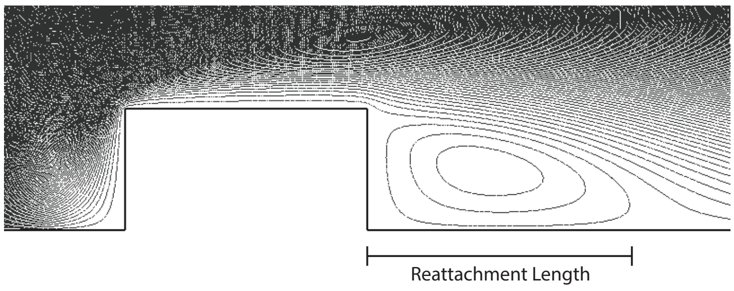


Figure 15. The reattachment length is measured as the distance from the back edge of the square to the transition point between recirculation and flow.

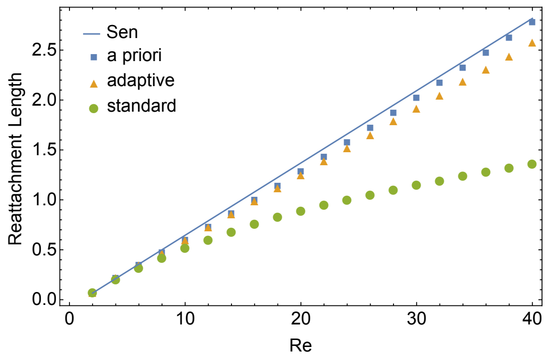


Figure 16. Comparison of reattachment length between different weighting methods (on $n = 30$) and the previous work of Sen et al. At very low Reynolds numbers all methods can be used to good approximation. At $\text{Re} > 10$ only the a priori and adaptive weighting methods continue to be a good approximations. At relatively low mesh resolution, the a priori and adaptive weighting methods produce significantly better results than the standard method.

specific setup, relative to the full domain, as

$$C_D = C_D p + C_D v \tag{19}$$

where

$$C_D p = \frac{2}{\text{Re}} F_p = \frac{2}{\text{Re}} \int_{E,W} p \, dy \tag{20}$$

is the pressure drag, F_p is the force due to pressure,

$$C_D v = \frac{2}{\text{Re}} F_v = \frac{2}{\text{Re}} \int_{N,S} \partial_y u_1 \, dx \tag{21}$$

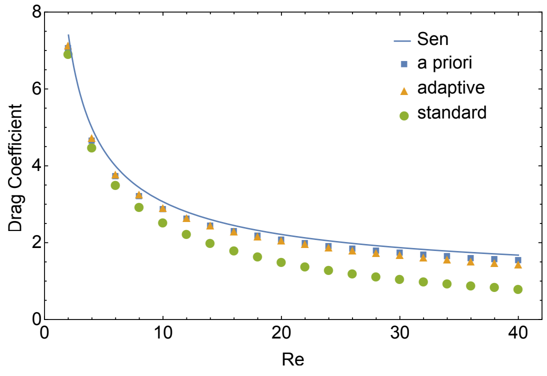


Figure 17. Comparison of the drag coefficient between different weighting methods (on $n = 30$) and the previous work of Sen et al. [31]. Although on a comparatively coarse mesh, both the a priori and adaptive weighting methods are a good approximation to the reference solution.

Mesh	C_D standard	C_D adaptive	C_D a priori	RL standard	RL adaptive	RL a priori
M1	0.60	0.92	0.74	–	–	–
M2	0.87	1.43	1.21	0.41	0.86	0.71
M3	1.14	1.78	1.86	0.64	1.15	1.20
M4	1.46	2.04	2.09	0.86	1.26	1.29

Table 5. Drag coefficient and reattachment length convergence comparison (at Re $= 20$) between the standard least squares approximation ($w = 1$), the adaptively weighted approach, and the a priori weighted approach. The published values for the coefficient of drag and reattachment length are $C_d \approx 2.21$ and RL ≈ 1.37 [31].

is the viscous drag, and F_v is the force due to viscous shear. Here, N, S, E, W represents the north, south, east, and west sides of the square obstacle, respectively. Figure 17 compares drag coefficient values for a range of Reynolds numbers for the three approaches, each computed on a mesh with $n = 30$. As before, the standard approach underpredicts the values while each of the weighted methods produce values close to the reference solution. As a final consideration, in Table 5 we report convergence of drag coefficients and reattachment lengths for a fixed Reynolds number (Re $= 20$) on a sequence of mesh refinements. For each method, values seem to be approaching that of the reference solution, but the weighted methods show better convergence to the reference values, indicating a mitigation of the pollution effect induced by the nonsmooth solution at the reentrant corners.

5. Conclusion

The adaptively weighted least squares approach presented here represents a practical way to treat problems with nonsmooth solutions without requiring the use of exotic finite element spaces or special reformulations of the problem. The general idea can be implemented naturally within an adaptive mesh refinement routine, or within a nonlinear or implicit time stepping iteration, and the additional cost of generating the weight function is small compared with the work required for the full PDE solve. Numerical results demonstrate that the pollution effect due to problems with nonsmooth solutions can be reduced or eliminated, suggesting that the adaptively weighted approach is able to minimize the error in a more optimal norm than using standard L^2 minimization principles.

References

[1] T. Apel and B. Heinrich, *Mesh refinement and windowing near edges for some elliptic problem*, SIAM J. Numer. Anal. **31** (1994), no. 3, 695–708. MR Zbl

[2] M. Berndt, *Adaptive refinement and the treatment of discontinuous coefficients for multilevel First-Order System Least Squares* (*FOSLS*), Ph.D. thesis, University of Colorado Boulder, 1999.

[3] M. Berndt, T. A. Manteuffel, and S. F. McCormick, *Analysis of first-order system least squares* (*FOSLS*) *for elliptic problems with discontinuous coefficients, II*, SIAM J. Numer. Anal. **43** (2005), no. 1, 409–436. MR Zbl

[4] M. Berndt, T. A. Manteuffel, S. F. McCormick, and G. Starke, *Analysis of first-order system least squares* (*FOSLS*) *for elliptic problems with discontinuous coefficients, I*, SIAM J. Numer. Anal. **43** (2005), no. 1, 386–408. MR Zbl

[5] S. Bidwell, M. E. Hassell, and C. R. Westphal, *A weighted least squares finite element method for elliptic problems with degenerate and singular coefficients*, Math. Comp. **82** (2013), no. 282, 673–688. MR Zbl

[6] P. B. Bochev and M. D. Gunzburger, *Least-squares finite element methods*, Applied Mathematical Sciences, no. 166, Springer, 2009. MR Zbl

[7] D. Boffi, F. Brezzi, L. F. Demkowicz, R. G. Durán, R. S. Falk, and M. Fortin, *Mixed finite elements, compatibility conditions, and applications*, Lecture Notes in Mathematics, no. 1939, Springer, 2008. MR Zbl

[8] O. Botella and R. Peyret, *Benchmark spectral results on the lid-driven cavity flow*, Comput. Fluid. **27** (1998), no. 4, 421–433. Zbl

[9] J. H. Bramble, R. D. Lazarov, and J. E. Pasciak, *A least-squares approach based on a discrete minus one inner product for first order systems*, Math. Comp. **66** (1997), no. 219, 935–955. MR Zbl

[10] F. Brezzi and M. Fortin, *Mixed and hybrid finite element methods*, Springer Series in Computational Mathematics, no. 15, Springer, 1991. MR Zbl

[11] P. Burda, *On the FEM for the Navier–Stokes equations in the domains with corner singularities*, Finite element methods: superconvergence, post-processing, and a posterior estimates (M. Křížek, P. Neittaanmäki, and R. Stenberg, eds.), Lecture Notes in Pure and Applied Mathematics, no. 196, Dekker, 1998, pp. 41–52. MR Zbl

[12] P. Burda, J. Novotný, and J. Šístek, *Accurate solution of corner singularities in axisymmetric and plane flows using adjusted mesh of finite elements*, Computational Fluid Dynamics 2004 (C. Groth and D. W. Zingg, eds.), Springer, 2006, pp. 463–468.

[13] Z. Cai, T. A. Manteuffel, and S. F. McCormick, *First-order system least squares for the Stokes equations, with application to linear elasticity*, SIAM J. Numer. Anal. **34** (1997), no. 5, 1727–1741. MR Zbl

[14] Z. Cai, T. A. Manteuffel, S. F. McCormick, and J. Ruge, *First-Order System $\mathscr{L}\mathscr{L}^*$* (*FOSLL**)*: scalar elliptic partial differential equations*, SIAM J. Numer. Anal. **39** (2002), no. 4, 1418–1445.

[15] Z. Cai and C. R. Westphal, *A weighted $H(\mathrm{div})$ least-squares method for second-order elliptic problems*, SIAM J. Numer. Anal. **46** (2008), no. 3, 1640–1651. MR Zbl

[16] ———, *An adaptive mixed least-squares finite element method for viscoelastic fluids of Oldroyd type*, J. Non-Newton. Fluid. Mech. **159** (2009), no. 1–3, 72–80. Zbl

[17] G. F. Carey and H. T. Dinh, *Grading functions and mesh redistribution*, SIAM J. Numer. Anal. **22** (1985), no. 5, 1028–1040. MR Zbl

[18] L. Demkowicz and J. Gopalakrishnan, *Analysis of the DPG method for the Poisson equation*, SIAM J. Numer. Anal. **49** (2011), no. 5, 1788–1809. MR Zbl

[19] U. Ghia, K. N. Ghia, and C. T. Shin, *High-Re solutions for incompressible flow using the Navier–Stokes equations and a multigrid method*, J. Comput. Phys. **48** (1982), no. 3, 387–411. Zbl

[20] P. Grisvard, *Elliptic problems in nonsmooth domains*, Monographs and Studies in Mathematics, no. 24, Pitman, 1985. MR Zbl

[21] F. Hecht, *New development in FreeFem++*, J. Numer. Math. **20** (2012), no. 3–4, 251–265. MR Zbl

[22] S. D. Kim, C.-O. Lee, T. A. Manteuffel, S. F. McCormick, and O. Röhrle, *First-order system least squares for the Oseen equations*, Numer. Lin. Alg. Appl. **13** (2006), no. 7, 523–542. Zbl

[23] V. A. Kondratiev, *Boundary problems for elliptic equations in domains with conical or angular points*, Trans. Moscow Math. Soc. **16** (1967), 227–313.

[24] V. A. Kozlov, V. G. Maz' ya, and J. Rossmann, *Elliptic boundary value problems in domains with point singularities*, Mathematical Surveys and Monographs, no. 52, American Mathematical Society, 1997. MR Zbl

[25] ———, *Spectral problems associated with corner singularities of solutions to elliptic equations*, Mathematical Surveys and Monographs, no. 85, American Mathematical Society, 2001. MR Zbl

[26] J. R. Kweon, *Regularity of solutions for the Navier–Stokes system of incompressible flows on a polygon*, J. Differential Equations **235** (2007), no. 1, 166–198. MR Zbl

[27] E. Lee, T. A. Manteuffel, and C. R. Westphal, *Weighted-norm first-order system least squares (FOSLS) for problems with corner singularities*, SIAM J. Numer. Anal. **44** (2006), no. 5, 1974–1996. MR Zbl

[28] ———, *Weighted-norm first-order system least-squares (FOSLS) for div/curl systems with three dimensional edge singularities*, SIAM J. Numer. Anal. **46** (2008), no. 3, 1619–1639. MR Zbl

[29] K. Liu, T. A. Manteuffel, S. F. McCormick, J. W. Ruge, and L. Tang, *Hybrid first-order system least squares finite element methods with application to Stokes equations*, SIAM J. Numer. Anal. **51** (2013), no. 4, 2214–2237. MR Zbl

[30] R. H. Nochetto, K. G. Siebert, and A. Veeser, *Theory of adaptive finite element methods: an introduction*, Multiscale, nonlinear and adaptive approximation (R. A. DeVore and A. Kunoth, eds.), Springer, 2009, pp. 409–542. MR Zbl

[31] S. Sen, S. Mittal, and G. Biswas, *Flow past a square cylinder at low Reynolds numbers*, Int. J. Numer. Meth. Fluid. **67** (2011), no. 9, 1160–1174. Zbl

[32] J. A. Sethian and J. Wilkening, *A numerical model of stress driven grain boundary diffusion*, J. Comput. Phys. **193** (2004), no. 1, 275–305. MR Zbl

[33] A. Sharma and V. Eswaran, *Heat and fluid flow across a square cylinder in the two-dimensional laminar flow regime*, Numer. Heat Transf. A **45** (2004), no. 3, 247–269.

[34] Y. Sun and C. R. Westphal, *An adaptively weighted Galerkin finite element method for boundary value problems*, Commun. Appl. Math. Comput. Sci. **10** (2015), no. 1, 27–41. MR Zbl

[35] J. A. Wilkening, *Mathematical analysis and numerical simulation of electromigration*, Ph.D. thesis, University of California, Berkeley, 2002. MR

Received November 23, 2015. Revised July 6, 2017.

BRIAN HAYHURST: brian00739@gmail.com
Department of Mathematics and Computer Science, Wabash College, Crawfordsville, IN, United States

MASON KELLER: mgkeller17@wabash.edu
Department of Mathematics and Computer Science, Wabash College, Crawfordsville, IN, United States

CHRIS RAI: chirsinator@gmail.com
Department of Mathematics and Computer Science, Wabash College, Crawfordsville, IN, United States

XIDIAN SUN: sunx23@uw.edu
Department of Mathematics, University of Washington, Seattle, WA, United States

CHAD R. WESTPHAL: westphac@wabash.edu
Department of Mathematics and Computer Science, Wabash College, Crawfordsville, IN, United States

COMM. APP. MATH. AND COMP. SCI.
Vol. 13, No. 1, 2018
dx.doi.org/10.2140/camcos.2018.13.27

ON THE CONVERGENCE OF ITERATIVE SOLVERS FOR POLYGONAL DISCONTINUOUS GALERKIN DISCRETIZATIONS

WILL PAZNER AND PER-OLOF PERSSON

We study the convergence of iterative linear solvers for discontinuous Galerkin discretizations of systems of hyperbolic conservation laws with polygonal mesh elements compared with traditional triangular elements. We solve the semidiscrete system of equations by means of an implicit time discretization method, using iterative solvers such as the block Jacobi method and GMRES. We perform a von Neumann analysis to analytically study the convergence of the block Jacobi method for the two-dimensional advection equation on four classes of regular meshes: hexagonal, square, equilateral-triangular, and right-triangular. We find that hexagonal and square meshes give rise to smaller eigenvalues, and thus result in faster convergence of Jacobi's method. We perform numerical experiments with variable velocity fields, irregular, unstructured meshes, and the Euler equations of gas dynamics to confirm and extend these results. We additionally study the effect of polygonal meshes on the performance of block ILU(0) and Jacobi preconditioners for the GMRES method.

1. Introduction

In recent years, the discontinuous Galerkin (DG) method has become a popular choice for the discretization of a wide range of partial differential equations [27; 6; 15]. This is partly because of its many attractive properties, such as the arbitrarily high degrees of approximation, the rigorous theoretical foundation, and the ability to use fully unstructured meshes. Also, due to its natural stabilization mechanism based on approximate Riemann solvers, it has in particular become widely used in fluid dynamics applications where the high-order accuracy is believed to produce improved accuracy for many problems [32].

Most work on DG methods has been based on meshes of either simplex elements (triangles and tetrahedra), block elements (quadrilaterals and hexahedra), or combinations of these such as prism elements. This is likely because of the availability of excellent automatic unstructured mesh generators, at least for the simplex case

MSC2010: 65F10, 65M60, 65N22.
Keywords: discontinuous Galerkin, iterative solvers, preconditioners.

[22; 28; 30], and also because of the advantages with the outer product structure of block elements. However, it is well known that since no continuity is enforced between the elements, it is straightforward to apply the DG methods to meshes with elements of any shapes (even nonconforming ones). For example, vertex-centered DG methods based on the polygonal dual meshes were studied in [5; 18]. This is a major advantage over standard continuous FEM methods, which need significant developments for the extension to arbitrary polygonal and polyhedral elements [19].

In the finite volume CFD community, there has recently been considerable interest in meshes of arbitrary polygonal and polyhedral elements. In fact, the popular vertex-centered finite volume method applied to a tetrahedral mesh can be seen as a cell-centered method on the dual polyhedral mesh. Because of this, a number of methods have been proposed for generation of polyhedral meshes, which in many cases have advantages over traditional simplex meshes [21; 12]. Although it is still unclear exactly what benefits these elements provide, they have been reported to be both more accurate per degree of freedom and to have better convergence properties in the numerical solvers than for a corresponding tetrahedral mesh [23; 2]. There have also been studies showing that vertex-centered schemes are preferred over cell-centered [10; 9], again indicating the benefits of polyhedral elements.

Inspired by the promising results for the polyhedral finite volume method, and the fact that DG is a natural higher-order extension of these schemes, in this work we study some of the properties of DG discretizations on polygonal meshes. To limit the scope, we only investigate the convergence properties of iterative solvers for the discrete systems, assuming an equal number of degrees of freedom per unit area for all element shapes. Future work will also investigate the accuracy of the solutions on the different meshes. We first consider the iterative block Jacobi method applied to a pure convection problem, which in the constant-coefficient case can be solved analytically using von Neumann analysis. Next we apply the solver to Euler's equations of gas dynamics for relevant model flow problems, to obtain numerical results for the convergence of the various element shapes. We consider regular meshes of hexagons, squares, and two different configurations of triangles, as well as the dual of fully unstructured triangular Delaunay refinement meshes. We also perform numerical experiments with the GMRES Krylov subspace solver and a block ILU preconditioner. Although the results are not entirely conclusive, most of the results indicate a clear benefit with the hexagonal and quadrilateral elements over the triangular ones.

The paper is organized as follows. In Section 2, we describe the spatial and the temporal discretizations, and introduce the iterative solvers. In Section 3 we perform the von Neumann analysis of the constant-coefficient advection problem, in 1D and for several mesh configurations in 2D. In Section 4 we show numerical results for more general advection fields, for more general meshes, as well as for the Euler equations and the GMRES solver. We conclude with a summary of our findings as well as directions for future work.

2. Numerical methods

2.1. ***The discontinuous Galerkin formulation.*** We consider a system of m hyperbolic conservation laws given by the equation

$$\begin{cases} \partial_t \boldsymbol{u} + \nabla \cdot \boldsymbol{F}(\boldsymbol{u}) = 0, & (t, \boldsymbol{x}) \in [0, T] \times \Omega, \\ \boldsymbol{u}(0, \boldsymbol{x}) = \boldsymbol{u}_0(\boldsymbol{x}). \end{cases} \tag{1}$$

In order to describe the discontinuous Galerkin spatial discretization, we divide the spatial domain $\Omega \subseteq \mathbb{R}^2$ into a collection of elements, to form the *triangulation* $\mathcal{T}_h = \{K_i\}$. Often the elements K_i are considered to be triangles or quadrilaterals, but in this paper we allow the elements to be arbitrary polygons in order to study the impact of different tessellations on the efficiency of the algorithm.

Let $V_h = \{v_h \in L_2(\Omega) : v_h|_{K_i} \in P^p(K_i)\}$ denote the space of piecewise polynomials of degree p. We let $\boldsymbol{V}_h^m$ denote the space of vector-valued functions of length m, with each component in V_h. Note that continuity is not enforced between the elements. We derive the discontinuous Galerkin method by replacing $\boldsymbol{u}$ in (1) by an approximate solution $\boldsymbol{u}_h \in \boldsymbol{V}_h^m$, and then multiplying equation by a test function $\boldsymbol{v}_h \in \boldsymbol{V}_h^m$. We then integrate by parts over each element. Because the approximate solution $\boldsymbol{u}_h$ is potentially discontinuous at the boundary of an element, the flux function $\boldsymbol{F}$ is approximated by a *numerical flux function* $\widehat{\boldsymbol{F}}$, which takes as arguments $\boldsymbol{u}^+$, $\boldsymbol{u}^-$, and $\boldsymbol{n}$, denoting the solution on the exterior and interior of the element, and the outward-pointing normal vector, respectively. Then, the discontinuous Galerkin method is as follows: find $\boldsymbol{u}_h \in \boldsymbol{V}_h^m$ such that, for all $\boldsymbol{v}_h \in \boldsymbol{V}_h^m$,

$$\int_{K_i} \partial_t \boldsymbol{u}_h \cdot \boldsymbol{v}_h \, dx - \int_{K_i} \boldsymbol{F}(\boldsymbol{u}_h) : \nabla \boldsymbol{v}_h \, dx + \oint_{\partial K_i} \widehat{\boldsymbol{F}}(\boldsymbol{u}^+, \boldsymbol{u}^-, \boldsymbol{n}) \cdot \boldsymbol{v}_h \, ds = 0. \tag{2}$$

2.2. ***Advection equation.*** As a first example, we consider the two-dimensional scalar advection equation

$$u_t + \nabla \cdot (\boldsymbol{\beta} u) = 0, \tag{3}$$

for a given (constant) velocity vector $\boldsymbol{\beta} = (\alpha, \beta)$. We solve this equation in the domain $[0, 2\pi] \times [0, 2\pi]$, with periodic boundary conditions. The exact solution to this equation is given by

$$u(t, x, y) = u_0(x - \alpha t, y - \beta t), \tag{4}$$

where u_0 is the given initial state.

In order to define the discontinuous Galerkin method for (3), we define the *upwind flux* by

$$\widehat{\boldsymbol{F}}(\boldsymbol{u}^+, \boldsymbol{u}^-, \boldsymbol{n}) = \begin{cases} \boldsymbol{u}^- & \text{if } \boldsymbol{\beta} \cdot \boldsymbol{n} \geq 0, \\ \boldsymbol{u}^+ & \text{if } \boldsymbol{\beta} \cdot \boldsymbol{n} < 0. \end{cases} \tag{5}$$

We represent the approximate solution function $\boldsymbol{u}_h$ as a vector $\boldsymbol{U}$ consisting of the coefficients of the expansion of $\boldsymbol{u}_h$ in terms of an orthogonal Legendre polynomial modal basis of the function space $\boldsymbol{V}_h^m$. Discretizing (3) results in a linear system of equations, which we can write as

$$\boldsymbol{M}(\partial_t \boldsymbol{U}) + \boldsymbol{L}\boldsymbol{U} = 0, \tag{6}$$

where the mass matrix $\boldsymbol{M}$ corresponds to the first term on the left-hand side of (2), and $\boldsymbol{L}$ consists of the second two terms on the left-hand side. The mass matrix is block-diagonal, and the matrix $\boldsymbol{L}$ is a block matrix, with blocks along the diagonal, and off-diagonal blocks corresponding to the boundary terms from the neighboring elements.

2.3. ***Temporal integration and linear solvers.*** We consider the solution of (6) by means of implicit time integration schemes, the simplest of which is the standard backward Euler scheme,

$$(\boldsymbol{M} + k\boldsymbol{L})\boldsymbol{U}^{n+1} = \boldsymbol{M}\boldsymbol{U}^n. \tag{7}$$

Furthermore, each stage of a higher-order scheme, such as a diagonally implicit Runge–Kutta (DIRK) scheme [1], can be written as a similar equation. The block sparse system can be solved efficiently by means of an iterative linear solver. In this paper, we consider two solvers: the simple block Jacobi method, and the preconditioned GMRES method.

2.3.1. *Block Jacobi method.* A popular and simple iterative solver is the block Jacobi method, defined as follows. Each iteration of the method for solving the linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ is given by

$$\boldsymbol{x}^{(n+1)} = \boldsymbol{D}^{-1}\boldsymbol{b} + \boldsymbol{R}_J \boldsymbol{x}^{(n)}, \tag{8}$$

where $\boldsymbol{D}$ is the block-diagonal part of $\boldsymbol{A}$, and $\boldsymbol{R}_J = \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{A}$. This simple method has the advantage that it is possible to analyze the convergence properties of the method simply by examining the eigenvalues of the matrix $\boldsymbol{R}_J$. An upper bound of 1 for the absolute value of the eigenvalues of the matrix $\boldsymbol{R}_J$ is a necessary and sufficient condition in order for Jacobi's method to converge (for any choice of initial vector $\boldsymbol{x}^{(0)}$). The spectral radius of $\boldsymbol{R}_J$ determines the speed of convergence.

2.3.2. *Preconditioned GMRES method.* Another popular and oftentimes more efficient [3] method for solving large, sparse linear systems is the GMRES (generalized minimal residual) method [26]. As with most Krylov subspace methods, the choice of preconditioner has a great impact on the efficiency of the solver [24]. A simple and popular choice of preconditioner is the block Jacobi preconditioner. Each application of this preconditioner is performed by multiplying by the inverse of the block-diagonal part of the matrix. Another, often more effective choice of preconditioner is the block ILU(0) preconditioner [8]. This preconditioner produces an approximate blockwise LU factorization, whose sparsity pattern is enforced to be the same as that of the original matrix. This factorization can be performed in place, and requires no more storage than the original matrix. Unlike the block Jacobi method, the block ILU(0) preconditioner can be highly sensitive to the ordering of the mesh elements [11; 4]. Because of this property, it is common to combine the use of ILU preconditioners with certain orderings of the mesh elements designed to increase efficiency, such as reverse Cuthill–McKee [7], minimum degree [20], nested dissection [13], or minimum discarded fill [26].

In this paper, we focus our study on the block Jacobi method, which is simpler and more amenable to analysis. We then perform numerical experiments using both the block Jacobi method and the preconditioned GMRES method using ILU(0) and block Jacobi preconditioning.

3. Jacobi analysis

We compare tessellations of the plane by four sets of *generating patterns*, each consisting of one or more polygons. We consider tessellations consisting of squares, regular hexagons, two right triangles, and two equilateral triangles. The generating patterns considered are shown in Figure 1. Each generating pattern G_j consists of one or two elements, labeled K_j and $\widetilde{K}_j$. We will refer to these generating patterns as S, H, R, and E for squares, hexagons, right triangles, and equilateral triangles.

We are interested in computing the spectral radius of the Jacobi matrix $\boldsymbol{R_J}$ that arises from the discontinuous Galerkin discretization on the mesh resulting

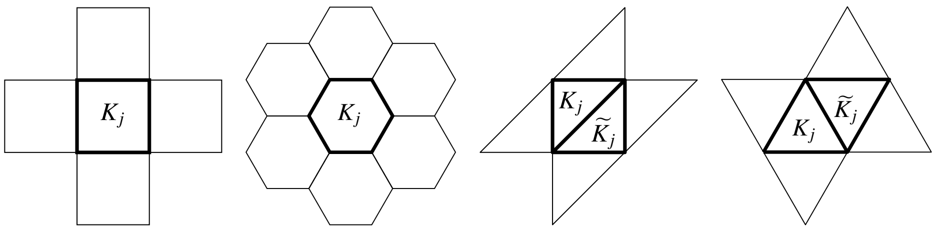


Figure 1. Examples of generating patterns G_j shown with bolded lines. Neighboring elements are shown unbolded. Left to right: square Cartesian grid, regular hexagons, isosceles right triangles, and equilateral triangles.

from tessellating the plane by each of the four generating patterns. For the sake of comparison, we choose the elements from each of the generating patterns to have the same area. Therefore, if the side length of the equilateral triangle is $h_E = h$, then the two equal sides of the isosceles right triangle have side length $h_R = (\sqrt[4]{3}/\sqrt{2})h_E$, the hexagon has side length $h_H = (1/\sqrt{6})h_E$, and the square has side length $h_S = (\sqrt[4]{3}/2)h_E$. Then, the global system will have the same number of degrees of freedom regardless of choice of generating pattern.

3.1. ***Von Neumann analysis.*** First, we compare the efficiency of each of the four types of generating patterns when used to solve the advection equation (3) with the discontinuous Galerkin spatial discretization and implicit time integration. We compute the spectral radius of the matrix $\boldsymbol{R_J}$ using the classical von Neumann analysis for each of the generating patterns, in a manner similar to [16].

Let $\boldsymbol{U}$ denote the solution vector, and let its j-th component, $\boldsymbol{U}_j$, which is itself a vector, denote the degrees of freedom in G_j, the j-th generating pattern. We remark that in the case of squares and hexagons, this corresponds exactly to the degrees of freedom in the element K_j, but in the case of the triangular generating patterns, this corresponds to the degrees of freedom from both of the elements K_j and $\widetilde{K}_j$. In order to determine the eigenvalues of $\boldsymbol{R_J}$, we consider the planar wave with wavenumber (n_x, n_y) defined by

$$\boldsymbol{U}_j = e^{i(n_x x_j + n_y y_j)}\widehat{\boldsymbol{U}}, \tag{9}$$

where (x_j, y_j) are fixed coordinates in G_j. Then, we let ℓ index the generating patterns neighboring G_j, and we let $\boldsymbol{\delta}_\ell = (\delta_{x\ell}, \delta_{y\ell}) = (x_j - x_\ell, y_j - y_\ell)$ be the offsets satisfying $G_j + \boldsymbol{\delta}_\ell = G_\ell$. We can then write the solution in each of the neighboring generating patterns as

$$\boldsymbol{U}_\ell = e^{i(n_x\delta_{x\ell} + n_y\delta_{y\ell})}\boldsymbol{U}_j. \tag{10}$$

In this case we write the semidiscrete equations (6) in the compact form

$$\boldsymbol{M}_j(\partial_t \boldsymbol{U}_j) + \sum_\ell e^{i(n_x\delta_{x\ell} + n_y\delta_{y\ell})}\boldsymbol{L}_{j\ell}\boldsymbol{U}_j = 0, \tag{11}$$

where the summation over ℓ ranges over all neighboring generating patterns, $\boldsymbol{M}_j$ denotes the diagonal block of $\boldsymbol{M}$ corresponding to the j-th generating pattern, and $\boldsymbol{L}_{j\ell}$ denotes the block of $\boldsymbol{L}$ in the j-th row and ℓ-th column. We can write

$$\widehat{\boldsymbol{L}}_j = \sum_\ell e^{i(n_x\delta_{x\ell} + n_y\delta_{y\ell})}\boldsymbol{L}_{j\ell} \tag{12}$$

to further simplify and obtain

$$\boldsymbol{M}_j(\partial_t \widehat{\boldsymbol{U}}) + \widehat{\boldsymbol{L}}_j\widehat{\boldsymbol{U}} = 0. \tag{13}$$

In order to solve (13) using an implicit method, we consider the backward-Euler-type equation

$$(\boldsymbol{M}_j + k\widehat{\boldsymbol{L}}_j)\widehat{\boldsymbol{U}}^{n+1} = \boldsymbol{M}_j\widehat{\boldsymbol{U}}^n. \tag{14}$$

The Jacobi iteration matrix $\boldsymbol{R_J}$ can then be written as

$$\widehat{\boldsymbol{R_J}}_j = \boldsymbol{I} - \boldsymbol{D}^{-1}(\boldsymbol{M}_j + k\widehat{\boldsymbol{L}}_j), \tag{15}$$

where the matrix $\boldsymbol{D} = \boldsymbol{M}_j + k\boldsymbol{L}_{jj}$ consists of the j-th diagonal block of $\boldsymbol{M} + k\boldsymbol{L}$. The eigenvalues of the matrix $\widehat{\boldsymbol{R_J}}_j$ control the speed of convergence of Jacobi's method. In the simple cases of piecewise-constant functions ($p = 0$), or in the case of a one-dimensional domain, the eigenvalues can be computed explicitly. In the more complicated case of $p \geq 1$ in a two-dimensional domain, we compute the eigenvalues numerically.

3.2. ***1D example.*** To illustrate the von Neumann analysis, we consider the one-dimensional scalar advection equation

$$u_t + u_x = 0 \tag{16}$$

on the interval $[0, 2\pi]$ with periodic boundary conditions. We divide the domain into N subintervals K_j, each of length h. Let $\boldsymbol{U}$ denote the solution vector, and let $\boldsymbol{U}_j$ denote the degrees of freedom for the j-th interval K_j. For example, if piecewise constants are used, the method is identical to the upwind finite volume method, and each $\boldsymbol{U}_j$ represents the average of the solution over the interval. If piecewise polynomials of degree p are used, each $\boldsymbol{U}_j$ is a vector of length $p+1$.

For the purposes of illustration, we choose $p = 1$, and let $\boldsymbol{U}_j = (u_{j,1}, u_{j,2})$ represent the value of the solution at the left and right endpoints of the interval K_j. Then, the local basis on the interval K_j consists of the functions

$$\phi_{j,1}(x) = j - x/h, \qquad \phi_{j,2}(x) = x/h - j + 1. \tag{17}$$

We remark that the upwind flux in this case is always equal to the value of the function immediately to the left of the boundary point:

$$[\widehat{\boldsymbol{F}}(u^+, u^-, x)v(x)]_{(j-1)h}^{jh} = u_{j,2}v_{j,2} - u_{j-1,2}v_{j,1}. \tag{18}$$

The entries of the j-th block of the mass matrix $\boldsymbol{M}$ are given by

$$(\boldsymbol{M}_j)_{i\ell} = \int_{(j-1)h}^{jh} \phi_{j,i}(x)\phi_{j,\ell}(x)\,dx. \tag{19}$$

Additionally, we remark that the diagonal blocks of $\boldsymbol{L}$ consist of the volume integrals and right boundary terms given by

$$(\boldsymbol{L}_{jj})_{i\ell} = \phi_{j,i}(jh)\phi_{i,\ell}(jh) - \int_{(j-1)h}^{jh} \phi'_{j,i}(x)\phi_{j,\ell}(x)\,dx. \tag{20}$$

We let $\boldsymbol{A}$ denote the backward-Euler-type operator defined by

$$\boldsymbol{A} = \boldsymbol{M} + k\boldsymbol{L}, \tag{21}$$

and solving the equation $\boldsymbol{Ax} = \boldsymbol{b}$ by means of Jacobi iterations, we define the Jacobi matrix $\boldsymbol{R_J}$ by

$$\boldsymbol{R_J} = \boldsymbol{I} - \boldsymbol{D}^{-1}\boldsymbol{A}, \tag{22}$$

where $\boldsymbol{D}$ is the matrix consisting of the diagonal blocks of $\boldsymbol{A}$. The entries of the diagonal blocks $\boldsymbol{M}_j$ and $\boldsymbol{L}_{jj}$ can be computed explicitly using (17) to obtain

$$\boldsymbol{M}_j = \begin{pmatrix} \frac{1}{3}h & \frac{1}{6}h \\ \frac{1}{6}h & \frac{1}{3}h \end{pmatrix}, \qquad \boldsymbol{L}_{jj} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix}, \qquad \boldsymbol{D}_j = \begin{pmatrix} \frac{1}{3}h + \frac{1}{2}k & \frac{1}{6}h + \frac{1}{2}k \\ \frac{1}{6}h - \frac{1}{2}k & \frac{1}{3}h + \frac{1}{2}k \end{pmatrix}. \tag{23}$$

In order to perform the von Neumann analysis, we seek solutions of the form $\boldsymbol{U}_j = e^{inhj}\widehat{\boldsymbol{U}}$, which allows us to explicitly compute the form of the matrix $\widehat{\boldsymbol{L}}_j$. Recalling the compact form from (13), we obtain

$$\widehat{\boldsymbol{L}}_j = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} - e^{-ihn} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix}. \tag{24}$$

Then, the Jacobi matrix $\widehat{\boldsymbol{R_J}}_j$ is given by

$$\widehat{\boldsymbol{R_J}}_j = \begin{pmatrix} 0 & \dfrac{2e^{-ihn}k(2h+3k)}{h^2+4kh+6k^2} \\ 0 & -\dfrac{2e^{-ihn}(h-3k)k}{h^2+4kh+6k^2} \end{pmatrix}, \tag{25}$$

whose eigenvalues λ_1 and λ_2 are given by

$$\lambda_1 = 0, \qquad \lambda_2 = \frac{2k(3k-h)e^{-ihn}}{h^2+4hk+6k^2}. \tag{26}$$

Therefore, each wavenumber n from 0 to $2\pi/h$ corresponds to an eigenvalue of the Jacobi matrix $\boldsymbol{R_J}$, and the magnitude of these eigenvalues determine the speed of convergence of Jacobi's method. In this case, the expression

$$\lambda_{\max} = \frac{2k|h-3k|}{h^2+4hk+6k^2} \tag{27}$$

determines the speed of convergence of Jacobi's method. This expression can easily be seen to be bounded above by 1 for all positive values of h and k, therefore indicating that Jacobi's method is guaranteed to converge, unconditionally, regardless of spatial resolution or time step.

3.3. ***2D analysis.*** We now turn to the analysis of the four generating patterns shown in Figure 1. The analysis proceeds along the same lines as in the one-dimensional example from Section 3.2. As an example, we present the case of piecewise constants, for which it is possible to explicitly compute the eigenvalues of the Jacobi matrix $\boldsymbol{R_J}$. In this case the discontinuous Galerkin formulation simplifies to the upwind finite volume method

$$\int_{K_j} \partial_t u_h \, dx + \oint_{\partial K_j} \widehat{\boldsymbol{F}}(u^+, u^-, \boldsymbol{n}) \, ds = 0. \tag{28}$$

For the sake of concreteness, we assume without loss of generality that the velocity vector $\boldsymbol{\beta} = (\alpha, \beta)$ satisfies $\alpha, \beta \geq 0$. In order to explicitly write the upwind flux on the meshes consisting of hexagons and equilateral triangles, we further assume that $\sqrt{3}\alpha - \beta \geq 0$, and on the mesh consisting of right triangles we assume that $\alpha - \beta \geq 0$. In the case of the square and hexagonal meshes, there is only one degree of freedom per generating pattern, and we will write u_j to represent the average value of the solution over the generating pattern G_j. We then consider the planar wave with wavenumber (n_x, n_y) given by $u_j = e^{i(n_x x_j + n_y y_j)}\hat{u}$. In the case of the square mesh with side length $h_S = (\sqrt[4]{3}/2)h_E$, the method can be written as

$$h_S^2(\partial_t \hat{u}) = -h_S(\alpha(1 - e^{-in_x h_S}) + \beta(1 - e^{-in_y h_S}))\hat{u}. \tag{29}$$

In this case, the mass matrix $\boldsymbol{M}$ is a diagonal matrix with h_S^2 along the diagonal, and the diagonal entries of the matrix $\boldsymbol{L}$ are given by $h_S(\alpha + \beta)$. Therefore, the eigenvalues of the Jacobi matrix $\boldsymbol{R_J^S} = \boldsymbol{I} - D^{-1}(\boldsymbol{M} + k\boldsymbol{L})$ are given by

$$\begin{aligned} \lambda(\boldsymbol{R_J^S}) &= 1 - \frac{1}{h_S^2 + h_S k(\alpha + \beta)}(h_S^2 + h_S k(\alpha(1 - e^{-in_x h_S}) + \beta(1 - e^{-in_y h_S}))) \\ &= \frac{k(\alpha e^{-in_x h_S} + \beta e^{-in_y h_S})}{h_S + k(\alpha + \beta)}. \end{aligned} \tag{30}$$

In the case of the hexagonal mesh with side length $h_H = (1/\sqrt{6})h_E$, the method is

$$\begin{aligned} \frac{3\sqrt{3}}{2} h_H^2(\partial_t \hat{u}) = -h_H\big((\sqrt{3}\alpha + \beta) + \big(-\tfrac{\sqrt{3}}{2}\alpha + \tfrac{\beta}{2}\big)e^{ih_H(-(3/2)n_x + (\sqrt{3}/2)n_y)} \\ + \big(-\tfrac{\sqrt{3}}{2}\alpha - \tfrac{\beta}{2}\big)e^{ih_H(-(3/2)n_x - (\sqrt{3}/2)n_y)} - \beta e^{-ih_H\sqrt{3}n_y}\big)\hat{u}. \end{aligned} \tag{31}$$

A similar analysis shows that the eigenvalues of the matrix $\boldsymbol{R_J^H}$ are given by

$$\begin{aligned} \lambda(\boldsymbol{R_J^H}) = \frac{ke^{-(1/2)ih_H(3n_x + \sqrt{3}n_y)}}{9h_H + 6\alpha k + 2\sqrt{3}\beta k} \times \\ \big(\sqrt{3}\beta(2e^{(1/2)ih_H(3n_x - \sqrt{3}n_y)} - e^{i\sqrt{3}h_H n_y} + 1) + 3\alpha(1 + e^{i\sqrt{3}h_H n_y})\big). \end{aligned} \tag{32}$$

In the case of the two triangular meshes, there are two degrees of freedom per generating pattern, corresponding to the elements K_j and $\widetilde{K}_j$ in the generating pattern G_j. We write $\boldsymbol{U}_j = (u_{j,1}, u_{j,2})$, where $u_{j,1}$ is the average of the solution over the element K_j, and $u_{j,2}$ is the average of the solution over $\widetilde{K}_j$. The planar wave solution is then given by $\boldsymbol{U}_j = e^{i(n_x x_j + n_y y_j)}\widehat{\boldsymbol{U}}$, for $\widehat{\boldsymbol{U}} = (\hat{u}_1, \hat{u}_2)$. We consider the case of a right-triangular mesh, where the two equal sides of the isosceles right triangles have length $h_R = (\sqrt[4]{3}/\sqrt{2})h_E$. The method then reads

$$\partial_t \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \end{pmatrix} = -\frac{2}{h_R} \begin{pmatrix} \alpha\hat{u}_1 - e^{-ih_R n_x}\alpha\hat{u}_2 \\ \alpha\hat{u}_2 + (\beta-\alpha)\hat{u}_1 - e^{-ih_R n_y}\beta\hat{u}_1 \end{pmatrix}. \tag{33}$$

In the case of the mesh consisting of equilateral triangles, each with side length h_E, the method reads

$$\partial_t \begin{pmatrix} \hat{u}_1 \\ \hat{u}_2 \end{pmatrix} = \frac{-4}{\sqrt{3}h_E} \begin{pmatrix} \left(\frac{\sqrt{3}}{2}\alpha + \frac{1}{2}\beta\right)\hat{u}_1 + \left(e^{-ih_E n_x}\left(-\frac{\sqrt{3}}{2}\alpha + \frac{1}{2}\beta\right) - e^{-ih_E n_y}\beta\right)\hat{u}_2 \\ \left(-\frac{\sqrt{3}}{2}\alpha - \frac{1}{2}\beta\right)\hat{u}_1 + \left(\frac{\sqrt{3}}{2}\alpha + \frac{1}{2}\beta\right)\hat{u}_2 \end{pmatrix}. \tag{34}$$

Computing the eigenvalues of the corresponding Jacobi matrices $\boldsymbol{R}_{\boldsymbol{J}}^{\boldsymbol{R}}$ and $\boldsymbol{R}_{\boldsymbol{J}}^{\boldsymbol{E}}$, we obtain

$$\lambda(\boldsymbol{R}_{\boldsymbol{J}}^{\boldsymbol{R}}) = \pm \frac{2ke^{-(1/2)ih_R(n_x+n_y)}\sqrt{\alpha}\sqrt{\beta + (\alpha-\beta)e^{ih_R n_y}}}{h_R + 2\alpha k}, \tag{35}$$

$$\lambda(\boldsymbol{R}_{\boldsymbol{J}}^{\boldsymbol{E}}) = \pm \frac{2k(3\alpha + \sqrt{3}\beta)\sqrt{2\beta e^{ih_E n_x} + (\sqrt{3}\alpha - \beta)e^{ih_E n_y}}}{(3h_E + 6\alpha k + 2\sqrt{3}\beta k)\sqrt{(\sqrt{3}\alpha + \beta)e^{ih_E(n_x+n_y)}}}. \tag{36}$$

Then, (30), (32), (35), and (36) completely determine the speed of convergence for Jacobi's method of each of the four generating patterns considered. In the case of a higher-order discontinuous Galerkin method with basis consisting of piecewise polynomials of degree $p > 0$, we obtain a Jacobi matrix given by (15), where the matrices $\widehat{\boldsymbol{R}_{\boldsymbol{J}}}_j$, $\boldsymbol{D}$, $\boldsymbol{M}_j$, and $\widehat{\boldsymbol{L}}_j$ are $\frac{1}{2}(p+1)(p+2) \times \frac{1}{2}(p+1)(p+2)$ blocks. In this case, we do not obtain closed-form expressions for the eigenvalues, but rather compute them numerically.

We normalize the velocity magnitude and consider $\boldsymbol{\beta} = (\cos(\theta), \sin(\theta))$. On the square mesh, θ can range from 0 to $\pi/2$. On the hexagonal and equilateral triangle meshes, θ ranges from 0 to $\pi/3$, and on the right-triangular mesh θ ranges from 0 to $\pi/4$. We consider a fixed spatial resolution h, and compare the efficiency of the four patterns for three choices of temporal resolution. We first consider an "explicit" time step, satisfying the CFL-type condition

$$k_{\text{exp}} = \frac{h}{|\boldsymbol{\beta}|}. \tag{37}$$

	$p=0$			$p=1$		
	k_1	k_2	k_3	k_1	k_2	k_3
hexagons	**1.000000**	**1.000000**	**1.000000**	**1.000000**	**1.000000**	**1.000000**
squares	1.128939	1.133989	1.136772	1.058098	1.118222	1.130101
right triangles	1.128939	1.133989	1.136772	1.084223	1.132326	1.137313
equilateral triangles	1.207328	1.215467	1.219948	1.137267	1.201638	1.214376
	$p=2$			$p=3$		
hexagons	**1.000000**	**1.000000**	**1.000000**	1.077183	1.070785	1.066101
squares	1.095785	1.118510	1.129314	**1.000000**	**1.000000**	**1.000000**
right triangles	1.111863	1.126951	1.133634	1.010482	1.005391	1.002733
equilateral triangles	1.177503	1.201918	1.213527	1.074570	1.074570	1.074570

Table 1. Ratio of logarithm of eigenvalues $\log \lambda_{\max}(\boldsymbol{R}_{\boldsymbol{J}}^{\min})/\log \lambda_{\max}(\boldsymbol{R}_{\boldsymbol{J}}^{*})$ ranging over angle θ and wavenumber (n_x, n_y), for piecewise polynomials of degree 0, 1, 2, and 3, for varying choices of time step k. The smallest eigenvalue in each column is in bold.

As one advantage of using an implicit method is that we are not limited by an explicit time step restriction of the form (37), we consider three implicit time steps given by $k_1 = 3k_{\text{exp}}$, $k_2 = 2k_1$, and $k_3 = 4k_1$. We then maximize over a discrete sample of $\theta \in [0, \pi/4]$ and over all wavenumbers (n_x, n_y), in order to compute the maximum eigenvalue for each of the generating patterns. As the number of iterations required to converge to a given tolerance scales like the reciprocal of the logarithm of the spectral radius, we compare the efficiency of the generating patterns by considering the ratio

$$\frac{\log \lambda_{\max}(\boldsymbol{R}_{\boldsymbol{J}}^{\min})}{\log \lambda_{\max}(\boldsymbol{R}_{\boldsymbol{J}}^{*})},$$

where $\lambda_{\max}(\boldsymbol{R}_{\boldsymbol{J}}^{*})$ is the largest eigenvalue of $\boldsymbol{R}_{\boldsymbol{J}}^{*}$, for $* = H, S, R, E$, and $\lambda_{\max}(\boldsymbol{R}_{\boldsymbol{J}}^{\min})$ is the smallest among all $\lambda_{\max}(\boldsymbol{R}_{\boldsymbol{J}}^{*})$. This ratio corresponds to the ratio of iterations required to converge to a given tolerance when compared with the most efficient among the generating patterns. The results obtained for $p = 0, 1, 2, 3$, and $k = k_1, k_2, k_3$ for each generating pattern are shown in Table 1 and Figure 2.

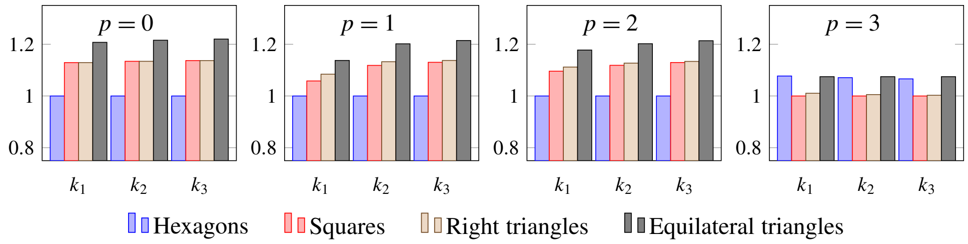


Figure 2. Ratios of the logarithm of the largest eigenvalues for each pattern.

We remark that for polynomials of degree 0, 1, and 2, the hexagonal mesh resulted in the smallest eigenvalues for all choices of time step considered, and the square mesh resulted in the second-smallest eigenvalues. For degree-3 polynomials, the square mesh resulted in the smallest eigenvalues for all cases considered. We notice a significant decrease in the expected performance of the hexagonal elements in the case of $p = 3$, although we have noticed that the effect observed in practice is not as significant as the theoretical results would suggest.

4. Numerical results

4.1. ***Advection with variable velocity field.*** To perform numerical experiments extending the analysis of (3) beyond the case of a constant velocity $\boldsymbol{\beta}$, we consider a variable velocity field $\boldsymbol{\beta}(x, y)$. In this case, the upwind numerical flux

$$\widehat{\boldsymbol{F}}(\boldsymbol{u}^+, \boldsymbol{u}^-, \boldsymbol{n}, x, y) = \begin{cases} \boldsymbol{u}^-(x, y) & \text{if } \boldsymbol{\beta}(x, y) \cdot \boldsymbol{n} \geq 0, \\ \boldsymbol{u}^+(x, y) & \text{if } \boldsymbol{\beta}(x, y) \cdot \boldsymbol{n} < 0 \end{cases} \tag{38}$$

is evaluated pointwise. As an example, we define the velocity to be given by the vector field $\boldsymbol{\beta}(x, y) = (2y - 1, -2x + 1)$ on the spatial domain $\Omega = [0, 1] \times [0, 1]$. This velocity field is shown in Figure 3. We let the initial conditions be given by the Gaussian centered at $(x_0, y_0) = (0.35, 0.5)$:

$$u_0(x, y) = \exp(-150((x - x_0)^2 + (y - y_0)^2)). \tag{39}$$

The exact solution is periodic with period π, and is given by the rotation about the center of the domain:

$$u(x, y, t) = \exp\bigl(-150\bigl((x - 0.5 + 0.15 \cos 2t)^2 + (y - 0.5 - 0.3 \cos t \sin t)^2\bigr)\bigr). \tag{40}$$

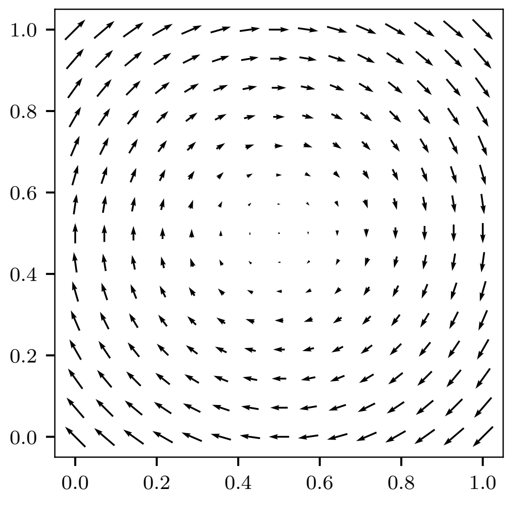


Figure 3. Velocity field $\boldsymbol{\beta}(x, y) = (2y - 1, -2x + 1)$.

	$p=0$			$p=1$			$p=2$			$p=3$		
	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3
hexagons	**33**	**57**	**104**	**21**	**41**	**77**	24	**41**	**77**	**21**	**39**	**75**
squares	35	61	109	**21**	42	83	**22**	42	83	22	42	81
right triangles	39	68	128	26	51	100	25	51	100	25	51	100
equilateral triangles	37	67	123	25	47	92	25	47	92	24	47	91

Table 2. Iterations required for the block Jacobi iterative method to converge in the case of a nonconstant velocity field. The smallest number of iterations in each column is in bold.

4.1.1. *Convergence of the block Jacobi method.* We consider meshes of the domain created by repeating each of the four generating patterns considered in the previous section. As before, for fixed spatial resolution h, we choose h_H, h_S, h_R, and h_E such that the number of degrees of freedom is the same for each mesh. We then solve the advection equation using the backward Euler time discretization, where the block Jacobi iterative method is used to solve the resulting linear system. The zero vector is used as the starting vector for the block Jacobi solver. We choose $h = 0.05$, and since $\max_{(x,y)} |\boldsymbol{\beta}(x, y)| = \sqrt{2}$, we consider time steps of $k_1 = h/\sqrt{2}$, $k_2 = 2k_1$, and $k_3 = 4k_1$. The number of iterations required for the block Jacobi method to converge to a tolerance of 10^{-14} are given in Table 2.

The results are similar to those from the analysis performed in Section 3.3. We note that the hexagonal and square meshes resulted in the lowest number of Jacobi iterations for all of the test cases considered. In contrast to the results of Section 3.3, we do not observe a decrease in the performance of the hexagonal elements for the case of $p = 3$, and instead the performance is similar among all choices of p considered.

4.1.2. *Randomly perturbed mesh.* We now consider the effect of polygonal elements on irregular meshes. To this end, we consider a set of *generating points* distributed evenly on a Cartesian grid with mesh size h. Then, each point is perturbed by a random perturbation sampled uniformly from the interval $[-\delta, \delta]$. We obtain two randomized meshes by constructing the Delaunay triangulation and Voronoi diagram resulting from this set of generating points. The Delaunay mesh consists entirely of triangular elements, whereas the Voronoi diagram is constructed out of arbitrary polygonal elements. Examples of the two meshes considered are shown in Figure 4. In contrast to the regular meshes considered in the previous examples, these two meshes do not consist of the same number of elements. The Voronoi diagram consists of about half the number of elements as the Delaunay triangulation. In the test case considered, the randomized polygonal mesh consists of 410 polygonal elements, whereas the randomized triangular mesh consists of 759 triangular elements.

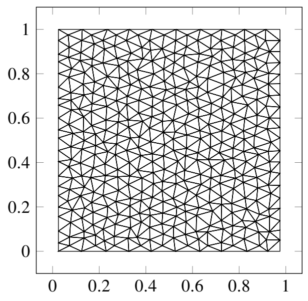
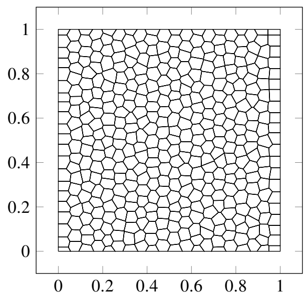



Figure 4. Randomized polygonal and triangular meshes corresponding to the same set of generating points. Left: Delaunay triangulation. Right: Voronoi diagram.

	$p=0$			$p=1$			$p=2$			$p=3$		
	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3
Voronoi diagram	**27**	**32**	**38**	**24**	**33**	**38**	**24**	**32**	**36**	**22**	**31**	**36**
Delaunay triangulation	38	48	52	33	45	48	33	46	50	33	44	48

Table 3. Iterations required for the block Jacobi iterative method to converge in the case of irregular, randomly perturbed meshes. The smallest number of iterations in each column is in bold.

The governing equations and setup are the same as in the previous section. We record the number of block Jacobi iterations required to converge to a tolerance of 10^{-14} in Table 3. Because there is a difference in the number of mesh elements, the resulting linear system will have a different total number of degrees of freedom. This difference will then have an additional effect on the speed of convergence of the block Jacobi method. We note that for polynomials of degree $p = 0, 1, 2, 3$ and for all choices of time step k considered, solving the system resulting from the Voronoi diagram requires fewer block Jacobi iterations than does solving the system resulting from the corresponding Delaunay triangulation.

4.1.3. *Convergence of the GMRES method.* The above analysis focused on the block Jacobi method largely because of the simplicity of the method. In practice, more sophisticated iterative methods are often used [26]. In this section, we consider the solution of the linear system (7) by means of the GMRES method, using both the block Jacobi and the block ILU(0) preconditioners. Since the computational work increases per iteration in GMRES, we choose a *restart parameter* of 20 iterations [29]. We repeat the above test case of the advection equation with variable velocity field and record the number of GMRES iterations required to converge to a tolerance of 10^{-14} using the block Jacobi preconditioner in Table 4.

We now consider the solution of the above problem using the GMRES method with the block ILU(0) preconditioner. Because of the sensitivity of the block ILU(0) factorization to the ordering of the mesh elements, and for the sake of a

	$p=0$			$p=1$			$p=2$			$p=3$		
	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3
hexagons	**31**	**53**	**92**	**25**	**42**	**80**	28	**47**	**86**	28	**49**	**90**
squares	37	64	116	27	51	101	**27**	51	98	**27**	52	100
right triangles	40	70	134	33	61	123	31	60	117	29	59	115
equilateral triangles	39	67	124	33	58	113	32	59	113	31	57	111

Table 4. Iterations required for the GMRES iterative method with block Jacobi preconditioner to converge. The smallest number of iterations in each column is in bold.

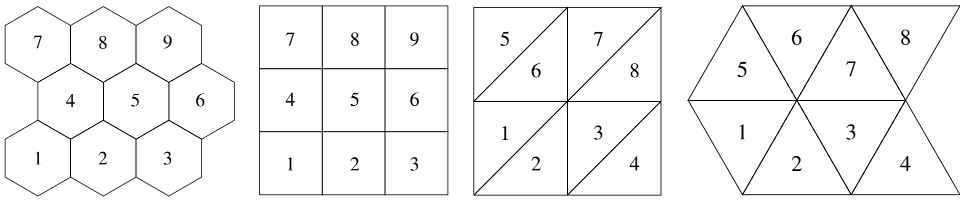


Figure 5. Illustration of the natural ordering of mesh elements. Left to right: square mesh, hexagonal mesh, right-triangular mesh, and equilateral-triangular mesh.

fair comparison between the generating patterns, we consider the *natural ordering* of mesh elements, illustrated in Figure 5. As in the case of the block Jacobi preconditioner, we repeat the test case of the advection equation with variable velocity field. We record the number of GMRES iterations required to converge to the above tolerance using the block ILU(0) preconditioner in Table 5. In this case, the square mesh resulted in the smallest number of iterations in all of the trials. The mesh consisting of right isosceles triangles resulted in the largest number of iterations in all trials. We further note that the number of GMRES iterations required when using the block Jacobi preconditioner scales similarly to the number of block Jacobi iterations required, as recorded in Table 2. We note that the block ILU(0) preconditioner requires fewer GMRES iterations to converge, and the number of iterations scales more favorably in k, when compared with the block Jacobi preconditioner.

	$p=0$			$p=1$			$p=2$			$p=3$		
	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3
hexagons	**8**	11	**16**	10	13	20	11	15	23	10	13	22
squares	**8**	**10**	**16**	**8**	**11**	**19**	**7**	**10**	**17**	**8**	**10**	**18**
right triangles	13	19	32	10	14	28	10	15	27	11	14	28
equilateral triangles	11	15	27	10	12	22	9	12	22	9	12	22

Table 5. Iterations required for the GMRES iterative method with ILU(0) preconditioner to converge. The smallest number of iterations in each column is in bold.

4.2. ***Compressible Euler equations.*** The compressible Euler equations of gas dynamics in two dimensions (see, e.g., [14]) are given by

$$\boldsymbol{u}_t + \nabla \cdot \boldsymbol{f}(\boldsymbol{u}) = 0, \tag{41}$$

for

$$\boldsymbol{u} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \qquad \boldsymbol{f}_1(\boldsymbol{u}) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho H u \end{pmatrix}, \qquad \boldsymbol{f}_2(\boldsymbol{u}) = \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + p \\ \rho H v \end{pmatrix}, \tag{42}$$

where ρ is the density, $\boldsymbol{v} = (u, v)$ is the fluid velocity, p is the pressure, and E is the specific energy. The total enthalpy H is given by

$$H = E + \frac{p}{\rho}, \tag{43}$$

and the pressure is determined by the equation of state

$$p = (\gamma - 1)\rho(E - \tfrac{1}{2}\boldsymbol{v}^2), \tag{44}$$

where $\gamma = c_p/c_v$ is the ratio of specific heat capacities at constant pressure and constant volume.

We consider the model problem of an unsteady compressible vortex in a rectangular domain [32]. The domain is taken to be a 20×15 rectangle, and the vortex is initially centered at $(x_0, y_0) = (5, 5)$. The vortex is moving with the free stream at an angle of θ. The exact solution is given by

$$u = u_\infty \left(\cos(\theta) - \frac{\epsilon((y - y_0) - \bar{v}t)}{2\pi r_c} \exp\left(\frac{f(x, y, t)}{2} \right) \right), \tag{45}$$

$$u = u_\infty \left(\sin(\theta) - \frac{\epsilon((x - x_0) - \bar{u}t)}{2\pi r_c} \exp\left(\frac{f(x, y, t)}{2} \right) \right), \tag{46}$$

$$\rho = \rho_\infty \left(1 - \frac{\epsilon^2(\gamma - 1)M_\infty^2}{8\pi^2} \exp(f(x, y, t)) \right)^{1/(\gamma-1)}, \tag{47}$$

$$p = p_\infty \left(1 - \frac{\epsilon^2(\gamma - 1)M_\infty^2}{8\pi^2} \exp(f(x, y, t)) \right)^{\gamma/(\gamma-1)}, \tag{48}$$

where $f(x, y, t) = (1 - ((x - x_0) - \bar{u}t)^2 - ((y - y_0) - \bar{v}t)^2)/r_c^2$, M_∞ is the Mach number, and u_∞, ρ_∞, and p_∞ are the free-stream velocity, density, and pressure, respectively. The free-stream velocity is given by $(\bar{u}, \bar{v}) = u_\infty(\cos(\theta), \sin(\theta))$. The strength of the vortex is given by ϵ, and its size is r_c. We choose the parameters to be $\gamma = 1.4$, $M_\infty = 0.5$, $u_\infty = 1$, $\theta = \arctan(\frac{1}{2})$, $\epsilon = 0.3$, and $r_c = 1.5$.

	$p=0$			$p=1$			$p=2$			$p=3$		
	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3
hexagons	**32**	**49**	**78**	**31**	**50**	**83**	**50**	**90**	**158**	**53**	**97**	**171**
squares	34	51	89	**31**	54	92	54	99	181	55	105	201
right triangles	37	56	97	41	64	112	58	101	189	59	113	217
equilateral triangles	37	57	95	39	62	113	54	99	179	60	114	215

Table 6. Block Jacobi iterations required per Newton solve of the compressible Euler equations. The lowest number of iterations in each column is in bold.

In the discontinuous Galerkin discretization of the Euler equations we use the Lax–Friedrichs numerical flux defined by

$$\widehat{\boldsymbol{F}}(\boldsymbol{u}^+,\boldsymbol{u}^-,\boldsymbol{n}) = \tfrac{1}{2}(\boldsymbol{f}(\boldsymbol{u}^-)\cdot\boldsymbol{n} + \boldsymbol{f}(\boldsymbol{u}^+)\cdot\boldsymbol{n} + \alpha(\boldsymbol{u}^- - \boldsymbol{u}^+)), \tag{49}$$

where α is the maximum absolute eigenvalue over $\boldsymbol{u}^-$ and $\boldsymbol{u}^+$ of the matrix $B(\boldsymbol{u},\boldsymbol{n})$ defined by

$$B(\boldsymbol{u},\boldsymbol{n}) = J_{f_1} n_1 + J_{f_2} n_2, \tag{50}$$

where J_{f_1} and J_{f_2} are the Jacobian matrices of the components of the numerical flux function $\boldsymbol{f}$ defined in (42).

We use the backward Euler time discretization, but remark that (2) results in a nonlinear set of equations, which is solved using Newton's method. Each iteration of Newton's method requires solving a linear equation of the form (7). We set $h = 1$, and consider three time steps: $k_1 = 0.03h$, $k_2 = 2k_1$, and $k_3 = 4k_1$. We use piecewise polynomials of degrees $p = 0, 1, 2, 3$. Each Newton solve requires between 3 and 8 iterations to converge within a tolerance of 5×10^{-13}. The tolerance used for the linear solvers is the same as in the previous test cases.

4.2.1. *The block Jacobi method.* Each iteration of Newton's method requires the solution of a linear system of equations. We solve these systems using the block Jacobi method. We compute the total number of Jacobi iterations required to complete one solve of Newton's method, and report the results in Table 6. We note that for each choice of p and time step k, the hexagonal mesh required the lowest number of block Jacobi iterations. As in the previous numerical experiments, we do not see a decrease in performance for the hexagonal elements in the case of $p = 3$. The square mesh resulted in the second-smallest number of iterations for most of the cases considered, while the two configurations of triangles resulted in generally similar numbers of iterations.

4.2.2. *The GMRES method.* We now repeat the above test case, using the GMRES method to solve the resulting linear systems. We consider both the block Jacobi and block ILU(0) preconditioners. We then compute the total number of GMRES iterations required to complete one solve of Newton's method. As in Section 4.1.3,

	$p=0$			$p=1$			$p=2$			$p=3$		
	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3
hexagons	**55**	**74**	**106**	**50**	**92**	**126**	**61**	**110**	**153**	**76**	**141**	**195**
squares	62	84	155	52	93	132	67	126	185	78	149	222
right triangles	63	87	162	81	106	184	96	132	242	85	159	299
equilateral triangles	66	90	167	81	108	187	72	133	197	85	161	245

Table 7. GMRES with block Jacobi preconditioner: iterations required per Newton solve of the compressible Euler equations. The lowest number in each column is in bold.

	$p=0$			$p=1$			$p=2$			$p=3$		
	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3	k_1	k_2	k_3
hexagons	**24**	32	**42**	**21**	36	48	29	48	57	29	50	64
squares	**24**	**28**	45	**21**	**33**	**40**	**24**	**41**	**49**	**27**	**48**	**60**
right triangles	31	40	70	35	40	60	36	48	69	31	49	75
equilateral triangles	28	37	65	37	44	70	33	56	68	38	64	80

Table 8. GMRES with block ILU(0) preconditioner: iterations required per Newton solve of the compressible Euler equations. The lowest number in each column is in bold.

the ordering of the mesh elements has a significant effect on the effectiveness of the block ILU(0) approximate factorization. For this reason, we use the natural ordering of elements, depicted in Figure 5. We present the results for the block Jacobi preconditioner in Table 7, and for the block ILU(0) preconditioner in Table 8. With the block Jacobi preconditioner, the hexagonal mesh required the smallest number of iterations for all test cases considered, and the square mesh the second-smallest. In the case of the block ILU(0) preconditioner, the square mesh required the lowest number of iterations, with the hexagonal mesh usually requiring the second-smallest number of iterations. As we observed in Section 4.1.3, the number of iterations required for both the block Jacobi method and GMRES with the block Jacobi preconditioner scales quite poorly with increasing time steps. The number of GMRES iterations required when using the block ILU(0) preconditioner is significantly better.

4.3. ***Inviscid flow problems.*** The following two numerical experiments extend the above results to larger-scale, more realistic flow problems. These problems, in contrast to the preceding test cases, are characterized by a large number of degrees of freedom, the presence of geometric features and wall boundary conditions, variably sized mesh elements, and shocks. As in the previous section, the equations considered here are the compressible Euler equations. For the following two problems, we choose the finite element function space to consist of piecewise-constant functions (corresponding to $p = 0$), which results in a finite-volume-type discretization. This choice of discretization allows for the solution of problems with shocks, without the use of slope limiters, artificial viscosity, or other shock-capturing techniques [17]. The Roe numerical flux is used as an approximate Riemann solver for these problems.

4.3.1. *Subsonic flow over a circular cylinder.* For a first test case, we consider the inviscid flow over a circular cylinder at Mach 0.2. The computational domain is defined as $\Omega = R \setminus C$, where $R = [-10, 30] \times [-10, 20]$, and C is a disk of radius 1 centered at the point $(5, 5)$. Far-field boundary conditions are enforced on ∂R, and a no-normal-flow condition is enforced on ∂C. The free-stream velocity is taken to be unity in the x-direction, and $\rho_\infty = 1$.

For this test case we use four unstructured meshes, two consisting entirely of triangles and two consisting of mixed polygons, generated using the PolyMesher algorithm [31]. All the meshes are created using a gradient-limited element size function that determines the initial distribution of seed points according to the rejection method [25], such that the element edge length near the surface of the cylinder is about one-fifth the edge length of elements away from the cylinder. For both the triangular and polygonal meshes, we consider a coarse mesh, with 15,404 elements, and a fine mesh with 62,270 elements. Thus, the average area of each element is the same for both the polygonal and triangular meshes. Additionally, the number of degrees of freedom in the solution is the same, allowing for a fair comparison. The coarse polygonal mesh and a zoom-in around the surface of the cylinder are shown in Figure 6.

Starting from free-stream initial conditions, we integrate the equations until $t = 5 \times 10^{-3}$ in order to obtain a representative solution. Using this solution, we then compute 10 time steps using a third-order A-stable DIRK method [1]. Each stage of the DIRK method requires the solution of a nonlinear system of equations,

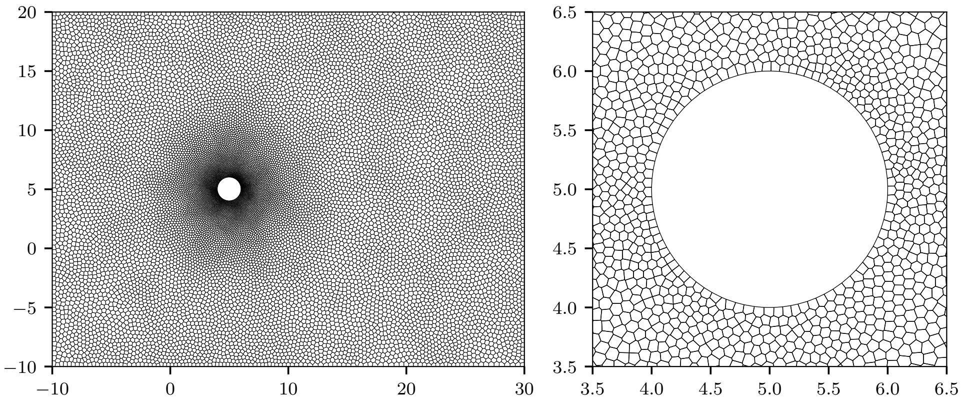


Figure 6. Overview of the coarse mesh with 15,404 elements, with zoom-in showing polygonal elements near the surface of the cylinder.

	ILU		Jacobi		ratios	
Δt	polygonal	triangular	polygonal	triangular	ILU	Jacobi
1.0×10^{-1}	793	932	2092	3126	0.85	0.67
2.5×10^{-1}	1569	1829	4405	6870	0.86	0.64
5.0×10^{-1}	2470	3090	7145	11859	0.80	0.60
1.0	3651	4486	11054	18880	0.81	0.59
1.0×10^{-1}	1443	1673	4075	6137	0.86	0.66
2.5×10^{-1}	2998	3344	8732	12741	0.90	0.69
5.0×10^{-1}	4720	5423	14084	21882	0.87	0.64
1.0	7205	8151	22814	34706	0.88	0.66

Table 9. Total GMRES iterations per 10 time steps for inviscid flow over a circular cylinder. Top: coarse grid with 15,404 elements. Bottom: fine mesh with 95,932 elements.

Δt	polygonal	triangular	ratio	polygonal	triangular	ratio
1.0×10^{-1}	2474	3159	0.78	4788	6281	0.76
2.5×10^{-1}	4895	6697	0.73	9609	12406	0.77
5.0×10^{-1}	7882	12158	0.65	15580	20946	0.74
1.0	13181	19072	0.69	26628	33934	0.78

Table 10. Total block Jacobi iterations per 10 time steps for inviscid flow over a circular cylinder. Left: coarse grid with 15,404 elements. Right: fine mesh with 95,932 elements.

which we solve by means of Newton's method. In each iteration of Newton's method, we solve the resulting linear system of the form (7) using both the block Jacobi method and the preconditioned GMRES method. The nonlinear system is solved to within a tolerance of 10^{-8}, and each linear system is solved using a relative tolerance of 10^{-5}. For the GMRES method, we consider two preconditioners: block Jacobi, and block ILU(0). In order to compare the iterative solver performance differences between meshes, we compute the total number of solver iterations required to complete all 10 time steps. The results for the GMRES method are shown in Table 9, and for the block Jacobi solver in Table 10.

These results demonstrate a consistent trend, corroborating both the numerical results and the analysis from the previous sections. When using the block Jacobi solver or GMRES with block Jacobi preconditioner, the polygonal mesh results in convergence in 60–70% of the iterations required for the triangular mesh. The effect is smaller when using the ILU(0) preconditioner, but we do still observe a modest reduction in the number of iterations required. When using the block Jacobi iterative solver, we observe iteration counts very similar to when using GMRES with block Jacobi as a preconditioner. In these cases, the polygonal mesh requires 70–80% of the iterations as the all-triangular mesh.

4.3.2. *Supersonic flow over a circular cylinder.* The next numerical example is designed to investigate the performance of the iterative solvers for steady-state problems, in the presence of shocks and h-adapted meshes. For this problem, we let the domain be $\Omega = R \setminus C$, where $R = [0, 5] \times [0, 10]$ and, as before, C is a circle of radius 1 centered at $(5, 5)$. Free-stream conditions are enforced at the left, top, and bottom boundaries, an inviscid wall condition is enforced on the boundary of the cylinder, and an outflow condition is enforced on the right boundary. The Mach number is set to $M = 2.0$, resulting in the formation of a shock upstream from the cylinder. In order to accurately capture the shock, we refine the mesh in its vicinity. As in the previous case, we consider a set of four meshes: two all-triangular and two polygonal. For both the triangular and polygonal meshes, we consider coarse and fine versions, with 31,162 and 95,932 elements, respectively. The coarse mesh is depicted in Figure 7, left, with Mach isolines overlaid to indicate the position of the shock. Additionally, Mach contours of the steady-state solution are shown in Figure 7, right.

Beginning with free-stream initial conditions, the solution rapidly approaches a steady state. We integrate in time until $t = 100$ in order to obtain a solution which can be used as an initial guess for the steady-state Newton solve. Then, starting with this solution, we set the time derivative of the solution to zero and solve the resulting nonlinear equations using Newton's method to find a steady-state solution. The

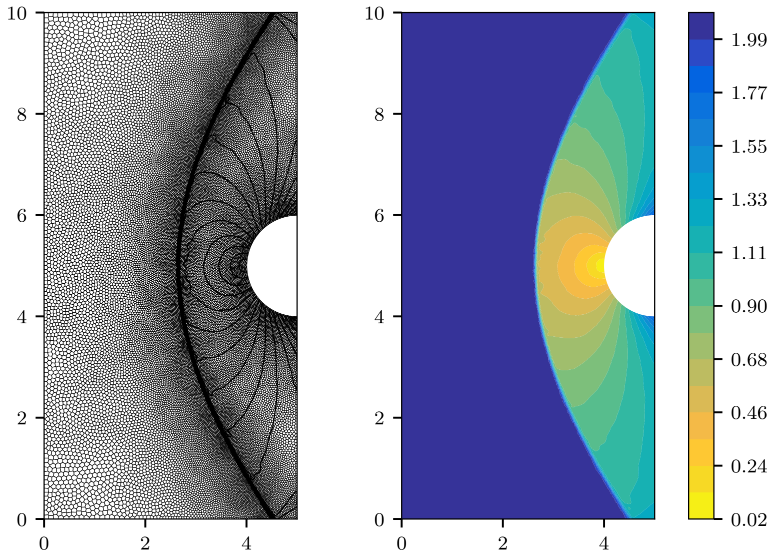


Figure 7. Overview of coarse polygonal mesh with 31,162 elements, showing Mach number contours for steady-state solution. Left: coarse mesh for supersonic test problem, showing Mach isolines for steady-state solution. Right: contours of Mach number for steady-state solution.

	polygonal	triangular	ratio	polygonal	triangular	ratio
ILU	469	640	0.73	953	1947	0.49
Jacobi	2340	6464	0.36	–	–	–

Table 11. Total GMRES iterations per steady-state solve for supersonic flow over a cylinder. Left: coarse grid with 31,162 elements. Right: fine mesh with 95,932 elements.

resulting linear system that is required to be solved at each iteration can be thought of as corresponding to (7), where formally we set $k = \infty$. The nonlinear system is solved to within a tolerance of 10^{-10}, and each linear system is solved using a relative tolerance of 10^{-5}. Since the mass matrix in (7) acts to regularize the linear system, the conditioning becomes worse for larger values of k, and the number of iterations required per linear solve grows. Hence, effective preconditioners are particularly important for the solution of such steady-state problems. For these problems, the block Jacobi iterative solver did not converge in fewer than 10,000 iterations, and so we consider only the GMRES method, using block ILU(0) and block Jacobi preconditioners.

We present the comparison of iteration counts for this problem in Table 11. On the coarse meshes, the ILU(0) preconditioner required about 73% as many iterations on the polygonal mesh when compared with the triangular mesh. This difference is more significant when using the block Jacobi preconditioner, consistent with the results observed in previous sections. In this case, the polygonal mesh requires only slightly more than one third the number of iterations as the all-triangular mesh. On the fine mesh, there are close to half a million degrees of freedom. For a problem of this scale, we did not observe convergence in fewer than 10,000 iterations per linear solve using the block Jacobi preconditioner, and so we only compare performance using the block ILU(0) preconditioner. In this case, the polygonal mesh required about half as many iterations per steady-state solve when compared with the all-triangular mesh.

5. Conclusions

In this paper we have analyzed the effect of the generating pattern of a regular mesh on the convergence of iterative linear solvers applied to implicit discontinuous Galerkin discretizations. We considered four generating patterns: a hexagon, a square, two right triangles, and two equilateral triangles.

A classical von Neumann analysis applied to the constant-velocity advection equation allowed us to compute the eigenvalues of the block Jacobi matrix, and therefore estimate the speed of convergence of the block Jacobi method. In more than half of the cases considered, the hexagonal generating pattern resulted in the smallest eigenvalues, and in the remaining cases, the square generating pattern resulted in the smallest eigenvalues.

In order to extend these results beyond the case of the constant-velocity advection equation, we performed numerical experiments on the variable-velocity advection equation and compressible Euler equations. In the case of the advection equation, in all but one case the hexagonal mesh resulted in the fastest convergence, and in the remaining case the square mesh resulted in the fastest convergence. In the case of the Euler equations, the hexagonal mesh resulted in the fastest convergence in all test cases.

We additionally considered two irregular meshes resulting from the random perturbation of a set of regularly spaced generating points. We obtain a triangular mesh by performing the Delaunay triangulation on these points, and we obtain a polygonal mesh by constructing the Voronoi diagram dual to the Delaunay triangulation. Solving the advection equation on these irregular meshes, we observed that the block Jacobi method converged faster on the polygonal mesh in every test case. Additionally, we performed numerical experiments examining the performance of the GMRES iterative method when used with the ILU(0) preconditioner. We found that in all of the test cases, the square generating pattern resulted in the lowest number of GMRES iterations, and in all but two cases, the hexagonal generating pattern resulted in the second-lowest number of iterations.

For a final set of numerical experiments, we performed two inviscid fluid flow simulations on sets of coarse and fine meshes. Each mesh was either all-triangular, or was composed of arbitrary polygons. We measured iteration counts for both time-dependent and steady-state problems, using the block Jacobi method, and GMRES with block ILU(0) and block Jacobi preconditioners. We found that the polygonal meshes resulted in faster convergence of the iterative solvers, with a larger difference being observed for the block Jacobi method and preconditioner. This difference was more pronounced for the steady-state problem, with quite a significant difference observed on the fine mesh using GMRES with ILU(0).

These results suggest that certain types of polygonal meshes have the advantage of rapid convergence of iterative solvers. Future research directions involve the study of accuracy of DG methods on polygonal and polyhedral meshes, efficient computation of quadrature rules over arbitrary polygonal domains, and the extension of the above results to three spatial dimensions.

Acknowledgements

This work was supported by the AFOSR Computational Mathematics program under grant number FA9550-15-1-0010. Pazner was supported by the Department of Defense through the National Defense Science and Engineering Graduate Fellowship Program and by the Natural Sciences and Engineering Research Council of Canada.

References

[1] R. Alexander, *Diagonally implicit Runge–Kutta methods for stiff O.D.E.'s*, SIAM J. Numer. Anal. **14** (1977), no. 6, 1006–1021. MR Zbl

[2] G. Balafas, *Polyhedral mesh generation for CFD-analysis of complex structures*, master's thesis, Technische Universität München, 2014.

[3] F. Bassi and S. Rebay, *GMRES discontinuous Galerkin solution of the compressible Navier–Stokes equations*, Discontinuous Galerkin methods (B. Cockburn, G. E. Karniadakis, and C.-W. Shu, eds.), Lect. Notes Comput. Sci. Eng., no. 11, Springer, 2000, pp. 197–208. MR Zbl

[4] M. Benzi, W. Joubert, and G. Mateescu, *Numerical experiments with parallel orderings for ILU preconditioners*, Electron. Trans. Numer. Anal. **8** (1999), 88–114. MR Zbl

[5] M. Berggren, *A vertex-centered, dual discontinuous Galerkin method*, J. Comput. Appl. Math. **192** (2006), no. 1, 175–181. MR Zbl

[6] B. Cockburn and C.-W. Shu, *Runge–Kutta discontinuous Galerkin methods for convection-dominated problems*, J. Sci. Comput. **16** (2001), no. 3, 173–261. MR Zbl

[7] E. Cuthill and J. McKee, *Reducing the bandwidth of sparse symmetric matrices*, ACM '69: proceedings of the 1969 24th National Conference, Association for Computing Machinery, 1969, pp. 157–172.

[8] L. T. Diosady and D. L. Darmofal, *Preconditioning methods for discontinuous Galerkin solutions of the Navier–Stokes equations*, J. Comput. Phys. **228** (2009), no. 11, 3917–3935. MR Zbl

[9] B. Diskin and J. L. Thomas, *Comparison of node-centered and cell-centered unstructured finite-volume discretizations: inviscid fluxes*, AIAA J. **49** (2011), no. 4, 836–854.

[10] B. Diskin, J. L. Thomas, E. J. Nielsen, H. Nishikawa, and J. A. White, *Comparison of node-centered and cell-centered unstructured finite-volume discretizations: viscous fluxes*, AIAA J. **48** (2010), no. 7, 1326–1338.

[11] I. S. Duff and G. A. Meurant, *The effect of ordering on preconditioned conjugate gradients*, BIT **29** (1989), no. 4, 635–657. MR Zbl

[12] R. V. Garimella, J. Kim, and M. Berndt, *Polyhedral mesh generation and optimization for non-manifold domains*, Proceedings of the 22nd International Meshing Roundtable (J. Sarrate and M. Staten, eds.), Springer, 2014, pp. 313–330.

[13] A. George, *Nested dissection of a regular finite element mesh*, SIAM J. Numer. Anal. **10** (1973), 345–363. MR Zbl

[14] R. Hartmann, *Discontinuous Galerkin methods for compressible flows: higher order accuracy, error estimation and adaptivity*, 34th CFD: higher order discretization methods (H. Deconinck and M. Ricchiuto, eds.), Von Karman Institute Lecture Series, no. 2006-01, Von Karman Institute, 2006.

[15] J. S. Hesthaven and T. Warburton, *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*, Texts in Applied Mathematics, no. 54, Springer, 2008. MR Zbl

[16] E. J. Kubatko, C. Dawson, and J. J. Westerink, *Time step restrictions for Runge–Kutta discontinuous Galerkin methods on triangular grids*, J. Comput. Phys. **227** (2008), no. 23, 9697–9710. MR Zbl

[17] R. J. LeVeque, *Finite volume methods for hyperbolic problems*, Cambridge University, 2002. MR Zbl

[18] H. Luo, J. D. Baum, and R. Löhner, *A discontinuous Galerkin method based on a Taylor basis for the compressible flows on arbitrary grids*, J. Comput. Phys. **227** (2008), no. 20, 8875–8893. MR Zbl

[19] G. Manzini, A. Russo, and N. Sukumar, *New perspectives on polygonal and polyhedral finite element methods*, Math. Models Methods Appl. Sci. **24** (2014), no. 8, 1665–1699. MR Zbl

[20] H. M. Markowitz, *The elimination form of the inverse and its application to linear programming*, Management Sci. **3** (1957), 255–269. MR Zbl

[21] W. Oaks and S. Paoletti, *Polyhedral mesh generation*, 9th International Meshing Roundtable, Sandia National Laboratories, 2000, pp. 57–67.

[22] J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz, *Adaptive remeshing for compressible flow computations*, J. Comput. Phys. **72** (1987), no. 2, 449–466. Zbl

[23] M. Peric, *Flow simulation using control volumes of arbitrary polyhedral shape*, ERCOFTAC Bull. **62** (2004), 25–29.

[24] P.-O. Persson and J. Peraire, *Newton–GMRES preconditioning for discontinuous Galerkin discretizations of the Navier–Stokes equations*, SIAM J. Sci. Comput. **30** (2008), no. 6, 2709–2733. MR Zbl

[25] P.-O. Persson, *Mesh generation for implicit geometries*, Ph.D. thesis, Massachusetts Institute of Technology, 2005. MR

[26] ———, *Scalable parallel Newton–Krylov solvers for discontinuous Galerkin discretizations*, 47th AIAA Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, 2009.

[27] W. H. Reed and T. R. Hill, *Triangular mesh methods for the neutron transport equation*, technical report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.

[28] J. Ruppert, *A Delaunay refinement algorithm for quality* 2*-dimensional mesh generation*, J. Algorithms **18** (1995), no. 3, 548–585. MR Zbl

[29] Y. Saad, *Iterative methods for sparse linear systems*, 2nd ed., Society for Industrial and Applied Mathematics, 2003. MR Zbl

[30] J. R. Shewchuk, *Delaunay refinement algorithms for triangular mesh generation*, Comput. Geom. **22** (2002), no. 1–3, 21–74. MR Zbl

[31] C. Talischi, G. H. Paulino, A. Pereira, and I. F. M. Menezes, *PolyMesher: a general-purpose mesh generator for polygonal elements written in Matlab*, Struct. Multidiscip. Optim. **45** (2012), no. 3, 309–328. MR Zbl

[32] Z. J. Wang, K. Fidkowski, R. Abgrall, and et al., *High-order CFD methods: current status and perspective*, Internat. J. Numer. Methods Fluids **72** (2013), no. 8, 811–845. MR

Received August 8, 2016. Revised September 25, 2017.

WILL PAZNER: will_pazner@brown.edu
Division of Applied Mathematics, Brown University, Providence, RI, United States

PER-OLOF PERSSON: persson@berkeley.edu
Department of Mathematics, University of California, Berkeley, Berkeley, CA, United States

COMM. APP. MATH. AND COMP. SCI.
Vol. 13, No. 1, 2018
dx.doi.org/10.2140/camcos.2018.13.53

THEORETICALLY OPTIMAL INEXACT SPECTRAL DEFERRED CORRECTION METHODS

MARTIN WEISER AND SUNAYANA GHOSH

In several initial value problems with particularly expensive right-hand side evaluation or implicit step computation, there is a tradeoff between accuracy and computational effort. We consider inexact spectral deferred correction (SDC) methods for solving such initial value problems. SDC methods are interpreted as fixed-point iterations and, due to their corrective iterative nature, allow one to exploit the accuracy-work tradeoff for a reduction of the total computational effort. First we derive error models bounding the total error in terms of the evaluation errors. Then we define work models describing the computational effort in terms of the evaluation accuracy. Combining both, a theoretically optimal local tolerance selection is worked out by minimizing the total work subject to achieving the requested tolerance. The properties of optimal local tolerances and the predicted efficiency gain compared to simpler heuristics, and reasonable practical performance, are illustrated with simple numerical examples.

1. Introduction

The numerical solution of initial value problems of the form

$$y'(t) = f(y(t)), \quad y(0) = y_0,$$

can involve a significant amount of computation, where the most effort is usually spent either on evaluating complex right-hand sides in nonstiff problems or on solving large linear equation systems in stiff systems. Often, there is an accuracy-effort tradeoff, such that inexact results can be obtained much faster than exact results. Examples for the first type of problem are molecular and stellar dynamics, where the exact evaluation of long-range interactions is $\mathcal{O}(N^2)$ but can be approximated by clustering or fast multipole methods in $\mathcal{O}(N \log N)$ or $\mathcal{O}(N)$ time [4; 6], or cycle jump techniques for highly oscillatory problems of wear or fatigue [10; 13]. Typical examples of the second type of problem are reaction-diffusion equations, where implicit time-stepping schemes rely on iterative solvers [24; 28],

A preprint of this paper appeared as Zuse Institute Berlin report 16-52.
MSC2010: 65L05, 65L20, 65L70, 65M70.
Keywords: spectral deferred corrections, initial value problems, error propagation, adaptive control of tolerances, inexact, work models, accuracy models.

While the possibilities to exploit the tradeoff between accuracy and computational effort for improved simulation performance are rather limited in usual time-stepping schemes such as explicit or implicit Runge–Kutta, extrapolation, or multistep schemes, iterative methods for solving implicit Runge–Kutta equations [3; 12; 26] can in principle correct inexact evaluations of intermediate quantities in subsequent iterations. Spectral deferred correction (SDC) methods [11] as iterative solvers for collocation systems have a particularly simple structure and are therefore considered here. Inexact implicit SDC methods with errors due to truncation of multigrid iterations have been investigated numerically in [20; 24], where a small fixed number of V-cycles has been found to be sufficient for convergence. Mixed-precision arithmetic for SDC has been proposed in [15] and found to save some computational effort. In this paper, we will analyze the error propagation through the SDC iteration and, following the approach of Alfeld [1] for inexact fixed-point iterations, derive an a priori selection of local tolerances for right-hand side evaluation and substep computation that leads to theoretically optimal efficiency of the overall integration scheme. Usually, explicit Runge–Kutta methods are hard to beat in efficiency by more complex methods such as SDC, but the results derived here indicate that this might be possible if inexactness can be exploited.

The remainder of the paper is organized as follows. Section 2 states the precise problem setting before briefly recalling spectral deferred correction methods and discussing the impact of inexact evaluations. The main Section 3 introduces error models for quantifying the error propagation, work models for quantifying the computational cost, and the optimization of accuracy per work to derive an optimal selection of tolerances. Effectiveness and efficiency of the resulting methods are illustrated in Section 4 with some numerical examples.

2. Inexactness in spectral deferred correction methods

The autonomous initial value problem (IVP) to be solved is given by

$$\begin{cases} y'(t) = f(y(t)), & t \in [0, T], \\ y(0) = y_0, \end{cases} \tag{2-1}$$

where the right-hand side f is a mapping $f : Y \to Y$ on a Banach space Y, and $t \in [0, T]$ denotes the time variable. It is assumed that f is continuous and locally Lipschitz continuous. Under these assumptions, a unique solution $y(t)$ exists; see, e.g., [9; 25]. An approximate numerical solution can be determined with time-stepping schemes. We consider single-step methods, where the time interval $[0, T]$ is subdivided into individual steps and the connection between the subintervals consists of transferring the value of y at the end point of one subinterval as the initial value for the following subinterval. Without loss of generality, we therefore restrict the presentation to a single time step $[0, T]$. Also, without loss of generality, we assume (2-1) to be autonomous.

2A. ***Collocation conditions.*** Given the IVP (2-1), a collocation method approximates the exact solution y over the interval $[0, T]$ by a polynomial y_c satisfying (2-1) at N discrete collocation points t_i, $i = 1, \dots, N$, within the interval $[0, T]$:

$$\begin{cases} y_c'(t_i) = f(y_c(t_i)), & i = 1, \dots, N, \\ y_c(0) = y_0. \end{cases} \tag{2-2}$$

For simplicity of indexing, we define $t_0 = 0$. Popular choices for collocation points are equidistant nodes or Gauss–Legendre, Lobatto, or Radau points. For a detailed discussion of collocation methods, we refer to [9; 17].

The IVP (2-1) can be written equivalently as the Picard integral equation

$$y(t) = y_0 + \int_0^t f(y(\tau))\, d\tau,$$

which leads to corresponding Picard collocation conditions, as described in [18]:

$$\begin{cases} y_c(t_i) = y_c(t_{i-1}) + \sum_{k=1}^N S_{ik} f(y_c(t_k)), & i = 1, \dots, N, \\ y_c(0) = y_0, \end{cases} \tag{2-3}$$

where the entries of the spectral quadrature matrix $S \in \mathbb{R}^{N \times N}$ are defined in terms of the Lagrange polynomials $L_k \in \mathbb{P}_{N-1}[\mathbb{R}]$ satisfying $L_k(t_i) = \delta_{ik}$ for $i = 1, \dots, N$ as

$$S_{ik} = \int_{\tau = t_{i-1}}^{t_i} L_k(\tau)\, d\tau, \quad i, k = 1, \dots, N.$$

2B. ***Spectral deferred correction method.*** The direct solution of the collocation system (2-2) or (2-3) can be quite involved if N is larger than one or two. As the time discretization error of the collocation method is present anyway, an exact solution of (2-2) is not required. Thus, iterative methods form an interesting class of solvers; see, e.g., [7; 8; 19]. Here we consider spectral deferred correction (SDC) methods. They were introduced by Dutt, Greengard, and Rokhlin [11] for fixed iteration number as time-stepping schemes in their own right, and only later on have been interpreted as fixed-point iterations for collocation systems [18; 27]. In SDC, the Picard collocation conditions (2-3) are solved iteratively by a defect-correction procedure. Using the Picard formulation has the advantage of faster convergence for nonstiff problems [27].

Approximate solutions are polynomials $y^{[j]} \in \mathbb{P}_N[Y]$, identified with vectors in Y^{N+1} by interpolation of their values $y_i^{[j]} := y^{[j]}(t_i)$ at the $N+1$ grid points t_i. Given an approximate solution $y^{[j]}$, the *error* $y_c - y^{[j]}$ satisfies the Picard collocation conditions

$$\begin{aligned} y_c(t_i) - y_i^{[j]} &= (y_c - y^{[j]})(t_{i-1}) + \sum_{k=1}^{N} S_{ik}(f(y_c(t_k)) - y^{[j]\prime}(t_k)) \\ &= (y_c - y^{[j]})(t_{i-1}) + \sum_{k=1}^{N} S_{ik}(f(y_c(t_k)) - f(y_k^{[j]})) \\ &\qquad + \sum_{k=1}^{N} S_{ik}(f(y_k^{[j]}) - y^{[j]\prime}(t_k)) \end{aligned} \tag{2-4}$$

for $i = 1, \dots, N$ with initial condition $(y_c - y^{[j]})(0) = 0$. Defining the correction $d^{[j]} = y_c - y^{[j]}$ yields

$$\begin{aligned} d^{[j]}(t_i) = d^{[j]}(t_{i-1}) &+ \sum_{k=1}^{N} S_{ik}(f(y^{[j]}(t_k) + d^{[j]}(t_k)) - f(y_k^{[j]})) \\ &+ \sum_{k=1}^{N} S_{ik} f(y_k^{[j]}) - (y_i^{[j]} - y_{i-1}^{[j]}), \end{aligned}$$

which is not easier to solve than the original collocation problem (2-2) above. Different simple approximations of the middle integration term involving $d^{[j]}$, however, at least provide corrections that can be applied repeatedly to form a convergent stationary iteration.

Explicit SDC. Approximating the spectral integration term by the left-looking rectangular rule corresponding to the explicit Euler time-stepping scheme yields the explicit SDC correction

$$\begin{aligned} \delta_i^{[j]} = \delta_{i-1}^{[j]} &+ (t_i - t_{i-1})(f(y_{i-1}^{[j]} + \delta_{i-1}^{[j]}) - f(y_{i-1}^{[j]})) \\ &+ \sum_{k=1}^{N} S_{ik} f(y_k^{[j]}) - (y_i^{[j]} - y_{i-1}^{[j]}), \quad i = 1, \dots, N, \end{aligned} \tag{2-5}$$

suitable for nonstiff problems. The initial value is $\delta_0^{[j]} = 0$. Now, the interpolant $\delta^{[j]}$ is a polynomial approximation of the exact error function $d^{[j]}$. An improved approximation $y^{[j+1]}$ is then obtained as $y^{[j+1]} = y^{[j]} + \delta^{[j]}$. Note that the value $f(y_{i-1}^{[j]} + \delta_{i-1}^{[j]})$ appears again as $f(y_{i-1}^{[j+1]})$ in the next iteration, such that for each iteration only N right-hand side evaluations are required.

The expensive part in the explicit SDC method is usually the evaluation of the right-hand sides $f(y_i^{[c]})$. As mentioned above, an exact evaluation of the right-hand side $f(y_i^{[j]})$ is not necessary, because SDC iteration errors are already present due to the replacement of the spectral quadrature term by the rectangular rule. If approximate values $f_i^{[j]} \approx f(y_i^{[j]})$ can be computed faster, we can exploit the allowed inaccuracy for a reduction of the total computation effort.

It is clear that the evaluation error $f_i^{[j]} - f(y_i^{[j]})$ must be controlled in an appropriate way such as not to destroy convergence of the fixed-point scheme. We assume that for evaluation of $f(y_i^{[j]})$ we can prescribe a local absolute tolerance $\epsilon_i^{[j]}$ such that the computed value $f_i^{[j]}$ satisfies $\|f_i^{[j]} - f(y_i^{[j]})\|_Y \le \epsilon_i^{[j]}$.

As a consequence, the explicit SDC correction $\hat{\delta}^{[j]}$ for inexact right-hand sides $f_i^{[j]}$ is obtained as

$$\hat{\delta}_i^{[j]} = \hat{\delta}_{i-1}^{[j]} + (t_i - t_{i-1})(f_{i-1}^{[j+1]} - f_{i-1}^{[j]}) + \sum_{k=1}^{N} S_{ik} f_k^{[j]} - (y_i^{[j]} - y_{i-1}^{[j]}), \tag{2-6}$$

for $j = 0, \dots, J-1$ and $i = 1, \dots, N$ with $\hat{\delta}_0^{[j]} = 0$.

Implicit SDC. Assuming f is differentiable, linearizing f around $y_i^{[j]}$ and using the right-looking rectangular rule corresponds to the linearly implicit Euler scheme and leads to the implicit SDC correction

$$\delta_i^{[j]} = \delta_{i-1}^{[j]} + (t_i - t_{i-1}) f'(y_i^{[j]}) \delta_i^{[j]} + \sum_{k=1}^{N} S_{ik} f(y_k^{[j]}) - (y_i^{[j]} - y_{i-1}^{[j]}), \qquad i = 1, \dots, N, \tag{2-7}$$

suitable for stiff problems. As in the explicit case, N right-hand side evaluations are required, but additionally N evaluations of f' and N linear system solves with the matrices $I - (t_i - t_{i-1}) f'(y_i^{[j]})$.

Solving these systems, usually by an iterative solver, is often the expensive operation in the implicit SDC method. Early termination of the linear solver can reduce the computational effort significantly, but incurs a truncation error that must be controlled appropriately in terms of local tolerances $\epsilon_i^{[j]}$. Assuming the residuals $r_i^{[j]}$ are bounded by $\|r_i^{[j]}\|_Y \le \epsilon_i^j$, the implicit SDC correction $\hat{\delta}^{[j]}$ for inexact system solves is obtained as

$$(I - (t_i - t_{i-1}) f'(y_i^{[j]}))\hat{\delta}_i^{[j]} = \hat{\delta}_{i-1}^{[j]} + \sum_{k=1}^{N} S_{ik} f(y_k^{[j]}) - (y_i^{[j]} - y_{i-1}^{[j]}) + r_i^{[j]}, \qquad i = 1, \dots, N, \tag{2-8}$$

for $j = 0, \dots, J-1$ and $i = 1, \dots, N$ with $\hat{\delta}_0^{[j]} = 0$.

In both cases, the update $y^{[j]} \mapsto y^{[j+1]} := y^{[j]} + \hat{\delta}^{[j]}$ defines a parametrized fixed-point operator

$$\widehat{F} : Y^{N+1} \times \mathbb{R}_+^{N \times J+1} \times \mathbb{N} \to Y^{N+1}, \quad \widehat{F}(y^{[j]}; \epsilon, j) := y^{[j+1]},$$

with the exact limit case $F(y) := \widehat{F}(y; 0, 0)$.

For convergence analysis, we equip Y^{N+1} with a norm

$$\|y\| := \|[\|y_0\|_Y, \dots, \|y_N\|_Y]\|_p \tag{2-9}$$

in terms of the usual p-norm on $\mathbb{R}^{N+1}$ with $p \in [1, \infty]$ to be specified later. If F is Lipschitz continuous with constant $\rho < 1$, i.e.,

$$\|F(x) - F(y)\| \le \rho \|x - y\| \quad \text{for all } x, y \in Y^{N+1}$$

(which we will assume throughout the paper), Banach's fixed-point theorem yields q-linear convergence of the iteration to the unique collocation solution y_c independently of the initial iterate $y^{[0]}$. Note that the contraction property of F and hence the convergence of SDC depends on f, the collocation points t_i, the time step size T, and whether we use explicit or implicit SDC. For sufficiently small time steps, however, convergence is guaranteed if f is Lipschitz continuous.

Termination of the fixed-point iteration at iterate J can be based on either a fixed iteration count, resulting in a particular Runge–Kutta time-stepping scheme, or on an accuracy request of the form $\|y_c - y^{[J]}\| \le \mathrm{TOL}$. Given the contraction rate ρ, and assuming that $\|y_c - y^{[0]}\| > \mathrm{TOL}$, the number of exact iterations is then bounded by

$$J \le \left\lceil \frac{\log(\mathrm{TOL}/\|y_c - y^{[0]}\|)}{\log \rho} \right\rceil.$$

The choice of the initial iterate $y^{[0]}$ can not only have a significant impact on the number J of iterations needed to achieve the requested accuracy, but also on the properties of intermediate solutions. In particular for stiff problems, L-stability of intermediate solutions is obtained only if $y^{[0]}$ is computed by an L-stable basic scheme, e.g., implicit Euler, or special DIRK sweeps as proposed in [27]. For simplicity, however, we choose $y_i^{[0]} \equiv y_0$ in this paper.

Given the requirement of computing a final iterate $y^{[J]}$ satisfying the requested accuracy $\|y_c - y^{[J]}\| \le \mathrm{TOL}$, the immediate questions that arise are how to select the local tolerances $\epsilon_i^{[j]}$, and how many iterations to perform, in order to obtain the most efficient method. This question will be addressed in the following section.

3. A priori tolerance selection

Following the approach taken by Alfeld [1], an attractive choice of local tolerances $\epsilon_i^{[j]}$ and iteration count J is to minimize the overall computational effort $W(\epsilon, J)$ while bounding the final error $\|y^{[J]} - y_c\| \le \Phi(\epsilon, J)$:

$$\min_{J \in \mathbb{N},\, \epsilon \in \mathscr{E} \subset \mathbb{R}^{N \times J+1}} W(\epsilon, J) \quad \text{subject to } \Phi(\epsilon, J) \le \mathrm{TOL}. \tag{3-1}$$

Here, ϵ denotes the $N \times (J + 1)$ matrix of local tolerances $\epsilon_i^{[j]}$, restricted to an admissible set $\mathscr{E}$. We will consider different admissible sets in Sections 3C–3E below.

For this abstract framework to be useful, a *work model* W and an *error model* Φ are needed. These two building blocks will be established in the following two subsections.

Remark 3.1. Both the error model and the work model will involve quantities that are not explicitly known in actual computation. For the error model, these parameters are in particular the SDC contraction factor ρ, Lipschitz constants, and the initial iteration error $\|y^{[0]} - y_c\|$. The work models derived below rely on typical or asymptotic computational effort, which may not very well describe the actual effort spent on a concrete problem. Therefore, the efficiency predicted by the solution of the optimization problem (3-1) may not be reached.

Moreover, even if the assumptions are satisfied and the parameters entering into the error model are known, the estimates are not sharp. The actual error will typically be smaller than its bound, which means that the local tolerances derived from the error bound will be smaller than necessary, and the computational effort in turn higher than need be. Therefore, solving (3-1) provides only theoretically optimal local tolerances.

Nevertheless, the approximation results developed in the subsequent sections provide not only theoretical insight, but can also guide algorithmic choices, if computable estimates for the required parameters are available. For example, the SDC contraction factor ρ can be assumed not to change quickly over the integration time, such that the convergence on the previous time step could provide sufficient information. Lipschitz constants can at least be bounded from below by inspecting the evaluated right-hand sides. Inserting such estimates into the optimization problem can yield reasonable heuristics for choosing local tolerances in actual computations. Of course, such heuristics will need to be complemented by a posteriori error estimators and heuristics for updating parameter values in case the estimated actual error is larger than predicted. This, however, is beyond the scope of the present work.

3A. ***Error model.*** The error model bounds the final iteration error by $\Phi(\epsilon, J)$ in terms of the local tolerances $\epsilon_i^{[j]}$ and the iteration count J. Focusing on SDC as a fixed-point iteration, we estimate Φ in terms of inexact fixed-point iterations [1; 22]. Below we consider the convergence of

$$y^{[j+1]} = \widehat{F}(y^{[j]}; \epsilon, j), \quad j = 0, \dots, J-1, \tag{3-2}$$

to the fixed point y_c of F, and derive a bound on $\|y^{[J]} - y_c\|$ for given $y^{[0]}$, J, and ϵ.

First we establish estimates of how the errors bounded by $\epsilon_i^{[j]}$ are transported through the SDC sweep, and then address the complete iteration, both for explicit and implicit SDC schemes. For this we need some notation.

Definition 3.2. Let us assume there is a nonnegative function $L_f : \mathbb{R}_+ \to \mathbb{R}_+$ such that the right-hand side f satisfies the following Lipschitz-type conditions: for explicit SDC

$$\|\delta+\tau(f(y+\delta)-f(y))\|_Y \le L_f(\tau)\|\delta\|_Y \quad \text{for all } \tau\in\,]0,T] \text{ and } \delta, y\in Y \tag{3-3}$$

and for implicit SDC

$$\|(I-\tau f'(y))^{-1}\|_Y \le L_f(\tau) \quad \text{for all } \tau\in\,]0,T] \text{ and } y\in Y. \tag{3-4}$$

Then we define the invertible lower-triangular matrix $L\in\mathbb{R}^{N\times N}$ as

$$L_{im} := \begin{cases} \prod_{l=m}^{i-1} L_f(t_{l+1}-t_l), & m\le i,\\ 0, & \text{otherwise,}\end{cases}$$

and introduce $\|e\|_L := \|Le\|_p$ for $e\in\mathbb{R}^N$ and $\|\kappa\|_L := \max_{\|e\|_L=1}\|\kappa e\|_L = \|L\kappa L^{-1}\|_p$ for $\kappa\in\mathbb{R}^{N\times N}$.

Note that the nonstandard Lipschitz condition (3-3) follows from the standard Lipschitz condition on f (because $\|f(y+\delta)-f(y)\|_Y\le L_*\|\delta\|_Y$ implies $L_f(\tau)\le 1+\tau L_*$), but is weaker, in particular for slightly stiff systems. For example, for $f(y)=-y$ we obtain $L_f(\tau)=|1-\tau|\ll 1+\tau$ for $\tau\approx 1$. Nevertheless, the weaker condition (3-3) is sufficient for bounding the error transport through explicit SDC sweeps in the following theorem. Analogously, condition (3-4) describes the error transport through linearly implicit Euler sweeps in the implicit SDC method.

Explicit SDC. Now we derive error bounds, first for single SDC sweeps and then for the whole iteration.

Theorem 3.3. *Assume that the ODE's right-hand side satisfies the Lipschitz-like condition* (3-3). *Then, for* $\epsilon\in\mathbb{R}_+^{N\times J+1}$,

$$\|\widehat{F}(y;\epsilon,j)-F(y)\| \le \|\kappa(\epsilon^{[j]}+\epsilon^{[j+1]})+|S|\epsilon^{[j]}\|_L \tag{3-5}$$

holds for the explicit SDC iteration with $\kappa\in\mathbb{R}^{N\times N}$, $\kappa_{mk}:=\delta_{m-1,k}(t_m-t_{m-1})$, *where* $\delta_{m,k}$ *denotes the Kronecker delta.* $|S|\in\mathbb{R}^{N\times N}$ *denotes the entrywise absolute value of the integration matrix* S.

Proof. From (2-5) and (2-6) we obtain for the SDC corrections $\hat\delta_i$ the estimate

$$\begin{aligned}\|\widehat{F}(y;\epsilon,j)_i-F(y)_i\|_Y &= \|\hat\delta_i^{[j]}-\delta_i^{[j]}\|_Y\\ &\le \|\hat\delta_{i-1}^{[j]}-\delta_{i-1}^{[j]}+(t_i-t_{i-1})(f(y_{i-1}+\hat\delta_{i-1}^{[j]})-f(y_{i-1}+\delta_{i-1}^{[j]}))\|_Y\\ &\qquad + (t_i-t_{i-1})(\epsilon_{i-1}^{[j+1]}+\epsilon_{i-1}^{[j]})+\sum_{k=1}^{N}|S_{ik}|\epsilon_k^{[j]}\\ &\le L_f(t_i-t_{i-1})\|\hat\delta_{i-1}^{[j]}-\delta_{i-1}^{[j]}\|_Y+(\kappa(\epsilon^{[j]}+\epsilon^{[j+1]}))_i+(|S|\epsilon^{[j]})_i\end{aligned}$$

with $\hat{\delta}_0^{[j]} - \delta_0^{[j]} = 0$. By induction we obtain the discrete Gronwall result

$$\begin{aligned}
\|\hat{\delta}_i^{[j]} - \delta_i^{[j]}\|_Y &\le \sum_{m=1}^{i} \prod_{l=m}^{i-1} L_f(t_{l+1} - t_l)([\kappa(\epsilon^{[j]} + \epsilon^{[j+1]})]_m + (|S|\epsilon^{[j]})_m) \\
&= \sum_{m=1}^{i} L_{im}([\kappa(\epsilon^{[j]} + \epsilon^{[j+1]})]_m + (|S|\epsilon^{[j]})_m) \\
&= [L(\kappa(\epsilon^{[j]} + \epsilon^{[j+1]}) + |S|\epsilon^{[j]})]_i .
\end{aligned}$$

Taking the norm over $i = 1, \dots, N$ yields the claim (3-5). □

With the error bound (3-5) for a single inexact SDC sweep at hand, we are in the position to bound the final time error.

Theorem 3.4. *Let $y^{[0]} \in Y^N$ be given, and let $y^{[j+1]}$ be defined by*

$$y^{[j+1]} = \widehat{F}(y^{[j]}, \epsilon, j), \quad j = 0, \dots, J-1,$$

for some $J \in \mathbb{N}$ and some local tolerance matrix $\epsilon \in \mathbb{R}^{N \times J+1}$. Then

$$\|y^{[J]} - y_c\| \le \alpha \sum_{j=0}^{J-1} \rho^{J-1-j} \|\epsilon^{[j]}\|_L + \|\kappa\epsilon^{[J]}\|_L + \rho^J \|y^{[0]} - y_c\| =: \Phi(\epsilon, J) \quad (3\text{-}6)$$

holds with $\alpha = \|\kappa + |S|\|_L + \rho\|\kappa\|_L$ and κ and $|S|$ as defined in Theorem 3.3.

Proof. First we show the (slightly stronger) result

$$\|y^{[J]} - y_c\| \le \sum_{j=1}^{J} \rho^{J-j} \|\kappa(\epsilon^{[j-1]} + \epsilon^{[j]}) + |S|\epsilon^{[j-1]}\|_L + \rho^J \|y^{[0]} - y_c\| \quad (3\text{-}7)$$

by induction over J. The claim holds trivially for $J = 0$. Otherwise, we obtain

$$\begin{aligned}
\|y^{[J]} - y_c\| &= \|\widehat{F}(y^{[J-1]}; \epsilon, j) - F(y_c)\| \\
&\le \|\widehat{F}(y^{[J-1]}; \epsilon, J-1) - F(y^{[J-1]})\| + \|F(y^{[J-1]}) - F(y_c)\| \\
&\le \|\kappa(\epsilon^{[J-1]} + \epsilon^{[J]}) + |S|\epsilon^{[J-1]}\|_L + \rho\|y^{[J-1]} - y_c\| \\
&\le \|\kappa(\epsilon^{[J-1]} + \epsilon^{[J]}) + |S|\epsilon^{[J-1]}\|_L \\
&\quad + \rho\left(\sum_{j=1}^{J-1} \rho^{J-1-j} \|\kappa(\epsilon^{[j-1]} + \epsilon^{[j]}) + |S|\epsilon^{[j-1]}\|_L + \rho^{J-1}\|y^{[0]} - y_c\|\right),
\end{aligned}$$

which is just (3-7). Applying the triangle inequality and rearranging terms in the sum yields

$$\begin{aligned}\|y^{[J]}-y_c\| &\le \rho^{J-1}\|(\kappa+|S|)\epsilon^{[0]}\|_L+\sum_{j=1}^{J-1}\rho^{J-1-j}(\|(\kappa+|S|)\epsilon^{[j]}\|_L+\rho\|\kappa\epsilon^{[j]}\|_L)\\ &\qquad\qquad\qquad\qquad +\|\kappa\epsilon^{[J]}\|_L+\rho^J\|y^{[0]}-y_c\|\\ &\le \sum_{j=0}^{J-1}\rho^{J-1-j}\underbrace{(\|\kappa+|S|\|_L+\rho\|\kappa\|_L)}_{=\alpha}\|\epsilon^j\|_L+\|\kappa\epsilon^{[J]}\|_L+\rho^J\|y^{[0]}-y_c\|\end{aligned}$$

and thus the claim (3-6). □

Note that $\epsilon^{[J]}$ enters the error bound $\Phi(\epsilon, J)$ given in (3-6) in a different way than $\epsilon_i^{[j]}$ for $j < J$. This is due to the fact that all right-hand sides evaluated enter twice into the computation (see (2-6)) except for the very last sweep evaluations, which enter only once. This turns out to be quantitatively important in Section 4.

Implicit SDC. Error bounds for inexact implicit SDC follow the same line of argument as for the explicit method, but are slightly simpler.

Theorem 3.5. *Assume that the ODE's right-hand side satisfies the Lipschitz-like condition* (3-4). *Then, for* $\epsilon \in \mathbb{R}_+^{N\times J+1}$,

$$\|\widehat{F}(y;\epsilon,j)-F(y)\|\le\|\sigma\epsilon^{[j]}\|_L \tag{3-8}$$

holds for the implicit SDC iteration with $\sigma\in\mathbb{R}^{N\times N}$, $\sigma_{kk}=L_f(t_k-t_{k-1})$.

Proof. From (2-7) and (2-8) we obtain for the SDC corrections $\hat{\delta}_i$ the estimate

$$\begin{aligned}\|\widehat{F}(y;\epsilon,j)_i-F(y)_i\|_Y &\le \|(I+(t_i-t_{i-1}))^{-1}(\hat{\delta}_{i-1}^{[j]}-\delta_{i-1}^{[j]}+r_i^{[j]})\|_Y\\ &\le L_f(t_i-t_{i-1})(\|\hat{\delta}_{i-1}^{[j]}-\delta_{i-1}^{[j]}\|_Y+\epsilon_i^{[j]})\\ &= L_f(t_i-t_{i-1})\|\hat{\delta}_{i-1}^{[j]}-\delta_{i-1}^{[j]}\|_Y+(\sigma\epsilon^{[j]})_i\end{aligned}$$

with $\hat{\delta}_0^{[j]}-\delta_0^{[j]}=0$. As before, induction provides the discrete Gronwall result

$$\|\widehat{F}(y;\epsilon,j)_i-F(y)_i\|_Y\le[L\sigma\epsilon^{[j]}]_i$$

and hence the claim (3-8). □

With the error bound (3-8) for a single implicit inexact SDC sweep at hand, we are in the position to bound the final time error.

Theorem 3.6. *Let* $y^{[0]}\in Y^N$ *be given, and let* $y^{[j+1]}$ *be defined by*

$$y^{[j+1]}=\widehat{F}(y^{[j]};\epsilon,j),\quad j=0,\dots,J-1,$$

for some $J \in \mathbb{N}$ and some local tolerance matrix $\epsilon \in \mathbb{R}^{N \times J+1}$. Then

$$\|y^{[J]} - y_c\| \le \alpha \sum_{j=0}^{J-1} \rho^{J-1-j} \|\epsilon^{[j]}\|_L + \|\kappa \epsilon^{[J]}\|_L + \rho^J \|y^{[0]} - y_c\| =: \Phi(\epsilon, J) \quad (3\text{-}9)$$

holds with $\alpha = \|\sigma\|_L$ and $\kappa = 0$, where σ is defined in Theorem 3.5.

Proof. First we show the (slightly stronger) result

$$\|y^{[J]} - y_c\| \le \sum_{j=1}^{J} \rho^{J-j} \|\sigma \epsilon^{[j-1]}\|_L + \rho^J \|y^{[0]} - y_c\| \qquad (3\text{-}10)$$

by induction over J. The claim holds trivially for $J = 0$. Otherwise, we obtain as in the proof of Theorem 3.4, now using (3-8),

$$\begin{aligned}\|y^{[J]} - y_c\| &\le \|\widehat{F}(y^{[J-1]}; \epsilon, J-1) - F(y^{[J-1]})\| + \|F(y^{[J-1]}) - F(y_c)\| \\ &\le \|\sigma \epsilon^{[J-1]}\|_L + \rho \|y^{[J-1]} - y_c\|,\end{aligned}$$

which implies (3-10). An index shift in j is all it takes to obtain the claim (3-9). □

Note that the error bounds $\Phi(\epsilon, J)$ as given in (3-6) and (3-9) for explicit and implicit SDC, respectively, have identical structure, and differ only in the values of the parameters α and κ. This allows a uniform analytical treatment of both explicit and implicit schemes in the following sections.

Remark 3.7. The choice of collocation nodes t_i affects the error bound (3-6) in three ways. First, the substep sizes $t_{i+1} - t_i$ enter into L_{ki} and hence into $\|\cdot\|_L$. Second, the integration matrix S enters into the factor α, and third, the contraction factor ρ depends on the collocation nodes in a nontrivial and up to now not well understood way.

The error model Φ as defined in (3-6) is an upper bound of the inexact SDC iteration for arbitrary errors bounded by the local tolerances $\epsilon_i^{[j]}$, and hence also an upper bound for the error $\rho^J \|y^{[0]} - y_c\|$ of the exact SDC iteration. Consequently, meeting the accuracy requirement $\Phi(\epsilon, J) \le \mathrm{TOL}$ implies $\rho^J \|y^{[0]} - y_c\| \le \mathrm{TOL}$ and

$$J \ge J_{\min} := \frac{\log \mathrm{TOL} - \log \|y^{[0]} - y_c\|}{\log \rho}.$$

3B. ***Work models.*** Let us assume that the computational effort of evaluating $f_i^{[j]}$ (in explicit SDC) or of solving for $\hat{\delta}_i^{[j]}$ (in implicit SDC) is given in terms of the work $W_i^{[j]} : \mathbb{R}_+ \to \mathbb{R}_+$ as $W_i^{[j]}(\epsilon_i^{[j]})$. The total work to spend for J SDC iterations,

$$W(\epsilon) = \sum_{j=0}^{J} \sum_{i=1}^{N} W_i^{[j]}(\epsilon_i^{[j]}), \qquad (3\text{-}11)$$

is just the sum of all efforts. Hence, common positive factors can be neglected, as they will not affect the minimizer of the optimization problem (3-1) at all. Note that, for optimizing just ϵ with fixed J, additive terms in the work model can also be neglected.

First we will discuss a few prototypical work models that cover common sources of controlled inaccuracy.

Finite element discretization. If the computation of basic steps within the SDC sweeps involves a PDE solution realized by adaptive finite elements, the discretization error can be expected to be proportional to $n^{-1/d}$, where n is the number of grid points and d is the spatial dimension. Assuming the work to be proportional to the number of grid points, we obtain

$$W_i^{[j]}(\epsilon_i^{[j]}) := \frac{1}{d}(\epsilon_i^{[j]})^{-d}. \tag{3-12}$$

The arbitrary factor d^{-1} has been introduced for notational convenience only.

Of course, the asymptotic behavior $W_i^{[j]} \to 0$ for $\epsilon_i^{[j]} \to \infty$ is not realistic, as there is a fixed amount $W_{\min}$ of work necessary on the coarse grid. Thus, the work model is valid only for $\epsilon_i^{[j]} \le \epsilon_{\max} = (dW_{\min})^{-1/d}$. We will address this in Section 3F.

Truncation errors. Let us assume the basic step computation involves the solution of a linear equation system by a linearly converging iterative solver. This is usually the case in implicit SDC methods applied to PDE problems. Starting the iterative solver at zero, the residual after m iterations may be assumed to be bounded by $\|r_i^{[j]}\|_Y \le \rho_{\text{it}}^m R^{[j]}$, where $\rho_{\text{it}} < 1$ is the contraction rate of the linear solver and $R^{[j]} \sim \|\delta_i^{[j]}\|_Y$ the size of the initial residual. The number of iterations necessary to reach the local tolerance $\|r_i^{[j]}\|_Y \le \epsilon_i^{[j]}$ is expected to be

$$m \ge \frac{\log \epsilon_i^{[j]} - \log R^{[j]}}{\log \rho_{\text{it}}}.$$

If the outer SDC iteration converges linearly with unperturbed contraction factor ρ, an assumption that will be justified in (3-27), the initial residual is roughly $R^{[j]} \approx \rho^j \|y^{[0]} - y_c\|$, which leads to

$$W_i^{[j]}(\epsilon_i^{[j]}) := -\log \epsilon_i^{[j]} + \log\|y^{[0]} - y_c\| + j \log \rho. \tag{3-13}$$

Of course, a negative number of iterations cannot be realized, and therefore, this work model is limited to $\log \epsilon_i^{[j]} < \epsilon_{\max}^{[j]} = \rho^j \|y^{[0]} - y_c\|$.

Remark 3.8. Simplifying the work model (3-13) by ignoring the additive contribution $\log(c\text{TOL}) + (j-J)\log \rho$ is sufficient for optimizing the local tolerances ϵ, but affects optimizing the number J of iterations and renders work ratios $W(\epsilon)/W(\hat{\epsilon})$ for comparing different local tolerance choices ϵ and $\hat{\epsilon}$ meaningless.

Stochastic sampling. In case the right-hand side contains a high-dimensional integral to be evaluated by Monte Carlo sampling, the accuracy can be expected to be proportional to the inverse square root of the number of samples. The work proportional to the number of samples is then

$$W_i^{[j]}(\epsilon_i^{[j]}) := \tfrac{1}{2}(\epsilon_i^{[j]})^{-2},$$

just a special case of (3-12). Of course, as the error bound of Monte Carlo sampling is not strict, the error model from the previous section gives no guarantee in this case.

The prototypical work models presented here share a common structure. Minimization of the total work is based on derivatives of the work with respect to the local tolerances. Here we see that all three models satisfy

$$(W_i^{[j]})'(\epsilon_i^{[j]}) = (\epsilon_i^{[j]})^{-(d+1)},$$

with $d = 0$ for truncation of iterative solvers, $d = 2$ for Monte Carlo sampling, and d giving the spatial dimension in adaptive linear finite element computations. This will allow us to treat all work models uniformly in the work optimization.

Moreover, the work models exhibit some qualitative properties, which we conjecture to be general properties of plausible work models.

Definition 3.9. A *work model* is a family of strictly convex, positive, and monotonically decreasing functions $W_i^{[j]} :]0, (\epsilon_{\max})_i^{[j]}] \to \mathbb{R}_+$ mapping requested tolerances to the associated computational effort. The functions $W_i^{[j]}$ exhibit the barrier property $W_i^{[j]}(\epsilon) \to \infty$ for $\epsilon \to 0$.

The properties of $W_i^{[j]}$ are inherited by the total work W of (3-11), which is strictly convex and monotone.

3C. ***Fixed local tolerance.*** To begin with, we consider heuristic choices of the admissible set $\mathscr{E}$ of local tolerances. The simplest possibility is to take the same value $\epsilon_i^{[j]} \equiv \epsilon_{\text{fix}}$ for all right-hand side evaluations. This corresponds to a fixed *absolute* solver tolerance in inexact implicit SDC.

In this case, the error bound (3-6) reduces to

$$\|y^{[J]} - y_c\| \le \epsilon_{\text{fix}}\left(\alpha\|\mathbf{1}\|_L \frac{1-\rho^J}{1-\rho} + \|\kappa\mathbf{1}\|_L\right) + \rho^J\|y^{[0]} - y_c\|,$$

where $\mathbf{1} \in \mathbb{R}^N$ with $\mathbf{1}_i = 1$. Consequently,

$$\epsilon_{\text{fix}} = \min\left(\epsilon_{\max}, \frac{\text{TOL} - \rho^J\|y^{[0]} - y_c\|}{\alpha\|\mathbf{1}\|_L(1-\rho^J)/(1-\rho) + \|\kappa\mathbf{1}\|_L}\right) \tag{3-14}$$

provides the largest admissible choice, and hence the one that incurs the least computational effort, for given J. With $\epsilon_{\text{fix}}(J)$ fixed, what remains is to choose the number J of SDC sweeps such that the overall work is minimized. To this extent, we consider the slightly more restrictive but easier to analyze variant

$$\epsilon_{\text{fix}} = \min\left(\epsilon_{\max}, \frac{\text{TOL} - \rho^J \|y^{[0]} - y_c\|}{\alpha\|\mathbf{1}\|_L/(1-\rho) + \|\kappa\mathbf{1}\|_L}\right).$$

For the general work model (3-12), the total work is just $W = N(J+1)\epsilon_{\text{fix}}(J)^{-d}/d$. Assuming $\epsilon_{\text{fix}} < \epsilon_{\max}$ and eliminating constant factors, we need to minimize $W(J) \sim (J+1)/(\text{TOL} - \rho^J \|y^{[0]} - y_c\|)^d$. A simple analysis reveals that $W(J)$ is quasiconvex, such that there is exactly one minimizer in $]J_{\min}, \infty[$; see the Appendix. Unfortunately, no closed expression seems to exist, but a numerical computation is straightforward. Due to the quasiconvexity, the optimal $J \in \mathbb{N}$ is one of the neighboring integer values.

The local tolerance is bounded by $\epsilon_{\text{fix}} \le c\text{TOL}$ for some generic constant c independent of J and TOL. Consequently, the total work is at least

$$W \ge c(J_{\min} + 1)\text{TOL}^{-d} = c\left(\frac{\log(\text{TOL}/\|y^{[0]} - y_c\|)}{\log \rho} + 1\right)\text{TOL}^{-d}. \tag{3-15}$$

Apparently, a complexity of $\mathbb{O}(\text{TOL}^{-d})$ is unavoidable, as this is already required for a single right-hand side evaluation to the requested accuracy. The logarithmic factor in (3-15), however, appears to be suboptimal. As this corresponds to the number J of SDC sweeps, which, depending on the concrete problem, can easily exceed a factor of ten, the suboptimality may induce a significant inefficiency in actual computation. We will address this shortcoming in the following Sections 3D and 3E and investigate it numerically in Section 4.

For completeness we note that in the less interesting case $\epsilon_{\text{fix}} = \epsilon_{\max}$, J is determined by minimizing $W = N(J+1)\epsilon_{\max}^{-d}/d$ subject to

$$\text{TOL} \ge \epsilon_{\max}(\alpha\|\mathbf{1}\|_L/(1-\rho) + \|\kappa\mathbf{1}\|_L) + \rho^J \|y^{[0]} - y_c\| \ge \Phi(\epsilon_{\max}, J) \ge \|y^{[J]} - y_c\|,$$

i.e.,

$$J \ge (\log\rho)^{-1} \log \frac{\text{TOL} - \epsilon_{\max}(\alpha\|\mathbf{1}\|_L/(1-\rho) + \|\kappa\mathbf{1}\|_L)}{\|y^{[J]} - y_c\|}.$$

3D. ***Geometrically decreasing local tolerances.*** The next step is to exploit the fact that, due to the linear convergence of the SDC iteration, larger evaluation errors are acceptable in the early iterations, and to make the heuristic choice

$$(\epsilon_{\text{geo}})_i^{[j]} = \min(\epsilon_{\max}, \beta\rho^{\gamma j}) \quad \text{for some } \beta, \gamma > 0. \tag{3-16}$$

This has been considered in [5] for $\gamma = 1$ as an “adaptive strategy” and is closely related to evaluating implicit Euler steps up to a fixed *relative* precision in implicit SDC methods, as suggested in [16] or realized in [24] by a fixed number of multigrid V-cycles.

We will assume that γ is given and optimize β as we have done before with ϵ_{fix}. Ignoring the impact of $\epsilon_{\max}$, (3-6) results in the slightly stronger accuracy requirement

$$\|y^{[J]} - y_c\| \le \beta\biggl(\alpha\|\mathbf{1}\|_L \sum_{j=0}^{J-1} \rho^{J-1-j+\gamma j} + \rho^{\gamma J}\|\kappa\mathbf{1}\|_L\biggr) + \rho^J\|y^{[0]} - y_c\| \overset{!}{\le} \text{TOL}.$$

Note that this implies a convergence rate of $\|y^{[J]} - y_c\| = \mathbb{O}(\rho^{\min(1,\gamma)J})$. For $\gamma \ne 1$ (there is a continuous extension to $\gamma = 1$, though) we obtain

$$\beta \le \frac{\text{TOL} - \rho^J\|y^{[0]} - y_c\|}{\rho^{\gamma(J-1)}(\alpha\|\mathbf{1}\|_L(1-\rho^{(1-\gamma)J})/(1-\rho^{1-\gamma}) + \rho^{\gamma}\|\kappa\mathbf{1}\|_L)}. \tag{3-17}$$

The total work $W(\epsilon)$ is monotonically decreasing in β due to (3-16) and Definition 3.9, such that the work-minimization problem (3-1) is solved by equality in (3-17), and we obtain

$$(\epsilon_{\text{geo}})_i^{[j]} = \min\biggl(\epsilon_{\max}, \frac{\text{TOL} - \rho^J\|y^{[0]} - y_c\|}{\rho^{\gamma(J-1)}(\alpha\|\mathbf{1}\|_L(1-\rho^{(1-\gamma)J})/(1-\rho^{1-\gamma}) + \rho^{\gamma}\|\kappa\mathbf{1}\|_L)}\rho^{\gamma j}\biggr). \tag{3-18}$$

Of course, $\epsilon_{\text{fix}} = \lim_{\gamma\to 0}\epsilon_{\text{geo}}$ is recovered in the limit.

Optimizing the iteration count J for the generic work model (3-12) with $d > 0$, we minimize

$$W = N\beta^{-d}\sum_{j=0}^{J}\rho^{-d\gamma j}/d \sim \beta^{-d}\frac{1-\rho^{-d\gamma(J+1)}}{1-\rho^{-d\gamma}}.$$

We distinguish between $\gamma < 1$ and $\gamma > 1$. In the first case, we obtain

$$\frac{1-\rho^{(1-\gamma)J}}{1-\rho^{1-\gamma}} \le (1-\rho^{1-\gamma})^{-1} \quad \text{and thus} \quad \beta \ge \frac{\text{TOL} - \rho^J\|y^{[0]} - y_c\|}{\rho^{\gamma(J-1)}c}$$

for some $c > 0$ independent of J. Neglecting constant factors independent of J yields the upper bound

$$W \lesssim \left(\frac{\text{TOL} - \rho^J\|y^{[0]} - y_c\|}{\rho^{\gamma(J-1)}}\right)^{-d}\rho^{-d\gamma(J+1)}$$

decreasing monotonically with J towards $\lim_{J\to\infty} W \lesssim \text{TOL}^{-d}$. Compared to (3-15), the complexity to reach the requested tolerance is improved, independently of γ, from $\mathbb{O}(\text{TOL}^{-d}|\log\text{TOL}|)$ to $\mathbb{O}(\text{TOL}^{-d})$. In the next section we will see that this complexity is indeed optimal, but the constants can be improved further by considering a larger admissible set $\mathscr{E}$.

In the second case $\gamma > 1$, we obtain the upper bound

$$W \lesssim \left(\frac{\mathrm{TOL} - \rho^J \|y^{[0]} - y_c\|}{c\rho^J + \rho^{\gamma(J-1)}} \right)^{-d} \rho^{-d\gamma(J+1)} \sim \left(\frac{c\rho^{(1-\gamma)J} + b}{\mathrm{TOL} - \rho^J \|y^{[0]} - y_c\|} \right)^d$$

for some generic constants b and c independent of J and TOL. Inserting $J \geq \log(\mathrm{TOL}/\|y^{[0]} - y_c\|)/\log\rho$ reveals a complexity of $\mathcal{O}(\mathrm{TOL}^{-\gamma d})$, indeed worse than the fixed choice $\epsilon_i^{[j]} \equiv \epsilon_{\mathrm{fix}}$ before. As a certain number of SDC iterations have to be performed with sufficient accuracy, increasing the accuracy too quickly is a waste of resources. Fortunately, a fixed relative accuracy will always lead to $\gamma \leq 1$.

3E. ***Variable local tolerances.*** Finally, let us consider the most general admissible set $\mathscr{E} = \{\epsilon \in \mathbb{R}_+^{N\times J+1} \mid \epsilon_i^{[j]} \leq \epsilon_{\max}\}$ in greater detail than we have treated the heuristic choices. Again, we will proceed in two steps, first assuming J to be given, optimizing only the local tolerances ϵ, and considering the integer variable J of the mixed integer program later on.

We obtain the nonlinear program

$$\min_{\epsilon\in\mathbb{R}_+^{N\times J+1}} W(\epsilon) \quad \text{subject to } \Phi(\epsilon, J) \leq \mathrm{TOL},\ \epsilon \leq \epsilon_{\max}. \tag{3-19}$$

From the properties of W and Φ, we immediately obtain the following result.

Theorem 3.10. *If $\rho^J \|y^{[0]} - y_c\| < \mathrm{TOL}$, i.e., the exact SDC iteration converges to the given tolerance, the optimization problem* (3-19) *has a unique solution $\epsilon(y^{[0]}, J)$. In the generic case $\epsilon_i^{[j]} < (\epsilon_{\max})_i^{[j]}$ for some i and j, i.e., if not all of the local tolerance constraints are active, the accuracy constraint is active, i.e., $\Phi(\epsilon(y^{[0]}, J), J) = \mathrm{TOL}$.*

Proof. From (3-6) it is apparent that sufficiently small values $\epsilon_i^{[j]} > 0$ lead to

$$\alpha \sum_{j=0}^{J-1} \rho^{J-1-j} \|\epsilon^{[j]}\|_L + \|\kappa\epsilon^{[J]}\|_L \leq \mathrm{TOL} - \rho^J \|y^{[0]} - y_c\|,$$

such that the admissible set is nonempty. Strict convexity of W and convexity of Φ imply uniqueness of a solution. Strict convexity and monotonicity of W imply its strict monotonicity, and hence, the constraint must be active unless all local tolerances are actively bound by $\epsilon \leq \epsilon_{\max}$. □

The activity of the accuracy constraint in the generic case means that, as expected, no effort is wasted on reducing the error below the requested tolerance.

We may reasonably expect the local tolerances to decrease monotonically. This is indeed true in general, as the following result shows.

Theorem 3.11. *Assume that $\rho \in (0,1)$, $J \in \mathbb{N}$, and $\mathrm{TOL} \in \mathbb{R}_+$ are given constants. Let the local tolerance matrix ϵ be the minimizer of* (3-19). *Then $\|\epsilon^{[j]}\|_L \le \|\epsilon^{[j-1]}\|_L$ holds for all $j = 1, \dots, J-1$.*

For the norm exponent $p = 1$ in (2-9), *componentwise monotonicity holds as well, i.e., $\epsilon_i^{[j]} \le \epsilon_i^{[j-1]}$ holds for all i and $j < J$.*

Proof. Let $\tilde\epsilon$ be an admissible point for (3-19) with $\|\tilde\epsilon^{[k_1]}\|_L < \|\tilde\epsilon^{[k_2]}\|_L$ for some $1 \le k_1 < k_2 < J$. Then we consider ϵ with $\epsilon^{[j]} = \tilde\epsilon^{[j]}$ except for $\epsilon^{[k_2]} = \tilde\epsilon^{[k_1]}$ and $\epsilon^{[k_1]} = \tilde\epsilon^{[k_2]}$. Obviously, $W(\epsilon) = W(\tilde\epsilon)$.

The error bound (3-6), however, is reduced:

$$\begin{aligned}
\Phi(\tilde\epsilon, J) - \Phi(\epsilon, J)& \\
&= \alpha(\rho^{J-1-k_1}(\|\tilde\epsilon^{[k_1]}\|_L - \|\epsilon^{[k_1]}\|_L) + \rho^{J-1-k_2}(\|\tilde\epsilon^{[k_2]}\|_L - \|\epsilon^{[k_2]}\|_L))\\
&= \alpha(\rho^{J-1-k_1}(\|\tilde\epsilon^{[k_1]}\|_L - \|\tilde\epsilon^{[k_2]}\|_L) + \rho^{J-1-k_2}(\|\tilde\epsilon^{[k_2]}\|_L - \|\tilde\epsilon^{[k_1]}\|_L))\\
&= \alpha(\rho^{J-1-k_1} - \rho^{J-1-k_2})(\|\tilde\epsilon^{[k_1]}\|_L - \|\tilde\epsilon^{[k_2]}\|_L) > 0,
\end{aligned}$$

as $\alpha > 0$ and the other two factors on the last line are negative. Since $\Phi(\epsilon, J) < \Phi(\tilde\epsilon, J) \le \mathrm{TOL}$, ϵ is feasible. The constraint, however, is inactive, such that ϵ cannot be the minimizer $\epsilon(y^{[0]}, J)$. We conclude that

$$W(\epsilon(y^{[0]}, J)) < W(\epsilon) = W(\tilde\epsilon),$$

such that $\tilde\epsilon \ne \epsilon(y^{[0]}, J)$. The same line of argument holds for $p = 1$ and componentwise monotonicity, where however ϵ is constructed such that only $\epsilon_i^{[k_1]}$ and $\epsilon_i^{[k_2]}$ are swapped. □

Below the necessary and, due to convexity, also sufficient conditions for the solution of the constrained optimization problem are derived for $p < \infty$.

Theorem 3.12. *Let the norm exponent p be finite. Assume $\rho^J \|y^{[0]} - y_c\| < \mathrm{TOL}$ and $W_i^{[j]} \in C^1(0, \infty)$. Then $\epsilon \in \mathbb{R}_+^{N \times J+1}$ solves* (3-19), *if and only if there exist multipliers $\mu \in \mathbb{R}$ and $\eta \in \mathbb{R}^{N \times J+1}$ such that*

$$\begin{aligned}
&(W_i^{[j]})'(\epsilon_i^{[j]}) + \mu\alpha\rho^{J-1-j}\|\epsilon^{[j]}\|_L^{1-p}\sum_{k=1}^{N}(L\epsilon^{[j]})_k^{p-1}L_{ki} + \eta_i^{[j]} = 0,\\
&\qquad\qquad\qquad\qquad j = 0, 1, \dots, J-1,\\
&(W_i^{[J]})'(\epsilon_i^{[J]}) + \mu\|\kappa\epsilon^{[J]}\|_L^{1-p}\sum_{k=1}^{N}(L\kappa\epsilon^{[J]})^{p-1}(L\kappa)_{ki} + \eta_i^{[J]} = 0,\\
&\qquad\qquad (\mathrm{TOL} - \Phi(\epsilon, J))\mu = 0, \quad \mu \ge 0,\\
&\qquad\qquad (\epsilon_{\max} - \epsilon) : \eta = 0, \quad \eta \ge 0.
\end{aligned} \tag{3-20}$$

Here, $\epsilon : \eta$ denotes contraction or Frobenius product, and we use the convention $0^0 := 0$ (for $\kappa = 0$ and $p = 1$ this expression can formally arise).

Proof. The necessary and also sufficient condition for optimality of ϵ is the stationarity of the Lagrangian

$$L(\epsilon, \mu, \eta) = W(\epsilon, J) + \mu(\Phi(\epsilon, J) - \mathrm{TOL}) + \eta : (\epsilon_{\max} - \epsilon)$$

for some multiplier $\mu \in \mathbb{R}$ and $\eta \in \mathbb{R}^{N\times J+1}$; see, e.g., [21]. According to the (structurally identical) error bounds (3-6) and (3-9), and the total work (3-11), its partial derivatives are just the expressions in (3-20). □

At this point, the unique minimizer $\epsilon(y^{[0]}, J)$ of the convex program (3-19) can in principle be computed numerically. For an exponent $p = 1$ in the norm definition (2-9), however, explicit analytical expressions can easily be derived due to (3-6) reducing to

$$\begin{aligned}\Phi(\epsilon, J) &= \alpha \sum_{j=0}^{J-1} \rho^{J-1-j} \sum_{k=1}^{N}\sum_{i=1}^{N} L_{ki}\epsilon_i^{[j]} + \sum_{k=1}^{N}\sum_{i=1}^{N} (L\kappa)_{ki}\epsilon_i^{[J]} + \rho^J \|y^{[0]} - y_c\| \\ &= q : \epsilon + \rho^J \|y^{[0]} - y_c\| \end{aligned} \tag{3-21}$$

with

$$q_i^{[j]} = \begin{cases} \alpha\rho^{J-1-j} \sum_{k=1}^{N} L_{ki}, & j < J, \\ \sum_{k=1}^{N} (L\kappa)_{ki}, & j = J. \end{cases} \tag{3-22}$$

Then, the first-order necessary conditions (3-20) assume the particularly simple form

$$(W_i^{[j]})'(\epsilon_i^{[j]}) + \mu q_i^{[j]} + \eta_i^{[j]} = 0. \tag{3-23}$$

Below we will derive the analytical structure of solutions for $p = 1$ and different work models, which also sheds some more light on the structure of the solution as well as on the achieved efficiency. The following theorem applies to all work models from Section 3B, with $d = 0$ for iterative solvers and $d = 2$ for stochastic sampling.

Theorem 3.13. *Let $p = 1$ and $(W_i^{[j]})'(\epsilon_i^{[j]}) = -(\epsilon_i^{[j]})^{-(d+1)}$. Then there is $\mu > 0$ such that the solution $\epsilon = \epsilon(y^{[0]}, J)$ of* (3-19) *is given by*

$$\epsilon_i^{[j]} = \begin{cases} (\epsilon_{\max})_i^{[j]}, & q_i^{[j]} = 0, \\ \min((\epsilon_{\max})_i^{[j]}, (\mu q_i^{[j]})^{-1/(d+1)}), & \textit{otherwise}, \end{cases} \tag{3-24}$$

with $q_i^{[j]}$ given in (3-22). *Locally unconstrained tolerances $\epsilon_i^{[j]} < (\epsilon_{\max})_i^{[j]}$ decrease linearly up to $j = J - 1$:*

$$\epsilon_i^{[j]} \sim \rho^{j/(d+1)}. \tag{3-25}$$

Proof. From the necessary condition (3-23) we obtain a multiplier $\hat{\mu} \geq 0$. If $\hat{\mu} = 0$, then $\eta_i^{[j]} > 0$ for all i and j due to $(W_i^{[j]})' < 0$, which implies $\epsilon = \epsilon_{\max}$ via complementarity in (3-20). Choosing $\mu > 0$ sufficiently small verifies the claim (3-24).

Otherwise we choose $\mu = \hat{\mu} > 0$ and obtain

$$\epsilon_i^{[j]} = (\mu q_i^{[j]} + \eta_i^{[j]})^{-1/(d+1)} \tag{3-26}$$

from (3-23). In case $\epsilon_i^{[j]} < (\epsilon_{\max})_i^{[j]}$, $\eta_i^{[j]} = 0$ holds due to complementarity in (3-20), such that the claim (3-24) is satisfied. For $j < J$, the definition (3-22) of $q_i^{[j]}$ implies

$$\epsilon_i^{[j]} = (\mu\alpha\rho^{J-1-j} \sum_{k=1}^{N} L_{ki})^{-1/(d+1)} \sim \rho^{j/(d+1)}$$

and hence the geometric decrease (3-25).

In case $\epsilon_i^{[j]} = (\epsilon_{\max})_i^{[j]}$ and $q_i^{[j]} \neq 0$, (3-26) implies

$$(\mu q_i^{[j]})^{-1} = (((\epsilon_{\max})_i^{[j]})^{-(d+1)} - \eta)^{-1} \geq ((\epsilon_{\max})_i^{[j]})^{d+1}$$

and hence the claim (3-24). □

The result (3-25) reveals that the heuristic of geometrically decreasing local tolerances is indeed of optimal complexity, at least for $\gamma < 1$, and now theoretically justified. Beyond that, an optimal value of $\gamma = (d+1)^{-1}$ and different accuracies for the collocation points are provided. We will see in Section 4 that the last issue can have a nonnegligible impact on the computational effort. Moreover, the result (3-25) shows that the contraction rate of optimal inexact SDC iterations depends on the work model: ρ for the truncation of linearly convergent iterations and $\rho^{1/(d+1)}$ for linear finite element solutions. The latter convergence is actually slower than the exact SDC iteration. This is a consequence of the different work required to reduce the error: while a reduction of the SDC iteration error is relatively cheap, reducing finite element discretization errors is rather expensive. An optimal tolerance selection therefore assigns a larger portion of the total error to the discretization and has to ensure that the SDC iteration error is by a certain factor smaller than the discretization error.

As expected, the geometric decrease (3-25) translates directly into linear convergence of the inexact SDC iteration:

Corollary 3.14. *If $\epsilon < \epsilon_{\max}$ holds, there is some c independent of j (though it depends on J) such that*

$$\|y^{[j]} - y_c\| \leq c\rho^{j/(d+1)}. \tag{3-27}$$

Proof. The result (3-25) yields

$$\begin{aligned}\|y^{[j]} - y_c\| &\le \alpha \sum_{k=0}^{j-1} \rho^{j-1-k} \|\epsilon^{[k]}\|_L + \|\kappa\epsilon^{[j]}\|_L + \rho^j \|y^{[0]} - y_c\| \\ &\le c\left(\sum_{k=0}^{j-1} \rho^{j-1-k}\rho^{k/(d+1)} + \rho^{j/(d+1)}\right) + \rho^j \|y^{[0]} - y_c\| \\ &\le c\rho^{j/(d+1)} \end{aligned} \tag{3-28}$$

and hence the claim. □

For $d = 0$, this linear convergence justifies the contraction rate assumed in defining the work model (3-13) for iterative solvers.

Let us state two more observations. First, it pays off to treat the final local tolerances $\epsilon_i^{[J]}$ separately in Theorem 3.4: now $\epsilon_i^{[J]} > \epsilon_i^{[J-1]}$ holds instead of $\epsilon_i^{[J]} = \rho\epsilon_i^{[J-1]}$. Thus, the effort for the otherwise greatest expense, due to having the most accurate right-hand side evaluations, is reduced, as illustrated in Figure 1. Similarly, for implicit SDC schemes with $\kappa = 0$ defined in Theorem 3.6, $q_i^{[J]} = 0$ holds, which implies $\epsilon_i^{[J]} = (\epsilon_{\max})_i^{[J]}$.

Second, (3-24) is monotone in μ, such that the actual value of μ and in turn ϵ is easily computed numerically by solving $\Phi(\epsilon, J) = \text{TOL}$. In case $\epsilon_{\max}$ is sufficiently large such that $\epsilon < \epsilon_{\max}$ holds, combining (3-24), (3-22), and (3-21) yields an explicit expression

$$\epsilon_i^{[j]} = \frac{\text{TOL} - \rho^J \|y^{[0]} - y_c\|}{\sum_{j=0}^{J} \sum_{i=0}^{N} (q_i^{[j]})^{d/(d+1)}} (q_i^{[j]})^{-1/(d+1)}. \tag{3-29}$$

3F. ***Iteration count optimization.*** As in the case of uniform local tolerances, the number J of inexact SDC iterations has to be selected in order to minimize the total work. For the generic work model (3-12), and stripping it of common factors and additive terms, we obtain with Theorem 3.13

$$W(J) = \sum_{j=0}^{J} \sum_{i=1}^{N} (\epsilon_i^{[j]})^{-d} = \sum_{j=0}^{J} \sum_{i=1}^{N} (\mu q_i^{[j]})^{d/(d+1)}$$

as long as $\epsilon_i^{[j]} < (\epsilon_{\max})_i^{[j]}$ for all i and j. Inserting the definition (3-22) of $q_i^{[j]}$ and neglecting constant factors independent of J and N yields

$$\begin{aligned} W \le \sum_{j=0}^{J} \sum_{i=1}^{N} \left(\mu\alpha\rho^{J-1-j} \sum_{k=1}^{N} L_{ki}\right)^{d/(d+1)} &\sim N\mu^{d/(d+1)} \sum_{j=0}^{J} \rho^{(d/(d+1))(J-1-j)} \\ &\sim N\mu^{d/(d+1)} \frac{1-\rho^{d(J+1)/(d+1)}}{1-\rho^{d/(d+1)}} \end{aligned} \tag{3-30}$$

as long as $\max_i(t_i - t_{i-1}) \le c/N$ for some constant c.

The multiplier μ is obtained from $\Phi(\epsilon, J) = \mathrm{TOL}$ with $\epsilon_i^{[j]} = (\mu q_i^{[j]})^{-1/(d+1)}$. We obtain

$$\begin{aligned}
\mathrm{TOL} &= \alpha \sum_{j=0}^{J-1} \rho^{J-1-j} \|\epsilon^{[j]}\|_L + \|\kappa\epsilon^{[J]}\|_L + \rho^J \|y^0 - y_c\| \\
&= \mu^{-1/(d+1)} \biggl(\alpha \sum_{j=0}^{J-1} \rho^{J-1-j} \|(q^{[j]})^{-1/(d+1)}\|_L + \|\kappa (q^{[J]})^{-1/(d+1)}\|_L\biggr) + \rho^J \|y^0 - y_c\| \\
&= \mu^{-1/(d+1)} \biggl(a \sum_{j=0}^{J-1} \rho^{(d/(d+1))(J-1-j)} + b\biggr) + \rho^J \|y^0 - y_c\| \\
&= \mu^{-1/(d+1)} \biggl(a \frac{1-\rho^{dJ/(d+1)}}{1-\rho^{d/(d+1)}} + b\biggr) + \rho^J \|y^0 - y_c\|
\end{aligned}$$

with constants $a = \alpha \|(\alpha \sum_{k=1}^{N} L_{ki})^{-1/(d+1)}\|_L$ and $b = \|\kappa (q^{[J]})^{-1/(d+1)}\|_L$ independent of J. Consequently,

$$\mu^{d/(d+1)} = \biggl(\frac{a(1-\rho^{dJ/(d+1)})/(1-\rho^{d/(d+1)}) + b}{\mathrm{TOL} - \rho^J \|y^0 - y_c\|}\biggr)^d$$

holds. Entering this into the work bound (3-30) yields

$$W \lesssim N \biggl(\frac{a(1-\rho^{dJ/(d+1)})/(1-\rho^{d/(d+1)}) + b}{\mathrm{TOL} - \rho^J \|y^0 - y_c\|}\biggr)^d \frac{1-\rho^{d(J+1)/(d+1)}}{1-\rho^{d/(d+1)}}.$$

Replacing $1 - \rho^{dJ/(d+1)}$ by 1 and neglecting constant factors independent of J and TOL provides the upper bound

$$W \lesssim N(\mathrm{TOL} - \rho^J \|y^0 - y_c\|)^{-d}. \tag{3-31}$$

The upper bound (3-31) is monotonically decreasing and suggests choosing J as large as possible. In the limit $J \to \infty$, the total work is bounded by

$$W \lesssim N \mathrm{TOL}^{-d}. \tag{3-32}$$

Compared to the work bound (3-15) for uniform local tolerances, the logarithmic factor $\log \mathrm{TOL}$ is missing, which yields the optimal complexity of evaluating N steps of the basic Euler scheme up to the requested tolerance.

Remark 3.15. The result (3-32) suggests that inexact explicit SDC methods might be able to reach or even surpass the efficiency of standard explicit Runge–Kutta methods.

However, the practical bound $\epsilon \le \epsilon_{\max}$ induces a lower bound $W_i^j \ge W_{\min}$ on the work per iteration, and hence, the total work $W(J)$ grows linearly with J

for $J \to \infty$. This contradicts the asymptotic work bound (3-32), which means that the assumption $\epsilon < \epsilon_{\max}$ used to derive (3-32) can only hold up to some finite iteration count. As closed expressions for a global minimizer of $W(J)$ when taking the local tolerance constraint $\epsilon \le \epsilon_{\max}$ into account are hard to get, a heuristic selection of J appears to be most promising in practice. The convexity of (3-31) and linear growth of W for large J suggest that we may select J as

$$J = \min\{j \in \mathbb{N} \mid W(j+1) > W(j)\}.$$

4. Numerical examples

Here we will illustrate and compare the effectiveness of the inexact SDC strategies worked out above. First, the properties of the strategies will be explored using a simple academic test problem. Then, inexactness due to iterative solvers and Monte Carlo sampling are considered with the heat equation and a molecular dynamics example, respectively.

4A. *An illustrative example.*

Problem setup. As a particularly simple example that allows a detailed investigation of the theoretical predictions, we consider the harmonic oscillator

$$\begin{aligned}\dot{u} &= v,\\ \dot{v} &= -u,\end{aligned}$$

with initial values $u_0 = 0$ and $v_0 = 1$, on the time interval $[0, \pi]$ subdivided into n equidistant time steps. The Lipschitz constant of the right-hand side is $L_* = 1$, and we estimate $L_f(\tau) = 1 + \tau$ using the triangle inequality. We use N Gauss–Legendre collocation points in each of the n time steps. The collocation error e_c at final time π can easily be obtained by comparing the result with the exact solution $u(t) = \sin(t)$, $v(t) = \cos(t)$. The contraction rate ρ of the exact SDC iteration is estimated numerically, and is virtually independent of the actual time t.

Exact right-hand side evaluation is of course straightforward, such that artificial inexactness and associated computational work are quite arbitrary. Here we use normally distributed random additive errors and the generic work model (3-12) with parameter d unless otherwise stated.

Aiming at a final time error comparable to the collocation error, we choose a tolerance $\mathrm{TOL} = e_c/\sqrt{n}$ for each time step, based on the assumption that the random errors of each time step simply add up, and yield a standard deviation of the final result of $\sqrt{n}\mathrm{TOL}$. With this setting, the quantities entering into the computation of the local tolerances ϵ are the same for all time steps. Unless otherwise stated $N = 3$ is used throughout, such that the collocation scheme is of order 6.

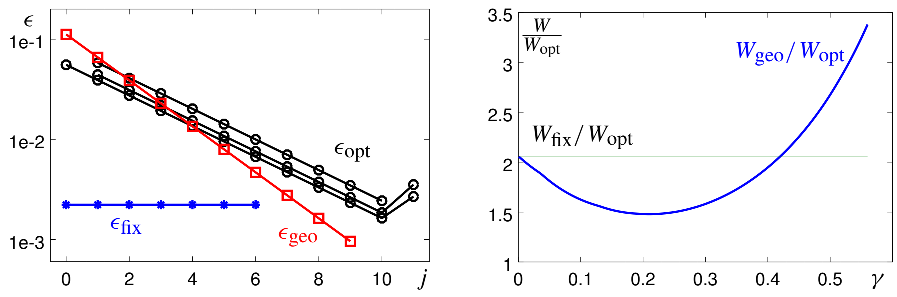


Figure 1. Left: exemplary local tolerances versus iteration number j for the different admissible design sets: fixed (ϵ_{fix}, stars), geometrically decreasing (ϵ_{geo}, squares, $\gamma = 0.5$), and variable (ϵ_{opt}, circles). Here $n = 2$ steps have been used to define the problem data $\rho = 0.35$, TOL $= 0.05$. Right: relative work $W_{\text{geo}}/W_{\text{opt}}$ for geometrically decreasing local tolerances versus the exponent γ. For larger γ, the work grows exponentially. The horizontal line denotes the relative work $W_{\text{fix}}/W_{\text{opt}}$ for fixed local tolerances.

Theoretical predictions. Let us first investigate the structure of local tolerances and the predicted efficiency gain in different situations.

To begin, we fix the iteration count J and time step T and compare local tolerances ϵ_{fix}, ϵ_{geo}, and ϵ_{opt} as given by the three considered strategies in (3-14), (3-18), and (3-24), respectively. Exemplary values for TOL $= 0.05$, $J = 11$, $\epsilon_{\max} = \infty$, $n = 2$, and estimated $\|y^{[0]} - y_c\| \approx 2.4$ are shown in Figure 1, left, versus the iteration number j. For the geometrically decreasing local tolerances, an exponent $\gamma = \frac{1}{2}$ has been chosen arbitrarily, but less than one due to the worse computational complexity for $\gamma > 1$; see Section 3D. For optimal variable tolerances, ϵ_{opt} has been obtained via (3-29). Clearly visible is the slow geometric decrease of the optimal variable local tolerances $\epsilon_{\text{opt}}^{[j]}$ with an order $\rho^{j/3}$, even slower than the explicitly chosen geometrical decrease $\rho^{\gamma j}$ with $\gamma = \frac{1}{2}$. The relative predicted work is $W_{\text{fix}}/W_{\text{opt}} = 2.06$ and $W_{\text{geo}}/W_{\text{opt}} = 2.67$. Somewhat surprisingly, exploiting the linear convergence of the SDC iteration does not necessarily pay off compared to a fixed accuracy, depending on the chosen parameter γ. The variable local tolerances approach achieves its low work by (i) choosing the appropriate decrease rate $\gamma = 1/(d+1)$, (ii) allowing for larger errors in later collocation points with less global impact, and (iii) imposing less restrictive requirements on the final sweep according to the definition (3-22) of $q_i^{[j]}$. The latter two aspects make up a reduction of work by a factor of 1.67 compared to the geometrically decreasing local tolerances with $\gamma = 1/(d+1)$. The relative work for different values of γ is shown in Figure 1, right, where the predicted total work induced by geometrically decreasing tolerances is plotted over the exponent γ. The optimum with a relative work of 1.48 is attained around $\gamma = 0.21$, even less than $1/(d+1)$. This can be attributed to avoiding high costs in the very last sweep, where high accuracy is actually not necessary, while ensuring sufficient accuracy in the next to last sweep.

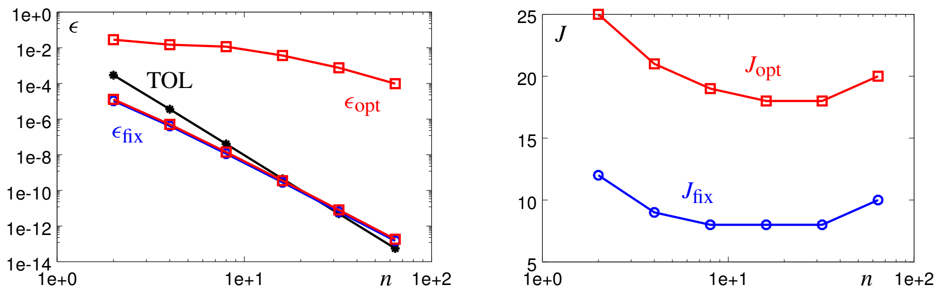


Figure 2. Left: local tolerances ϵ for the inexact SDC iterations versus number n of time steps. For optimal variable local tolerances (ϵ_{opt}, squares), the range between minimal and maximal local tolerance is shown. The requested tolerance TOL is shown with stars, the fixed local tolerance ϵ_{fix} with circles. Right: optimal number J of inexact SDC iterations versus number n of time steps for optimal variable (J_{opt}, squares) and fixed (J_{fix}, circles) local tolerances.

Next we look into the dependence of local tolerances and optimal iteration counts on the time step size T. Let us consider tolerances $\text{TOL} = e_c/\sqrt{n}$ depending on the time step size π/n. As shown in Figure 2, left, they decrease as $n^{-2N-1/2}$ according to the sixth-order collocation error and the error accumulation of order $\frac{1}{2}$. As expected, the fixed local tolerance ϵ_{fix} and the minimal variable local tolerance $\min_{i,j} \epsilon_i^{[j]}$ stay very close to each other and also close to TOL, but decrease roughly one order slower. This is due to $\alpha, \kappa = \mathbb{O}(t_N) = \mathbb{O}(n^{-1})$, and leads to the surprising fact that for small time steps the allowed evaluation error can be larger than the requested tolerance. Obviously, the heuristic choice $\epsilon_{\text{fix}} = c\text{TOL}$ for some fixed $c < 1$ is suboptimal for small time steps.

As intended, the maximal local tolerance, encountered in the very first inexact SDC sweep, is much larger than the minimal one, which is the basis for the envisioned performance gain. It also decreases much slower than the step tolerance TOL due to the fact that $\rho \to 0$ for $t_N \to 0$.

The optimal number of sweeps shown in Figure 2, right, is rather different for fixed and variable local tolerances, with a factor of two in between. This is due to the intended slower contraction rate in (3-24) compared to (3-14). As each sweep increases the order of the SDC integrator by one, and the tolerance TOL is of order $n^{-6.5}$, we expect at least seven sweeps to be necessary. This is nicely reflected by the fixed local tolerance scheme resorting to an optimal value of eight sweeps over a range of step sizes. For larger step sizes, the growth in the contraction rate ρ destroys this asymptotic property.

Finally, we take a look at the predicted efficiency gain over the simple fixed local tolerance strategy in dependence on time step size and overall tolerance. The total work per step induced by the choices of local tolerances is shown in Figure 3. The ratio of more than 10^{15} of computational effort between $n = 2$ and $n = 64$

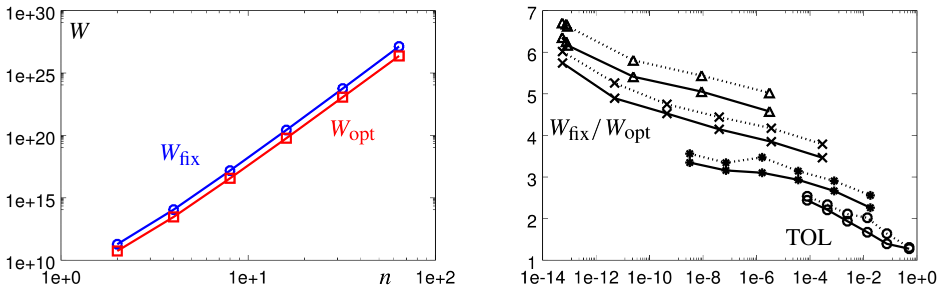


Figure 3. Left: total work per time step for fixed (W_{fix}, circles) and variable (W_{opt}, squares) local tolerances versus the number of time steps. Right: ratio of total work of fixed and variable local tolerances versus the requested tolerance TOL, for work model parameter $d = 2$ (solid lines) and $d = 3$ (dotted lines), number of collocation points $N \in \{1, 2, 3, 4\}$ (circles, stars, crosses, triangles), and different number n of time steps.

is due to the high accuracy of the Gauss collocation and the slow convergence of linear finite elements assumed with $d = 2$. According to (3-12), the work is of order $\mathbb{O}(\epsilon^{-d}) = \mathbb{O}(n^{d(2N-1/2)})$, which amounts here to a growth of n^{11}. Obviously, the high accuracies reached in the model problem are unrealistic in practical finite element computation. The ratio between the work for fixed and local tolerances shown in detail in Figure 3, right, adheres to the theoretical order $-\log \text{TOL}$, with minor differences due to different collocation order N. A small but consistent impact of spatial dimension d can be observed, with slightly larger efficiency gain for higher dimension.

Numerical computations. Up to here, the results were just predictions, theoretical values obtained from the work and error models derived in Section 3. Of particular interest is whether these model predictions coincide with actual computation.

In Figure 4, contraction rate and final time error of inexact SDC computations are shown. Inexact evaluation of the right-hand side is imitated by adding a random perturbation of size $\epsilon_i^{[j]}$ and uniformly distributed direction. On the left, estimated contraction rates are shown, obtained by regression over the complete SDC iteration. As expected, the exact SDC contraction factor ρ decreases roughly linearly with the time step size. The fixed local tolerance iteration converges with a very similar rate, since the rather small allowed errors can only affect the last sweeps. The optimal rate for variable local tolerances is larger: from (3-24) we expect a rate of $\rho^{1/(d+1)}$, which is indeed achieved. The slightly faster convergence can be attributed to the errors in actual computation not realizing the theoretical worst case.

In Figure 4, right, the final time deviation of the inexact SDC iterations from the limit point, the collocation solution, is shown, relative to the error of the collocation solution itself. The sample mean of twenty realizations is plotted together with the standard deviation, since, in contrast to all other figures, the actual errors depend

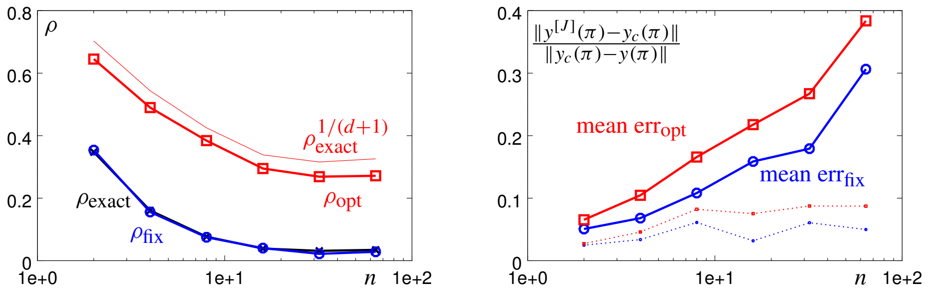


Figure 4. Left: observed contraction factors ρ for exact SDC (ρ_{exact}, crosses), fixed local tolerances (ρ_{fix}, circles), and optimal variable local tolerances (ρ_{opt}, squares) versus number n of time steps. The theoretical contraction rate of $\rho^{1/(d+1)}$ for variable local tolerances is plotted for reference. Right: final time difference between inexact SDC methods and collocation solution, relative to the collocation error. Solid lines are sample means, and dotted lines show the standard deviation.

significantly on the random inexactness of the realizations. We observe that the error model used in defining local tolerances works reasonably well, with comparable final time errors for fixed and optimal variable local tolerances. Again, numerical computations are more accurate than predicted by the worst case estimates. The slow but steady increase with the number n of time steps suggests that the normally distributed local errors do not simply add up, as has been assumed when choosing the tolerance $\text{TOL} \sim n^{-1/2}$.

4B. ***Iterative solver example: heat equation.*** Diffusion processes like heat conduction are usually solved by implicit time-stepping schemes, where solving the arising sparse large-scale linear systems may dominate the computational effort. Here we consider as a prototypical example the linear heat equation

$$\begin{aligned} \dot{y} &= \Delta y, && \text{in } \Omega, \\ y &= 0, && \text{on } \partial\Omega, \\ y &= y_0, && \text{for } t = 0, \end{aligned}$$

on the domain $\Omega =]0, 2\pi[$ and the initial value $y_0 = \chi_{]0,\pi]}$. For spatial discretization, we employ second-order equidistant finite differences on $n = 128$ intervals. We consider a single SDC time step of length $T = 1$ using $N = 4$ Radau-IIa collocation nodes and implicit Euler as the basic method. The exact SDC contraction factor can thus be assumed to be $\rho \approx 0.62$ [27].

The arising linear systems (2-7) assume the form $(I - (t_i - t_{i-1})A)\delta_i^{[j]} = R_i^{[j]}$ with stiffness matrix A. Even though these tridiagonal systems can be solved efficiently with a direct solver, we use iterative solvers in order to evaluate the impact of truncation on inexact SDC performance. As extreme cases we consider simple Jacobi iterations with asymptotic contraction rate $\rho_{\text{Jac}} \approx 1 - 50/n^2$ and multigrid

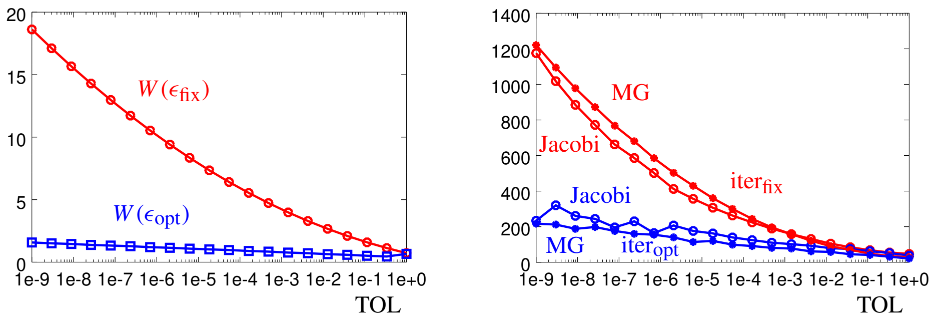


Figure 5. Computational effort of computing an inexact SDC step for the heat equation versus the desired accuracy TOL $\in [10^{-9}, 10^{-1}]\|y^{[0]} - y_c\|$. Left: predicted computational effort $W(\epsilon)$ in arbitrary work units for fixed absolute tolerance and optimized local tolerances. Right: total number of linear solver iterations for fixed and optimized local tolerances, for both Jacobi and multigrid linear solvers. The numbers of Jacobi iterations have been scaled down by a common factor such that they have the same mean as the multigrid iteration numbers, in order to allow a comparison of relative effort between fixed and optimized tolerance choices.

V-cycles with two damped Jacobi presmoothing steps resulting in a contraction rate $\rho_{\text{MG}} \approx 0.25$. The truncation work model (3-13) and consequently also the optimal local tolerances are, however, independent of the iterative solver's contraction rate.

Let us focus on the computational effort incurred by the different local tolerance choices, both predicted and realized. The predicted work $W(\epsilon)$ for fixed and optimized local tolerances is plotted versus the requested SDC iteration accuracy TOL in Figure 5, left, and shows a significant expected benefit of local tolerance optimization. The choice of geometrically decreasing local tolerances $\epsilon_i^{[j]} = \beta\rho^{\gamma j}$ as considered in Section 3D with optimal value $\gamma = 1$ leads to results almost indistinguishable from the optimized tolerances, and is therefore not considered separately.

The actually required work, in terms of number of linear solver iterations, is shown in Figure 5, right, for both Jacobi and multigrid solvers. Note that the iteration numbers of the Jacobi solver have been scaled down by a common factor, such that the relative work between fixed and optimized local tolerances can be observed. While the actual work reduction is less than the predicted one, a factor of five rather than ten for high accuracy, the qualitative behavior is captured very well by the theoretical work model. Moreover, despite the huge difference in convergence speed between Jacobi and multigrid solvers, the relative effort between fixed and optimized local tolerances, and between different required SDC tolerances, is essentially unaffected by the choice of solver, which agrees rather well with the truncation work model derivation in Section 3B.

As before, the accuracy actually achieved is better than the requested tolerance. The ratio $\|y^{[J]} - y_c\|/\text{TOL}$ of actual error and tolerance is (almost) always less

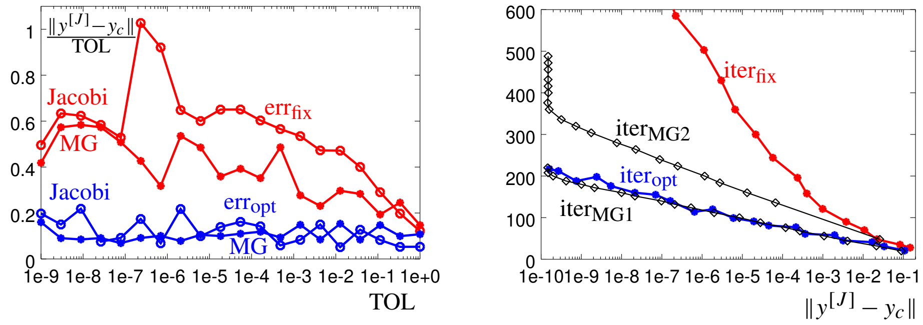


Figure 6. Left: relative deviation of achieved accuracy $\|y^{[J]} - y_c\|$ from the requested tolerance TOL for different choices of local tolerances and linear solvers. Right: total number of linear solver iterations versus the achieved SDC accuracy $\|y^{[J]} - y_c\|$ for different strategies of solving linear systems. Shown are fixed and optimized local tolerances with multigrid solver, as well as simple heuristics of performing exactly one or two multigrid V-cycles.

than one, as predicted by the error bound (3-9). The inefficiency incurred by the error bound not being sharp is, however, moderate, since the achieved accuracy is less than TOL by a factor between two and ten. It depends on the choice of local tolerances, but not much on the linear solver; see Figure 6, left. Consequently, the computational effort required to achieve a certain accuracy, shown in Figure 6, right, resembles very much the work versus requested tolerance shown in Figure 5.

Using a fixed number of linear solver iterations is a simple heuristic for inexact implicit SDC methods [20; 24]. With this choice, linearly convergent solvers can lead to a convergent scheme with expected contraction factor $\max(\rho, \rho_{\text{it}})$, and resembles the geometrically decreasing local tolerances with $\gamma \leq 1$. The efficiency on the heat equation example is comparable to optimized local tolerances for just one V-cycle, and slightly worse for two V-cycles. The best number of iterations will, of course, depend on the problem. A reasonable value can be assumed to be $\lceil \log \rho / \log \rho_{\text{it}} \rceil$.

4C. ***Monte Carlo example: smoothed molecular dynamics.*** Classical molecular dynamics [2] is generally described by Newtonian mechanics of the positions $x \in \mathbb{R}^{nd}$ of n atoms in $\mathbb{R}^d$ with mass M influenced by a potential V:

$$M\ddot{x} = -\nabla V(x). \tag{4-1}$$

One interesting quantity is the time it takes to exit a given potential well or to move between two wells. The computation of these times is expensive as the transitions are rare events, and long trajectories need to be computed before such an event is observed. Statistic reweighting techniques [23] allow one to compute the exit times of interest from exit times induced by a modified potential $\overline{V}$ with shorter

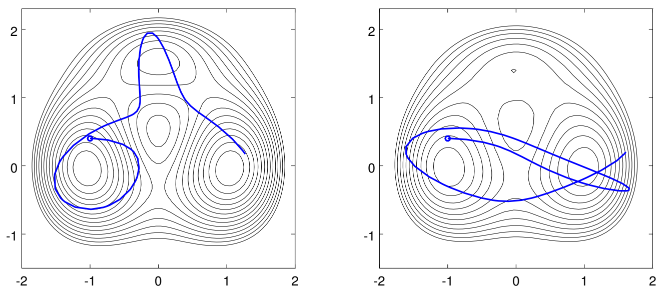


Figure 7. Potential and considered trajectory. Left: original potential V from (4-2). Right: the smoothed potential $\bar{V}$ for $\lambda = 0.316$. The equipotential lines are at the same levels in both pictures.

exit times. One of the modifications in use is potential smoothing by diffusion, i.e., $\bar{V} := V(\lambda)$ with $\partial V/\partial\lambda = \Delta V$. As the number n of involved atoms is usually large, computing $\bar{V}$ by finite element or finite difference methods is out of the question. Instead, pointwise evaluation by convolution with the Green's function is performed [14] using importance sampling:

$$\begin{aligned}\nabla\bar{V}(x) &= (\lambda\sqrt{2\pi})^{-nd}\int_{\mathbb{R}^{nd}} \nabla V(x+s)\exp(-s^2/(2\lambda^2))\,ds\\ &= (\lambda\sqrt{2\pi})^{-nd}\int_{\mathbb{R}^{nd}} (\nabla V(x+s) - Hs)\exp(-s^2/(2\lambda^2))\,ds\\ &\approx \frac{1}{m}\sum_{i=1}^{m}(\nabla V(\xi_i) - H(\xi_i - x)) =: \nabla\widehat{V}_m(x),\end{aligned}$$

where the random variable ξ is normally distributed with mean x and covariance λI, and $H \in \mathbb{R}^{nd}$ is arbitrary. The expected error is proportional to $m^{-1/2}$ and can be estimated in terms of the sample covariance

$$\sigma_m^2 = \frac{1}{m-1}\sum_{i=1}^{m} s_i s_i^T, \qquad s_i = \nabla V(\xi_i) - H(\xi_i - x) - \nabla\widehat{V}(x),$$

as

$$E[\|\nabla\bar{V}(x) - \nabla\widehat{V}_m(x)\|] \approx \frac{\|\sigma_m\|}{\sqrt{m}}.$$

Obviously, s_i and consequently σ_m are particularly small if H is the Hessian of V.

When evaluating $\widehat{V}$ with a requested local tolerance ϵ, the number of sampling points is doubled until $\|\sigma_m\| \le \sqrt{m}\epsilon$. This defines a realization of $\widehat{V}_\epsilon(x)$. Note that this does not give an actual error bound, such that the error analysis and tolerance selection from Section 3 only hold in a probabilistic sense.

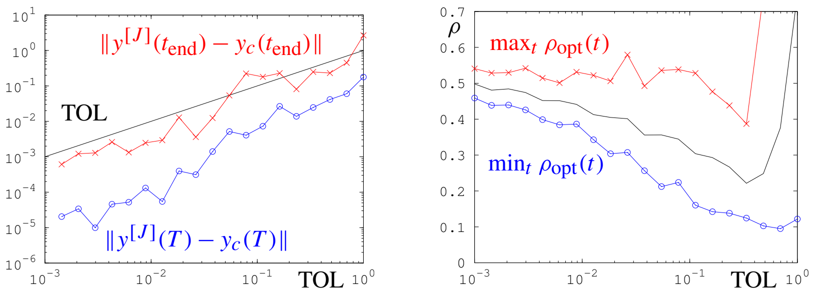


Figure 8. Numerical result averaged over fifteen realizations. Left: estimated error after the first time step of length T (circles) and at final time $t_{\rm end}$ (crosses) versus the requested tolerance TOL. Right: maximal, average, and minimal observed contraction factors $\rho_{\rm opt}$ of the inexact SDC method in all time steps versus the requested step tolerance.

As a simple test problem of this type we consider $n = 1$ and $d = 2$ with $M = I$,

$$V(x) = 3\exp(-\|x-e_2\|^2) - 3\exp(-\|x-5e_2\|^2) - 5\exp(-\|x-e_1\|^2) \\ - 5\exp(-\|x+e_1\|^2) + (x_1^4 + (x_2-1/3)^4)/5, \quad \text{where } (e_i)_j = \delta_{ij}, \quad (4\text{-}2)$$

initial value $x(0) = [-1, 0.4]^T$, $\dot{x}(0) = [2.1, 0]^T$ in the vicinity of one of the three local energy minimizers, final time $t_{\rm end} = 6$, and variance $\lambda = 0.316$. Despite its simplicity, the potential (4-2) as shown in Figure 7 is an interesting test case, as the direct path between the two deep wells crosses a higher potential barrier than the indirect path via the third, shallow well.

Figure 7 shows the original potential V as defined in (4-2) and the considered trajectory on the left, and the smoothed potential $\bar{V}$ for $\lambda = 0.1$ on the right. The shallow well on the top has almost vanished, and the potential barrier between the two dominant wells is much lower. Consequently, the trajectory crosses the barrier easily now and alternates between the two wells.

The ODE (4-1) is transformed into a first-order system to fit into the setting (2-1). For the tests, $N = 4$ collocation points have been used and $n = 15$ equidistant time steps. The numerically observed exact SDC contraction factor varies roughly in a range $[0.15, 0.24]$. For simplicity, a fixed value of 0.2 has been used for computing local tolerances. For the Lipschitz condition (3-3), we notice that f' has values with purely imaginary spectrum, and estimate $L_f(\tau) = \max_{y\in B}\|I + \tau f'(y)\|$ numerically by evaluating $f'(y_0)$ in each step using Monte Carlo integration of V''. In each time step, the initial iteration error $\|y^{[0]} - y_c\|$ is estimated by substituting a single explicit Euler step for y_c, which here yields a reasonable estimation error of usually less than 50% with a minor impact on local tolerances.

The results shown in Figure 8 indicate that the inexact SDC method works essentially as expected, even though the obtained errors $\|y(T) - y_c(T)\|$ are smaller than the target value TOL by one to two orders of magnitude. This is probably due to the error propagation result (3-5) reflecting the worst case rather than the average case. Replacing the generously used triangle inequality by sharper bounds, however, would require not only prescribing the magnitude of the evaluation error but also restricting its direction. If possible and practicable at all, this would require the error analysis to be very much specific for particular problems or right-hand side evaluation schemes.

The interpretation that the observed, better than desired accuracies are due to average versus worst case is supported by the observed inexact SDC contraction rates shown in Figure 8, right. With an exact SDC contraction rate $\rho \approx 0.2$, the targeted inexact contraction rate is $\rho^{1/(d+1)} \approx 0.58$, very close to the worst cases observed in actual computation. There is, however, a significant gap between the best and the worst encountered contraction rates, suggesting that the worst-case behavior is captured well by the theoretical derivations.

Conclusion

The theoretically optimal choice of iteration counts and local tolerances when evaluating basic steps in SDC methods as derived here allows significant savings in computational effort compared to naive strategies. Effort reduction factors between two and six have been observed in examples. Thus, exploiting the inexactness that is possible in SDC methods appears to be attractive for expensive simulations.

The local tolerances are defined in terms of problem-dependent quantities, in particular Lipschitz constants L_f, initial iteration error $\|y^{[0]} - y_c\|$, and contraction factor ρ of exact SDC iterations, which are usually not directly available a priori. For a practical implementation of the optimal choice, adaptive methods based on cheap a posteriori estimates of these quantities are needed. We have considered a particular weak model of error type: independent errors for each evaluation, which are likely to line up to the worst case. Correspondingly, worst-case error bounds have been derived and optimized. In concrete computational problems, often more of the error structure is known, and slightly different approaches would be more appropriate. In sampling problems such as the smoothed molecular dynamics example, the random errors tend to cancel out to some extent. Looking at the average behavior instead of the worst case allows us to use larger local tolerances. On the other hand, the errors are highly correlated in several finite element computations. Consequently, the error differences are small, which leads to different error propagation through the SDC iteration. Extending the approach to these settings is the subject of further research.

Acknowledgements

Partial funding by BMBF grant SOAK is gratefully acknowledged. The authors would like to thank Marcus Weber for providing the molecular dynamics example.

Appendix: Uniqueness of work minimizer

Here we prove that for fixed local tolerance ϵ_{fix}, the continuous relaxation of the work model with respect to the iteration count J is quasiconvex and thus has a unique minimizer.

Theorem. *Let*

$$W(J) = \frac{J+1}{(\mathrm{TOL} - \rho^J \delta)^d}$$

with $\delta > \mathrm{TOL} > 0$, $d > 0$, *and* $0 < \rho < 1$. *Then W has exactly one local minimizer on* $]J_{\min}, \infty[$, *where* $J_{\min} = \log(\mathrm{TOL}/\delta)/\log\rho$.

Proof. The derivative of W is

$$W'(J) = \frac{(\mathrm{TOL} - \rho^J\delta)^d - (J+1)d(\mathrm{TOL} - \rho^J\delta)^{d-1}(-\delta)\rho^J \log\rho}{(\mathrm{TOL} - \rho^J\delta)^{2d}}.$$

We are just interested in the zeros and the sign of the derivative, and multiply with $\delta^{-1}(\mathrm{TOL} - \rho^J\delta)^{d+1} > 0$ for simplification, which gives $\operatorname{sgn} W'(J) = \operatorname{sgn} q(J)$ with

$$q(J) := \frac{\mathrm{TOL}}{\delta} - \rho^J + (J+1)d\rho^J \log\rho.$$

We obtain $q(J_{\min}) = (J_{\min}+1)d\rho^{J_{\min}} \log\rho < 0$ and $q(J) \to \mathrm{TOL}/\delta > 0$ for $J \to \infty$. Since q is continuous, it has an odd number of zeros in $]J_{\min}, \infty[$.

Next we consider

$$\begin{aligned} q'(J) &= \rho^J \log\rho((J+1)d\log\rho - 1) + \rho^J d \log\rho \\ &= \rho^J \log\rho((J+1)d\log\rho + d - 1). \end{aligned}$$

Any zeros of q' have to satisfy $(J+1)d\log\rho + d - 1 = 0$, such that there is at most one zero of q' and correspondingly at most one extremum of q. If q had more than one zero, i.e., at least three zeros, it would have at least two extrema, which is not the case. Thus, q has exactly one zero and consequently W exactly one extremum. The sign of W' changes from negative to positive there, such that W has exactly one local minimizer. $\square$

References

[1] P. Alfeld, *Fixed point iteration with inexact function values*, Math. Comp. **38** (1982), no. 157, 87–98. MR Zbl

[2] M. P. Allen and D. J. Tildesley, *Computer simulation of liquids*, Oxford University, 1987. Zbl

[3] P. Amodio and L. Brugnano, *A note on the efficient implementation of implicit methods for ODEs*, J. Comput. Appl. Math. **87** (1997), no. 1, 1–9. MR Zbl

[4] J. Barnes and P. Hut, *A hierarchical $O(N \log N)$ force-calculation algorithm*, Nature **324** (1986), 446–449.

[5] P. Birken, *Termination criteria for inexact fixed-point schemes*, Numer. Linear Algebra Appl. **22** (2015), no. 4, 702–716. MR Zbl

[6] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Statist. Comput. **9** (1988), no. 4, 669–686. MR Zbl

[7] G. J. Cooper and J. C. Butcher, *An iteration scheme for implicit Runge–Kutta methods*, IMA J. Numer. Anal. **3** (1983), no. 2, 127–140. MR Zbl

[8] G. J. Cooper and R. Vignesvaran, *A scheme for the implementation of implicit Runge–Kutta methods*, Computing **45** (1990), no. 4, 321–332. MR Zbl

[9] P. Deuflhard and F. Bornemann, *Scientific computing with ordinary differential equations*, Texts Appl. Math., no. 42, Springer, 2002. MR Zbl

[10] F. P. E. Dunne and D. R. Hayhurst, *Efficient cycle jumping techniques for the modelling of materials and structures under cyclic mechanical and thermal loading*, Eur. J. Mech. A Solid. **13** (1994), no. 5, 639–660. Zbl

[11] A. Dutt, L. Greengard, and V. Rokhlin, *Spectral deferred correction methods for ordinary differential equations*, BIT **40** (2000), no. 2, 241–266. MR Zbl

[12] B. V. Faleichik, *Analytic iterative processes and numerical algorithms for stiff problems*, Comput. Methods Appl. Math. **8** (2008), no. 2, 116–129. MR Zbl

[13] K. Frischmuth and D. Langemann, *Numerical calculation of wear in mechanical systems*, Math. Comput. Simulation **81** (2011), no. 12, 2688–2701. MR Zbl

[14] E. Gallicchio, S. A. Egorov, and B. J. Berne, *On the application of numerical analytic continuation methods to the study of quantum mechanical vibrational relaxation processes*, J. Chem. Phys. **109** (1998), no. 18, 7745–7755.

[15] R. W. S. Grout, *Mixed-precision spectral deferred correction*, preprint CP-2C00-64959, National Renewable Energy Laboratory, 2015.

[16] S. Güttel and J. W. Pearson, *A rational deferred correction approach to PDE-constrained optimization*, preprint, University of Kent, 2016.

[17] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving ordinary differential equations, I: Nonstiff problems*, 2nd ed., Springer Series Comput. Math., no. 8, Springer, 1993. MR Zbl

[18] J. Huang, J. Jia, and M. Minion, *Accelerating the convergence of spectral deferred correction methods*, J. Comput. Phys. **214** (2006), no. 2, 633–656. MR Zbl

[19] L. O. Jay and T. Braconnier, *A parallelizable preconditioner for the iterative solution of implicit Runge–Kutta-type methods*, J. Comput. Appl. Math. **111** (1999), no. 1–2, 63–76. MR Zbl

[20] M. L. Minion, R. Speck, M. Bolten, M. Emmett, and D. Ruprecht, *Interweaving PFASST and parallel multigrid*, SIAM J. Sci. Comput. **37** (2015), no. 5, S244–S263. MR Zbl

[21] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer, 1999. MR Zbl

[22] J. M. Ortega and W. C. Rheinboldt, *Iterative solution of nonlinear equations in several variables*, Academic, 1970. MR Zbl

[23] C. Schütte, A. Nielsen, and M. Weber, *Markov state models and molecular alchemy*, Mol. Phys. **113** (2015), no. 1, 69–78.

[24] R. Speck, D. Ruprecht, M. Minion, M. Emmett, and R. Krause, *Inexact spectral deferred corrections*, Domain decomposition methods in science and engineering XXII (T. Dickopf, M. J. Gander, L. Halpern, R. Krause, and L. F. Pavarino, eds.), Lect. Notes Comput. Sci. Eng., no. 104, Springer, 2016, pp. 389–396. MR Zbl

[25] W. Tutschke, *Solution of initial value problems in classes of generalized analytic functions*, Springer, 1989. MR Zbl

[26] P. J. van der Houwen and J. J. B. de Swart, *Triangularly implicit iteration methods for ODE-IVP solvers*, SIAM J. Sci. Comput. **18** (1997), no. 1, 41–55. MR Zbl

[27] M. Weiser, *Faster SDC convergence on non-equidistant grids by DIRK sweeps*, BIT **55** (2015), no. 4, 1219–1241. MR Zbl

[28] M. Wilhelms, G. Seemann, M. Weiser, and O. Dössel, *Benchmarking solvers of the monodomain equation in cardiac electrophysiological modeling*, Biomed. Eng. **55** (2010), 99–102.

Received February 14, 2017. Revised October 30, 2017.

MARTIN WEISER: weiser@zib.de
Zuse Institute Berlin, Berlin, Germany

SUNAYANA GHOSH: sunayanag@gmail.com
Zuse Institute Berlin, Berlin, Germany

COMM. APP. MATH. AND COMP. SCI.
Vol. 13, No. 1, 2018
dx.doi.org/10.2140/camcos.2018.13.87

A THIRD ORDER FINITE VOLUME WENO SCHEME FOR MAXWELL'S EQUATIONS ON TETRAHEDRAL MESHES

MARINA KOTOVSHCHIKOVA, DMITRY K. FIRSOV AND SHIU HONG LUI

A third order type II WENO finite volume scheme for tetrahedral unstructured meshes is applied to the numerical solution of Maxwell's equations. Stability and accuracy of the scheme are severely affected by mesh distortions, domain geometries, and material inhomogeneities. The accuracy of the scheme is enhanced by a clever choice of a small parameter in the WENO weights. Also, hybridization with a polynomial scheme is proposed to eliminate unnecessary and costly WENO reconstructions in regions where the solution is smooth. The proposed implementation is applied to several test problems to demonstrate the accuracy and efficiency, as well as usefulness of the scheme to problems with singularities.

1. Introduction

Weighted essentially nonoscillatory (WENO) schemes are high order numerical methods developed to solve hyperbolic partial differential equations (PDEs) with solutions containing discontinuities. In a finite volume (FV) framework these schemes can be implemented on unstructured meshes making them an attractive option for solving problems with singularities due to geometry and/or inhomogeneity in material properties. For linear nondispersive media the system of Maxwell's equations is linear, and for simple geometries it can be solved analytically. In many practical applications the challenge in solving Maxwell's equations is due to complex geometrical features, broadband complex signal types, and/or inhomogeneous material properties. In these cases, finite volume time-domain (FVTD) algorithms are often implemented with success. FV schemes were adapted to Maxwell's equations from computational fluid dynamics (CFD) in the late 1980s by Shankar et al. [26] and include both central [25; 23] and upwind formulations [26; 7; 6].

The upwind FVTD formulation based on the method of characteristics and MUSCL reconstruction has shown good results on a wide range of problems. The

Kotovshchikova was supported by an NSERC postgraduate scholarship, while Lui was supported by an NSERC Discovery grant.
MSC2010: 65M08, 78M12.
Keywords: weighted essentially nonoscillatory (WENO) schemes, finite volume schemes, Maxwell's equations, tetrahedral meshes.

main drawback of the MUSCL scheme is that it is only second order accurate. Moreover, in the presence of singularities, it employs a slope limiter [5; 12] to suppress oscillations and maintain monotonicity. This unavoidably decreases the accuracy to first order at critical points. Higher order finite volume approximation schemes based on polynomial reconstruction are implemented in [22; 13]. These schemes are not a good choice for problems with nonsmooth signals or heterogeneous media nor for simulations on highly distorted meshes, where such schemes can be oscillatory or unstable. Also flux limiters when used with higher order schemes may produce results that are even less accurate than those by MUSCL schemes.

Essentially nonoscillatory (ENO) schemes were developed by Harten et al. [16] to overcome the problem of order degeneracy at critical points. Instead of using limiters to overcome a possible growth of total variation, ENO schemes use an adaptive selection of stencils according to the smoothness of the solution. Better accuracy near discontinuities is achieved by selecting the stencil that doesn't contain a singularity. ENO schemes for multidimensional unstructured meshes can be found in [15; 1; 29], and their implementations to Maxwell's equations in [9; 32].

WENO schemes were developed in [20] to improve the performance of ENO schemes. The key idea of WENO schemes is to use a weighted combination of all ENO stencils for the reconstruction. For unstructured 3D meshes there are two types of WENO schemes. Type I WENO schemes [10; 11; 31; 24] are easier to construct because the linear coefficients can be chosen as arbitrary positive numbers (usually a larger linear weight is given to the central small stencil). The accuracy of the resulting type I WENO scheme is not higher than that on each small stencil. In this work we employ the type II WENO scheme proposed by Zhang and Shu in [33] in which the weighted combination of second order reconstructions is third order accurate. The scheme is more difficult to construct for unstructured meshes as there is no freedom in selecting the linear weights. Linear weights depend solely on the mesh geometry, and in most cases there are negative weights which create stability issues. To overcome this, the criterion proposed in [21] can be used to eliminate reconstructions for which linear weights are large negative numbers.

Different modifications to the computation of nonlinear weights are suggested in the literature to improve the quality of the WENO reconstruction. For unstructured meshes the mapping technique introduced by Henrick et al. in [17] is often suggested [33; 21]. While theoretically mapping applied to third order classical WENO weights does not improve convergence, numerically this technique reduces computational errors. Another approach to improve the accuracy of WENO schemes is to modify the smoothness indicator [8]. In this work the accuracy of WENO schemes is controlled by appropriately choosing the small parameter ϵ in the WENO weights as a function of linear cell sizes. This choice was originally studied for WENO schemes on uniform meshes in 1D by Aràndiga et al. in [2].

This paper is devoted to an efficient application of the type II WENO scheme to a 3D FVTD approximation of Maxwell's equations. Based on the analysis for 1D nonuniform meshes, we implement the optimal choice of the small parameter in WENO weights for maximum achievable spatial accuracy. A threshold for very negative linear weights has been employed to eliminate possible instability. Moreover, to improve CPU time we use a criterion to determine when to apply WENO reconstruction. Basically, the WENO reconstruction is performed only for elements with large smoothness indicators in the WENO weights. The proposed implementation was tested on electromagnetic problems with analytic solutions to confirm that the accuracy and nonoscillatory effect are achieved with the proposed choices of parameters. The robustness of the type II WENO scheme for inhomogeneous media is also demonstrated numerically. Maxwell's equations are challenging due to discontinuities in the solution where the advantages of WENO schemes can be leveraged. The implementation discussed in this paper can be applied to other hyperbolic systems of PDEs.

The paper is organized as follows. Section 2 describes Maxwell's equations in time-domain and their finite volume discretization. Section 3 presents an overview of the type II finite volume WENO scheme together with improvements necessary for its efficient implementation. In Section 4, a 1D analysis of a third order WENO scheme on nonuniform meshes is discussed to support the choices made for accurate 3D applications. Finally, Section 5 presents numerical validations of the proposed implementation of a WENO scheme on tetrahedral meshes.

2. The finite volume scheme for Maxwell's equations

Consider the propagation of electromagnetic waves in a 3D heterogeneous linear isotropic medium with space varying electric permittivity $\epsilon = \epsilon(\mathbf{x})$ and magnetic permeability $\mu = \mu(\mathbf{x})$. Given a bounded region $\Omega \subset \mathbb{R}^3$, the electric field $\mathbf{E}(\mathbf{x}, \mathrm{t})$ and the magnetic field $\mathbf{H}(\mathbf{x}, \mathrm{t})$ are governed by the system of Maxwell's equations

$$\begin{cases} \epsilon \frac{\partial \mathbf{E}}{\partial \mathrm{t}} - \nabla \times \mathbf{H} = \mathbf{J}_E & \text{in } [0, T] \times \Omega, \\ \mu \frac{\partial \mathbf{H}}{\partial \mathrm{t}} + \nabla \times \mathbf{E} = \mathbf{J}_H & \text{in } [0, T] \times \Omega, \\ a\hat{\boldsymbol{n}} \times \mathbf{E} + b\hat{\boldsymbol{n}} \times (\hat{\boldsymbol{n}} \times \mathbf{H}) = 0 & \text{on } [0, T] \times \partial\Omega, \end{cases} \tag{1}$$

where $\mathbf{J}_E$ and $\mathbf{J}_H$ are the sources consisting of imposed currents and term introduced by scattered field formulation, and $\hat{\boldsymbol{n}}$ is the outward unit normal of the boundary $\partial\Omega$. Parameters a and b define different boundary conditions:

- perfect electric conductor (PEC), $a = 1$ and $b = 0$,
- perfect magnetic conductor (PMC), $a = 0$ and $b = 1$, and
- Silver–Müller absorbing boundary condition, $a = 1$ and $b = \sqrt{\mu/\epsilon}$.

Consider the normalized quantities

$$\boldsymbol{x} = l^{-1}\mathbf{x}, \qquad t = c_0 l^{-1}\mathrm{t}, \tag{2}$$

where l is a reference length, $c_0 = (\mu_0\epsilon_0)^{-1/2}$ is a dimensional speed of light in vacuum with $\epsilon_0 \approx 8.854 \cdot 10^{-12}\frac{\mathrm{A\cdot s}}{\mathrm{V\cdot m}}$, and $\mu_0 = 4\pi \cdot 10^{-7}\frac{\mathrm{V\cdot s}}{\mathrm{A\cdot m}}$. The fields $\mathbf{E}$ and $\mathbf{H}$ can be normalized to a typical electric field intensity E by

$$\boldsymbol{E} = \frac{\mathbf{E}}{E}, \qquad \boldsymbol{H} = \frac{Z_0}{E}\mathbf{H}, \qquad \boldsymbol{J}_E = \frac{lZ_0}{E}\mathbf{J}_E, \qquad \boldsymbol{J}_H = \frac{l}{E}\mathbf{J}_H, \tag{3}$$

where $Z_0 = \sqrt{\mu_0/\epsilon_0}$ is the dimensional free-space intrinsic impedance. Then the system (1) can be written in nondimensional form as

$$\begin{cases} \epsilon_r \frac{\partial \boldsymbol{E}}{\partial t} - \nabla \times \boldsymbol{H} = \boldsymbol{J}_E & \text{in } [0, c_0 l^{-1}T] \times \Omega, \\ \mu_r \frac{\partial \boldsymbol{H}}{\partial t} + \nabla \times \boldsymbol{E} = \boldsymbol{J}_H & \text{in } [0, c_0 l^{-1}T] \times \Omega, \\ a_r \hat{\boldsymbol{n}} \times \boldsymbol{E} + b_r \hat{\boldsymbol{n}} \times (\hat{\boldsymbol{n}} \times \boldsymbol{H}) = 0 & \text{on } [0, c_0 l^{-1}T] \times \partial\Omega, \end{cases} \tag{4}$$

where $\epsilon_r = \epsilon/\epsilon_0$, $\mu = \mu/\mu_0$, $a_r = a$, and $b_r = b/Z_0$. For a finite volume discretization, the first two equations of (4) are written in conservative form as

$$\boldsymbol{\alpha}\frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{U}) = \boldsymbol{J},$$

where

$$\boldsymbol{U} = \begin{bmatrix} \boldsymbol{E} \\ \boldsymbol{H} \end{bmatrix}, \qquad \boldsymbol{F}(\boldsymbol{U}) = \begin{bmatrix} \boldsymbol{F}_1(\boldsymbol{U}) & \boldsymbol{F}_2(\boldsymbol{U}) & \boldsymbol{F}_3(\boldsymbol{U}) \end{bmatrix}^T, \qquad \boldsymbol{F}_i = \begin{bmatrix} -e_i \times \boldsymbol{H} \\ e_i \times \boldsymbol{E} \end{bmatrix},$$

and

$$\boldsymbol{\alpha} = \begin{bmatrix} \epsilon_r & 0 \\ 0 & \mu_r \end{bmatrix}, \qquad \boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_E \\ \boldsymbol{J}_H \end{bmatrix}.$$

Consider a partition of the bounded domain $\Omega \subset \mathbb{R}^3$ into a tetrahedral mesh $\bar{\Omega}_T = \bigcup_{i=1}^N \bar{T}_i$. It is assumed that material properties are constant in each cell T_i. Integrating (4) over each tetrahedron T_i and defining the cell averaged values of a given function u as $\bar{u}_i = (1/|T_i|)\int_{T_i} u\, dV$, the semidiscrete finite volume scheme for Maxwell's equations is derived:

$$\boldsymbol{\alpha}_i \frac{\partial \bar{\boldsymbol{U}}_i}{\partial t} + \frac{1}{|T_i|}\int_{\partial T_i} \hat{\boldsymbol{n}} \cdot \boldsymbol{F}\, dS = \boldsymbol{\alpha}_i \frac{\partial \bar{\boldsymbol{U}}_i}{\partial t} + \frac{1}{|T_i|}\sum_{j=1}^{4} |S_{ij}|\hat{\boldsymbol{n}} \cdot \boldsymbol{F}|_{S_{ij}} = \boldsymbol{J}_i, \tag{5}$$

where $\hat{\boldsymbol{n}}$ is the outward unit normal of the tetrahedron boundary ∂T_i consisting of four triangular surfaces S_{ij}, $j = 1, \ldots, 4$. Fluxes in (6) are computed using physical properties on elements T_i and T_j. Physical properties are the same inside homogenous media and different on boundaries between dielectrics. To approximate the flux on each triangular surface S_{ij}, an upwind scheme based on the Steger–Warming flux vector splitting [30] is used. The splitting is based on the method of characteristics and separates the flux on a face into outgoing and incoming parts according to the sign of the eigenvalues of a 6×6 flux matrix [7]. An application of the flux splitting on each face S_{ij} of a tetrahedron T_i gives the upwind finite volume scheme

$$\hat{\boldsymbol{n}} \cdot \boldsymbol{F}|_{S_{ij}} = \begin{bmatrix} -\hat{\boldsymbol{n}}_{ij} \times [\hat{\boldsymbol{n}}_{ij} \times (\boldsymbol{E}_{ij} - \boldsymbol{E}_{ji}) + (Z_i \boldsymbol{H}_{ij} + Z_j \boldsymbol{H}_{ji})]/(Z_i + Z_j) \\ \hat{\boldsymbol{n}}_{ij} \times [-\hat{\boldsymbol{n}}_{ij} \times (\boldsymbol{H}_{ij} - \boldsymbol{H}_{ji}) + (Y_i \boldsymbol{E}_{ij} + Y_j \boldsymbol{E}_{ji})]/(Y_i + Y_j) \end{bmatrix}, \tag{6}$$

where $Z_i = \sqrt{\mu_i/\epsilon_i}$ denotes the intrinsic impedance, $Y_i = Z_i^{-1}$, and $\hat{\boldsymbol{n}}_{ij}$ denotes the outward unit normal of S_{ij}. The surface averaged electromagnetic fields consisting of outgoing $(\boldsymbol{E}_{ij}, \boldsymbol{H}_{ij})$ and incoming $(\boldsymbol{E}_{ji}, \boldsymbol{H}_{ji})$ plane wave contributions are approximated with the desired accuracy from cell averaged values on T_i and its neighbors. Third order approximations of the field components can be obtained with the four-point Gaussian quadrature rule [18; 33]

$$\boldsymbol{U}_{ij} = \sum_{k=1}^{4} g_k \boldsymbol{U}(\boldsymbol{x}_k^{(j)}), \tag{7}$$

where g_k and $\boldsymbol{x}_k^{(j)}$ are the Gaussian quadrature weights and points, respectively. At each Gaussian quadrature point $\boldsymbol{x}_k^{(j)}$, a third order WENO reconstruction is implemented to approximate the components of $\boldsymbol{U}(\boldsymbol{x}_k^{(j)})$ using the fields averages.

3. Third order WENO reconstruction on tetrahedra

In this work we employ the type II third order WENO scheme developed by Zhang and Shu [33]. Its key idea is to construct a nonlinear combination of second order reconstructions on small stencils $\{S_l\}_{l=1}^{s}$ that gives a third order accurate approximation of a smooth solution on the big stencil $S = \bigcup_{l=1}^{s} S_l$ for each quadrature point $\boldsymbol{x}_k^{(j)}$. Therefore, the main advantage over the type I WENO scheme is that much more compact stencils are used to achieve third order accurate nonoscillatory numerical solutions.

The big stencil $S = \{V_m\}_{m=0}^{r}$ is formed by the cell $V_0 = T_i$ and two layers of its neighbors, and consists of $r \leq 17$ elements. A third order approximation of u at each quadrature point $\boldsymbol{x}_k^{(j)}$ is obtained from a quadratic polynomial $p_2(\boldsymbol{x})$ for which

$$\bar{u}_0 = \frac{1}{|V_0|} \int_{V_0} p_2(\boldsymbol{x})\, dV. \tag{8}$$

In local variables

$$\boldsymbol{\xi} = (\xi_1, \xi_2, \xi_3) = \boldsymbol{\xi}(\boldsymbol{x}) = \frac{\boldsymbol{x} - \boldsymbol{x}_0}{h}, \quad h = |V_0|^{1/3}, \tag{9}$$

where $\boldsymbol{x}_0$ is the barycenter of V_0, the quadratic polynomial $p_2(\boldsymbol{x})$ can be written as

$$p_2(\boldsymbol{x}) = \sum_{0 \le i_1+i_2+i_3 \le 2} a_{i_1 i_2 i_3} \xi_1^{i_1} \xi_2^{i_2} \xi_3^{i_3}. \tag{10}$$

Coefficients $a_{i_1 i_2 i_3}$ are computed by matching the cell averages of $p_2(\boldsymbol{x})$ on every element of $S \setminus \{V_0\}$ to the cell averages of u in a least square sense [4]. To avoid computation of integrals $[\xi_1^{i_1} \xi_2^{i_2} \xi_3^{i_3}]_m$ over each element V_m, we use the approach from [22]. At the k-th quadrature point on the j-th face $\boldsymbol{x}_k^{(j)}$, the third order reconstruction polynomial is given by

$$p_2(\boldsymbol{x}_k^{(j)}) = \sum_{m=0}^{r} c_m \bar{u}_m. \tag{11}$$

The coefficients c_m, $m = 1, \dots, r$, depend on the geometry only and are precomputed for each quadrature point $\boldsymbol{x}_k^{(j)}$ at initialization.

Each small stencil consists of four elements from the big stencil, and includes the target element V_0. Typically there are up to $s = 16$ candidates for small stencils $\bigcup_{l=1}^{s} S_l = S$ [33]. For each small stencil $S_l = V_0 \cup \{V_m^{(l)}\}_{m=1}^{3}$, a linear polynomial $p_1^{(l)}(\boldsymbol{x}) = a_0^{(l)} + \sum_{i=1}^{3} a_i^{(l)} \xi_i$ is constructed using a similar procedure as for $p_2(\boldsymbol{x}_k^{(j)})$. Just as with the big stencil, the coefficients $c_m^{(l)}$ in

$$p_1^{(l)}(\boldsymbol{x}_k^{(j)}) = \sum_{m=0}^{3} c_m^{(l)} \bar{u}_m^{(l)} \tag{12}$$

depend on the local geometry only and are precomputed at initialization.

A third order type II WENO reconstruction is built as a nonlinear combination of linear polynomials $p_1^{(l)}$. The nonlinear weights of the WENO scheme are defined using the weights of the third order linear reconstruction. For each quadrature point $\boldsymbol{x}_k^{(j)}$ of the face S_j, the linear weights $\{\gamma_l\}_{l=1}^{s}$ are such that the linear combination of polynomials $p_1^{(l)} \boldsymbol{x}_k^{(j)}$ is closest to $p_2 \boldsymbol{x}_k^{(j)}$. The weights for each Gaussian point are found from the system of linear equations constructed from two parts. The first part is formed by taking $u = 1, \xi_1^2, \xi_2^2, \xi_3^2, \xi_1\xi_2, \xi_1\xi_3, \xi_2\xi_3$ in

$$u(\boldsymbol{x}_k^{(j)}) = \sum_{l=1}^{s} \gamma_l p_1^{(l)}(\boldsymbol{x}_k^{(j)}). \tag{13}$$

The second part is constructed using the requirement that

$$p_2(\boldsymbol{x}_k^{(j)}) \overset{1}{=} \sum_{l=1}^{s} \gamma_l p_1^{(l)}(\boldsymbol{x}_k^{(j)}) \tag{14}$$

holds for an arbitrary u, where $\overset{1}{=}$ represents the equality in the least square sense. Solution of two systems together gives the optimal linear weights for the type II third order linear reconstruction of u at the quadrature point $\boldsymbol{x}_k^{(j)}$:

$$u^{\mathrm{Lin}}(\boldsymbol{x}_k^{(j)}) = \sum_{l=1}^{s} \gamma_l p_1^{(l)}(\boldsymbol{x}_k^{(j)}). \tag{15}$$

While the reconstruction based on linear weights works well for smooth solutions and relatively good unstructured meshes, our goal is to adapt it to the case where the solution is not smooth and the mesh quality is arbitrary. We still take a linear combination of the reconstructions using small stencils, but now so-called nonlinear weights $\{\omega_l\}_{l=1}^{s}$ are employed. Those are designed so that $\omega_l \approx \gamma_l$ in cells where the solution is smooth (so third order accuracy is maintained) and $\omega_l \approx 0$ otherwise to suppress oscillations. The classic WENO weights are defined as

$$\omega_l = \frac{\tilde{\omega}_l}{\sum_{m=1}^{s} \tilde{\omega}_m}, \quad \tilde{\omega}_l = \frac{\gamma_l}{(\epsilon + \mathrm{SI}_l)^2}, \tag{16}$$

where ϵ is a small number traditionally chosen to be between 10^{-2} and 10^{-40} to avoid division by zero, and SI_l is the smoothness indicator on the l-th small stencil:

$$\mathrm{SI}_l = \sum_{i=1}^{3} \int_{T_0} |T_0|^{-1/3} \left(\frac{\partial p_1^{(l)}(\boldsymbol{x})}{\partial x_i} \right)^2 dV. \tag{17}$$

As was pointed out in [2; 17], the choice of ϵ has a crucial effect on the accuracy of classic 1D WENO reconstructions. It was shown that $\mathrm{SI}_l \sim h^2$ for smooth solutions and $\mathrm{SI}_l \sim h^4$ near critical points, suggesting $\epsilon \sim h^2$ as an optimal choice to preserve the accuracy near critical points. Assuming that this dependence is even more important for reconstructions on 3D unstructured meshes with high ratios between linear cell sizes, we implemented the choices

$$\epsilon_i = h_i^k, \quad k = 1, 2, 4, \tag{18}$$

in numerical experiments. These choices are based on the accuracy analysis for the 1D WENO3 scheme which will be presented in the next section. As suggested in [33] we also employ the mapped weights technique [17].

Now to form the type II WENO reconstruction at the point $\boldsymbol{x}_k^{(j)}$, we replace the linear weights γ_l in (15) by the nonlinear ω_l defined in (16)

$$u^{\mathrm{WENO}}(\boldsymbol{x}_k^{(j)}) = \sum_{l=1}^{s} \omega_l p_1^{(l)}(\boldsymbol{x}_k^{(j)}). \tag{19}$$

Since the type II WENO scheme uses smaller stencils than type I WENO, linear weights are completely dependent on the geometry. In 3D problems with complex geometry, the mesh quality is hard to control. The least square solution for linear weights always gives some negative weights. For mildly negative weights, the splitting technique from [27] is implemented. But on unstructured meshes there is always a small percentage of large negative linear weights which compromise the stability of the computation. To overcome this Liu and Zhang in [21] proposed to replace approximations for which

$$\max_l(|\gamma_l|) > \zeta, \quad 1 \le \zeta \le 10, \tag{20}$$

by a more expensive type I WENO reconstruction. To have the same compact stencil in all reconstructions, we replace a type II WENO reconstruction at quadrature points where (20) holds with a third order polynomial reconstruction (11). In numerical experiments we did not encounter any problem with such substitution even for discontinuous solutions. This can be explained by the fact that very negative linear weights appear only for some quadrature points of a given face. As a result the surface integral (7) is a combination of both WENO and polynomial reconstructions. Therefore, the WENO scheme partially compensates for the oscillatory effect of the polynomial scheme. While using ζ values of up to 10 gives good results on 2D triangular meshes [21], we used the upper limit of $\zeta = 1$ on 3D tetrahedral meshes for stability.

The CPU time for computations using a WENO scheme is significantly larger than that for third order polynomial schemes. Therefore, from a practical point of view, WENO schemes should only be used when their nonoscillatory properties benefit the solution. One way to reduce computational cost is to switch between polynomial and WENO reconstructions depending on the values of smoothness indicators. A naive criterion, such as

$$\max_l \mathrm{SI}_l > \frac{\epsilon}{2}, \tag{21}$$

for WENO reconstruction can significantly reduce the computational time without compromising either the smooth or discontinuous numerical solutions. This is referred to as accelerated WENO (WENOA) in the numerical experiments.

4. Accuracy of third order WENO scheme on nonuniform grid in 1D

Since the numerical solution of 3D Maxwell's equations using a WENO scheme with a fixed small value of ϵ in (16) has unpredictable accuracy, we turned to a 1D theory for selecting a proper value of ϵ. It was shown by Aràndiga et al. in [2] that the accuracy of WENO schemes in 1D can be controlled by defining ϵ based on the mesh size h. The focus of this section is on an analysis of a third order WENO scheme for 1D nonuniform meshes. We use it as a guideline to choose ϵ in 3D simulations.

Consider a nonuniform 1D mesh $a = x_{1/2} < \cdots < x_{i-1/2} < x_{i+1/2} < x_{i+3/2} < \cdots < x_{N+1/2} = b$ with sizes $h_i = x_{i+1/2} - x_{i-1/2}$ on an interval $I_i = [x_{i-1/2}, x_{i+1/2}]$.

The linear polynomials $p_{1,i}^{(l)}(x)$ defined on small stencils $S_i^{(l)} = \{I_{i+l-2}, I_{i+l-1}\}$, $l = 1, 2$, can be obtained as

$$p_{1,i}^{(l)}(x) = \bar{u}_i + \frac{2h_i}{h_{i+l-2} + h_{i+l-1}}[\bar{u}_{i+l-1} - \bar{u}_{i+l-2}]\xi, \quad l = 1, 2. \tag{22}$$

The classic WENO3 weights in 1D are defined by [20; 19]

$$\omega_{l,i} = \frac{\tilde{\omega}_{l,i}}{\tilde{\omega}_{1,i} + \tilde{\omega}_{2,i}} \quad \text{with } \tilde{\omega}_{l,i} = \frac{\gamma_l}{(\epsilon + \mathrm{SI}_{l,i})^p}, \ l = 1, 2, \tag{23}$$

where the linear weights are given by $\gamma_1 = \frac{1}{3}$ and $\gamma_2 = \frac{2}{3}$, and the smoothness indicators $\mathrm{SI}_{l,i}$, $l = 1, 2$, can obtained as

$$\mathrm{SI}_{l,i} = h_i \int_{I_i} (p_{1,i}^{(l)}(x))_x^2 \, dx = \frac{4h_i^2}{(h_{i+l-2}+h_{i+l-1})^2}[\bar{u}_{i+l-1} - \bar{u}_{i+l-2}]^2, \quad l = 1, 2. \tag{24}$$

If $u(x)$ is a smooth function on the big stencil $S_i = \bigcup_{l=1}^{2} S_i^{(l)}$, then the finite volume WENO3 reconstruction with weights given by (23) has the accuracy property [19]

$$u_{i+1/2}^{\text{WENO}} = u(x_{i+1/2}) + O(h^{2+k}), \tag{25}$$

provided that

$$\omega_{l,i} = \gamma_l + O(h^k), \quad k \in \{0, 1\}, \ l = 1, 2. \tag{26}$$

Theorem 1. *Let $u(x) \in C^3$ on the big stencil S_i. Then the smoothness indicators* (24) *have the following properties.*

(1) *If $u'(x) \neq 0$ for all $x \in S_i$, then*

$$\mathrm{SI}_{l,i} = \alpha_i(x_i)h_i^2 + O(h_i^3), \quad l \in \{1, 2\}, \tag{27}$$

$$\mathrm{SI}_{2,i} - \mathrm{SI}_{1,i} = \beta_i(x_i)h_i^3 + O(h_i^4) \tag{28}$$

for some locally Lipschitz continuous $\alpha_i(x)$ and $\beta_i(x)$.

(2) *If $u(x)$ has a point $x^* \in S_i \setminus \{x_i\}$ such that $u'(x^*) = 0$, then*

$$\mathrm{SI}_{l,i} = \alpha_{l,i}(x_i)h_i^4 + O(h_i^5), \quad l \in \{1, 2\}, \tag{29}$$

$$\mathrm{SI}_{2,i} - \mathrm{SI}_{1,i} = \beta_i(x_i)h_i^4 + O(h_i^5) \tag{30}$$

for some locally Lipschitz continuous $\alpha_{l,i}(x)$ and $\beta_i(x)$.

Proof. Using the Taylor series of the primitive function $U(x) = \int_{-\infty}^{x} u(\xi)\,d\xi$ about x_i, we get

$$\begin{aligned}\mathrm{SI}_{1,i} &= \frac{4h_i^2}{(h_{i-1}+h_i)^2}[\bar{u}_i - \bar{u}_{i-1}]^2\\ &= \frac{4h_i^2}{(h_{i-1}+h_i)^2}\left(\frac{U(x_i+\frac{1}{2}h_i)-U(x_i-\frac{1}{2}h_i)}{h_i} - \frac{U(x_i-\frac{1}{2}h_i)-U(x_i-\frac{1}{2}h_i-h_{i-1})}{h_{i-1}}\right)^2\\ &= \big(u'(x_i)h_i - \tfrac{1}{3}u''(x_i)(\tfrac{1}{2}h_i+h_{i-1})h_i + O(h_i^3)\big)^2.\end{aligned}$$

Similarly one can get

$$\mathrm{SI}_{2,i} = \big(u'(x_i)h_i + \tfrac{1}{3}u''(x_i)(\tfrac{1}{2}h_i+h_{i+1})h_i + O(h_i^3)\big)^2.$$

Let $\kappa_l = h_i/h_{i-2l-3}$; then

$$\mathrm{SI}_{l,i} = \big(u'(x_i)h_i + (\tfrac{2}{3}l-1)(\tfrac{1}{2}+\kappa_l)u''(x_i)h_i^2 + O(h_i^3)\big)^2. \tag{31}$$

Therefore, we deduce (27) and (28) with

$$\alpha_i(x_i) = [u'(x_i)]^2, \qquad \beta_i(x_i) = \frac{1+\kappa_1+\kappa_2}{3}u'(x_i)u''(x_i).$$

Now consider the case when $u'(x^*) = 0$ for some $x^* \in S_i \setminus \{x_i\}$. Let $x_i - x^* = \kappa h_i$ with $0 < |\kappa| < \frac{3}{2}$. Then using the Taylor series of $u'(x)$ about x_i at x^*, we get

$$u'(x_i) = u''(x_i)\kappa h_i + O(h_i^2),$$

which is then substituted into (31) to derive

$$\mathrm{SI}_{l,i} = (\delta_l u''(x_i)h_i^2 + O(h_i^3))^2,$$

where $\delta_l = \kappa + (\frac{2}{3}l-1)(\frac{1}{2}+\kappa_l)$. Therefore, we obtain the estimates (29) and (30) with $\alpha_{l,i}(x_i) = [\delta_l u''(x_i)]^2$ and $\beta_i(x_i) = (\delta_2^2-\delta_1^2)[u''(x_i)]^2$. □

Theorem 2. *Let $u(x) \in C^3$ on the big stencil S_i, and $\epsilon = Mh^m$ in* (23), *for some $M > 0$ and $m \geq 0$. Then the following hold.*

(1) *If $u'(x) \neq 0$ for all $x \in S_i$, then*

$$u_{i+1/2}^{\mathrm{WENO}} - u(x_{i+1/2}) = O(h^3). \tag{32}$$

(2) *If there is a point $x^* \in S_i \setminus \{x_i\}$ such that $u'(x^*) = 0$, then*

$$u_{i+1/2}^{\mathrm{WENO}} - u(x_{i+1/2}) = \begin{cases} O(h^3), & m \leq 3,\\ O(h^2), & m \geq 4.\end{cases} \tag{33}$$

Proof. As in [2] we start by writing

$$\frac{1}{(\epsilon+\mathrm{SI}_{1,i})^p} = \frac{1}{(\epsilon+\mathrm{SI}_{2,i})^p}\left(1+\frac{\mathrm{SI}_{2,i}-\mathrm{SI}_{1,i}}{\epsilon+\mathrm{SI}_{1,i}}\right)^p. \tag{34}$$

(1) Consider the case when $u'(x) \neq 0$ for all $x \in S_i$. Then using (27) and (28),

$$\frac{\mathrm{SI}_{2,i} - \mathrm{SI}_{1,i}}{\epsilon + \mathrm{SI}_{1,i}} = \frac{p\beta_i(x_i)}{\eta_1 M + \eta_2 \alpha_i(x_i)} h^r + O(h^{r+1}), \tag{35}$$

where

$$r = \max(1, 3-m) \geq 1, \qquad \eta_1 = \begin{cases} 1, & m \leq 2, \\ 0, & m > 2, \end{cases} \qquad \eta_2 = \begin{cases} 0, & m < 2, \\ 1, & m \geq 2. \end{cases}$$

Using (35) in (34) and substituting into (23), we get

$$\tilde{\omega}_{1,i} + \tilde{\omega}_{2,i} = \frac{1}{(\epsilon + \mathrm{SI}_{2,i})^p}(1 + \gamma_1 \nu_{1,i}(x_i) h^r + O(h^{r+1})),$$

where $\nu_{1,i} = p\beta_i(x_i)/(\eta_1 M + \eta_2 \alpha_i(x_i))$ is a locally Lipschitz continuous function. Then

$$\omega_{2,i} = \frac{\gamma_2}{1 + \gamma_1 \nu_{1,i}(x_i) h^r + O(h^{r+1})} = \gamma_2 + O(h^r).$$

Following the same steps, one can derive the same estimate for $\omega_{1,i}$. From (26) and (25) we deduce that (32) holds on S_i regardless of the value of ϵ.

(2) Now assume that at some point $x^* \in S_i \setminus \{x_i\}$, we have $u'(x^*) = 0$. Then

$$\frac{\mathrm{SI}_{2,i} - \mathrm{SI}_{1,i}}{\epsilon + \mathrm{SI}_{1,i}} = \frac{\beta_i(x_i) h^r}{\eta_1 M + \eta_2 \alpha_{1,i}(x_i)} + O(h^{r+1}), \tag{36}$$

where

$$r = \max(0, 4-m) \geq 0, \qquad \eta_1 = \begin{cases} 1, & m \leq 4, \\ 0, & m > 4, \end{cases} \qquad \eta_2 = \begin{cases} 0, & m < 4, \\ 1, & m \geq 4. \end{cases}$$

Following the same steps as before for $m \leq 3$, we get the third order estimate in (33). For $m \geq 4$ using (36) in (34) we get that

$$\frac{1}{(\epsilon + \mathrm{SI}_{1,i})^p} = \frac{1}{(\epsilon + \mathrm{SI}_{2,i})^p}[1 + \nu_{1,i}(x_i) + O(h)],$$

where $\nu_{1,i}(x_i) = (1 + \beta_i(x_i)/(\eta_1 M + \alpha_{1,i}(x_i)))^p - 1$ is a locally Lipschitz continuous function. Therefore,

$$\omega_{2,i} = \frac{\gamma_2}{(1 + \gamma_1 \nu_{1,i}(x_i) + O(h))} = \frac{\gamma_2}{(1 + \gamma_1 \nu_{1,i}(x_i))} + O(h) + \gamma_2 - \gamma_2 = \gamma_2 + O(1).$$

The same result can be obtained for $\omega_{1,i}$. As follows from (25)–(26) for $m \geq 4$, WENO3 gives only second order reconstruction near the critical point x^*. □

Theorem 3. *Let $u(x)$ be a piecewise smooth function with a jump discontinuity $[u^*] = [u(x^*)]$ in $S_i \setminus S_i^{(l)}$, $l \in \{1, 2\}$, at the point x^*. If $\epsilon = Mh^m$, where $m \geq 1$, in* (23), *then the WENO*3 *reconstruction with weights defined by* (16) *gives*

$$u^{\mathrm{WENO}}_{i+1/2} = u(x_{i+1/2}) + O(h^2). \tag{37}$$

Proof. Assume that $x^* \in S_i \setminus S_i^{(1)}$; then we have

$$u_{i+1/2}^{(1)} = u(x_{i+1/2}) + O(h^2), \qquad u_{i+1/2}^{(2)} = u(x_{i+1/2}) + O([u^*]). \tag{38}$$

Since $\mathrm{SI}_{1,i} = O(h^r)$, $r \in \{2, 4\}$, and $\mathrm{SI}_{2,i} = O([u^*]^2)$ for $\epsilon = O(h^m)$, we get

$$\omega_{1,i} = O(h^{2\min(m,r)}[u^*]^{-4}), \qquad \omega_{2,i} = O(1). \tag{39}$$

Therefore,

$$u_{i+1/2}^{\mathrm{WENO}} = \omega_{1,i} u_{i+1/2}^{(1)} + \omega_{2,i} u_{i+1/2}^{(2)} = u(x_{i+1/2}) + O(h^{2\min(m,r)}[u^*]^{-3}) + O(h^2), \tag{40}$$

which, for $m \geq 1$, gives (37). □

Numerical experiments support the validity of the above theory in 1D. Since it would be much more difficult to analyze the general 3D case with unstructured meshes, we use the analysis above to choose ϵ in (16) for our 3D experiments.

5. Numerical examples

In this section a set of 3D electromagnetic (EM) test problems that include plane wave propagation in a parallel plate waveguide, the scattering of a plane wave from a perfectly conducting (PEC) sphere, and plane wave reflection/transmission through a dielectric prism are discussed. Numerical experiments were carried out on an Intel i7-4790k 4.4 GHz quad core CPU with 32 GB of RAM. C++ OpenMP is used to utilize multicore architecture. For the temporal discretization we employ the third order strong stability-preserving (SSP) Runge–Kutta scheme [28].

Example 1: parallel plate waveguide. Consider the problem of a plane wave propagation in a parallel plate wave guide. In this example the computational domain is represented by a cube with linear size $l = 2$ m. We impose PEC boundary conditions on cube faces parallel to the x-y plane, and PMC boundary condition on two faces parallel to the z-x plane. A plane wave excited at $x = -1$ and propagating in x direction is given by the boundary conditions

$$E_z^{\mathrm{in}} = f(t), \quad H_y^{\mathrm{in}} = -f(t)\epsilon_0^{1/2}\mu_0^{-1/2}, \quad E_x^{\mathrm{in}} = E_y^{\mathrm{in}} = H_x^{\mathrm{in}} = H_z^{\mathrm{in}} = 0. \tag{41}$$

The geometry of the problem is shown in Figure 1.

First consider an incoming plane wave (41) given by the Gaussian pulse

$$f(t) = e^{-b^{-2}(t-t_0)^2}, \tag{42}$$

where $b = 1.2 \times 10^{-9}$ s and $t_0 = 0.5lc_0^{-1}$. Experiments are performed on meshes with relatively uniform linear size of tetrahedrons equal to 0.2, 0.1, and 0.05. To validate WENO schemes with $\epsilon = h, h^2, h^4$ in (16) (WENO-h, h^2, h^4) the discrete

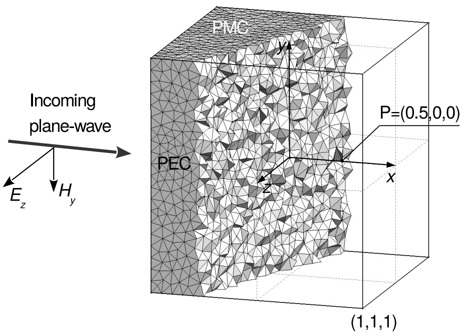


Figure 1. Propagation in a parallel plate waveguide: geometry of the problem.

# of cells	L^2 error	order	L^2 error	order	L^2 error	order
	MUSCL		Polynomial			
8040	$2.747358 \cdot 10^{-2}$		$1.580182 \cdot 10^{-2}$			
64076	$1.263663 \cdot 10^{-2}$	1.12	$2.045965 \cdot 10^{-3}$	2.95		
554668	$6.040267 \cdot 10^{-3}$	1.06	$2.235041 \cdot 10^{-4}$	3.19		
4028196	$3.140725 \cdot 10^{-3}$	0.94				
	WENO-h		WENO-h^2		WENO-h^4	
8040	$1.338234 \cdot 10^{-2}$		$1.409145 \cdot 10^{-2}$		$1.959543 \cdot 10^{-2}$	
64076	$1.942162 \cdot 10^{-3}$	2.78	$2.106160 \cdot 10^{-3}$	2.74	$5.156885 \cdot 10^{-3}$	1.93
554668	$2.133779 \cdot 10^{-4}$	3.19	$2.554968 \cdot 10^{-4}$	3.04	$1.268922 \cdot 10^{-3}$	2.02

Table 1. Propagation in a parallel plate waveguide: L^2 errors at $T = lc_0^{-1} (l = 2\,\mathrm{m})$ for MUSCL, third order polynomial, and WENO-h, h^2, h^4 schemes.

L^2 errors at time $T = lc_0^{-1}$ are computed by

$$l_2(\boldsymbol{U}(T)) = \frac{\left[\sum_{i=1}^{N} |T_i| \sum_{j=1}^{3} \frac{1}{2}(\epsilon_r \epsilon_0 (\bar{E}_i^j)^2 + \mu_r \mu_0 (\bar{H}_i^j)^2)\right]^{1/2}}{\left[\epsilon_0 \sum_{i=1}^{N} |T_i|\right]^{1/2}}. \tag{43}$$

In Table 1 discrete L^2 errors for WENO-h, h^2, h^4 schemes are compared to the ones by MUSCL [7] and third order polynomial schemes. Comparison of time-domain solutions at the observation point $P = (0.5, 0, 0)$ (see Figure 1) is shown in Figure 2. The best resolution of peaks is obtained with WENO-h, while WENO-h^4 significantly distorts the solution near critical points. These results suggest that the 1D theory on the choice of ϵ is applicable to 3D simulations.

Table 2 shows storage requirements as well as the CPU times for MUSCL, WENO, accelerated WENO (WENOA), and polynomial schemes. The speedup achieved by WENOA scheme compared to WENO is due to the application of

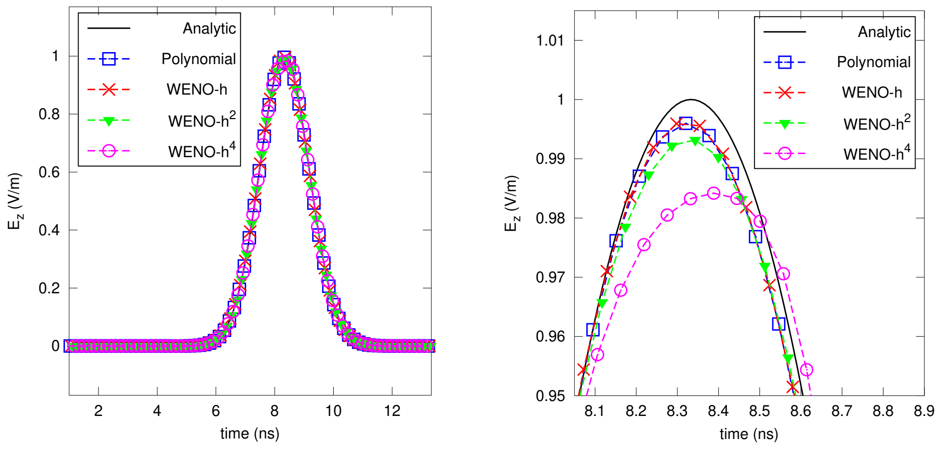


Figure 2. Propagation in a parallel plate waveguide: time-domain solution for the propagation of Gaussian pulse at the observation point (0.5, 0, 0).

# of cells	Storage (GB) MUSCL	Polyn.	WENO	CPU$_T$ (s), $T = lc_0^{-1}$ MUSCL	Polyn.	WENO-h^2	WENOA-h^2
64076	0.04	0.2	1.2	118	197	6601	851
554668	0.3	1.5	11.8	2193	3490	136194	16128
4028196	2.2			31580			

Table 2. Propagation in a parallel plate waveguide: storage and CPU time for MUSCL, third order polynomial, and WENO schemes.

the criterion $\max_l \mathrm{SI}_l > \epsilon/2$ for WENO reconstructions. In this case less than 5% of all flux computations use the expensive WENO approximation. At the same time computation of SI_l itself is computationally expensive, which degrades the performance of WENOA compared to the polynomial scheme. Therefore, a more efficient criterion could further improve the performance of WENOA schemes.

Next consider a discontinuous signal given by

$$f(t) = H(t - t_s)H(t_e - t),$$

where $H(t)$ is the Heaviside step function and $t_s = \frac{1}{8}lc_0^{-1}$ and $t_e = \frac{7}{8}lc_0^{-1}$. Figure 3 shows time-domain solutions of the E_z field at the observation point $P = (0.5, 0, 0)$ using polynomial and WENO-h, h^2, h^4 schemes. The results illustrate that the 1D analysis of the WENO3 scheme for discontinuous solutions regarding the choice of ϵ is also valid for 3D numerical simulations.

Example 2: scattering from a PEC sphere. Consider the classical scattering problem of a plane wave at a PEC sphere for which the analytic series solution is known [14; 3]. The computational domain is represented by a sphere of radius 3 m with

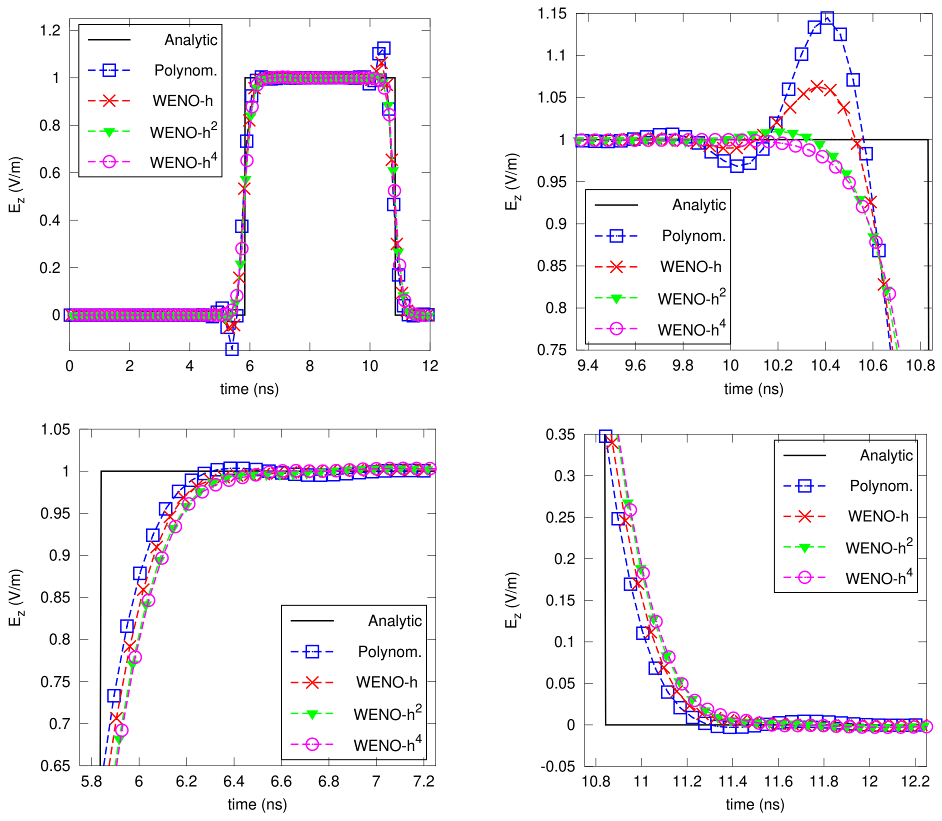


Figure 3. Propagation in a parallel plate waveguide: time-domain solution in time for the propagation of a discontinuous pulse at the observation point (0.5, 0, 0).

a sphere (PEC) of radius 0.5 m cut out at the origin (see Figure 4). The mesh consists of smaller tetrahedrons with average edge length 0.0625 in the region near a PEC surface and larger tetrahedrons with linear size 0.125 at the outer free space boundary. The generated mesh contains 539332 tetrahedra with 2026 of them containing a PEC face. The x component of the electric field of the incident plane wave E_x^I is given by the derivative of the Gaussian pulse

$$E_x^{\text{inc}} = -2\frac{t-t_0}{b^2} A e^{-(t-t_0)^2/b^2}, \tag{44}$$

where $A = 1.7489 \times 10^{-9}\ \frac{\text{V}\cdot\text{s}}{\text{m}}$, $b = 1.5 \times 10^{-9}$ s, and $t_0 = 6 \times 10^{-9}$ s.

The solution of the scattered FVTD formulation using WENO-h_i, h_i^2, h_i^4 schemes as well as the polynomial scheme are compared to the analytic solution at the observation point shown in Figure 4. The results for the E_x field presented in Figure 5 demonstrate that WENO-h_i^4 generates much larger errors than those of WENO-h_i or WENO-h_i^2. This again agrees with the theory for WENO3 in the 1D case.

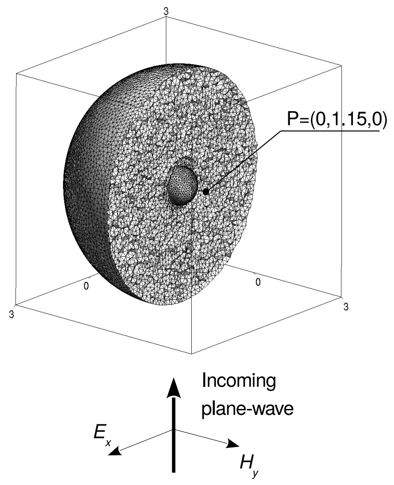


Figure 4. Scattering from PEC sphere: problem geometry and mesh.

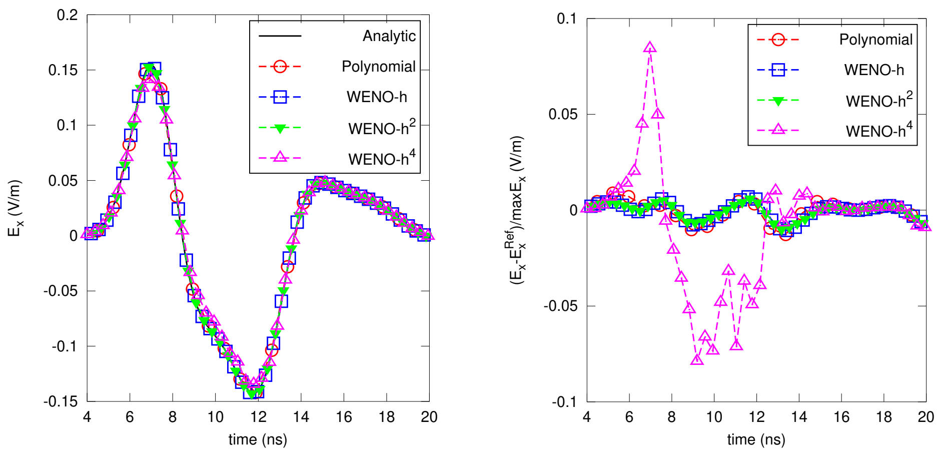


Figure 5. Scattering from PEC sphere: time-domain solution at observation points using third order linear and WENO schemes.

Example 3: glass prism in a waveguide. The last example demonstrates how a WENO scheme handles a problem with inhomogeneous media. This is also the test case where a third order polynomial scheme may not be stable for a reasonable time step. Like in the first example, consider a free space cube domain enclosed between two parallel PEC and two PMC plates. A plane wave signal propagating in the x direction is given by the Gaussian pulse (41)–(42). Inside the cube a glass rhombus prism with dielectric properties $\epsilon_r = 2$ and $\mu_r = 1$ is placed. The dimensions of the prism are shown in Figure 6. Numerical simulation using a third

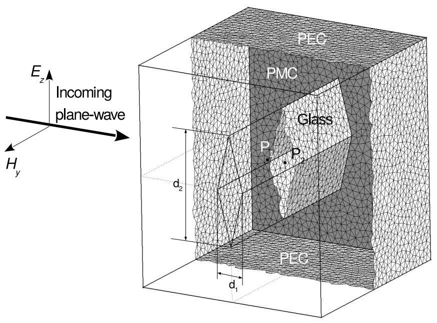


Figure 6. Glass prim in a waveguide: problem geometry and mesh. Observation points shown have coordinates $P_1 = (-0.2, 0, 0)$ and $P_2 = (0, 0, 0)$, and rhombus diagonals are $d_1 = 0.28$ and $d_2 = 1.0$.

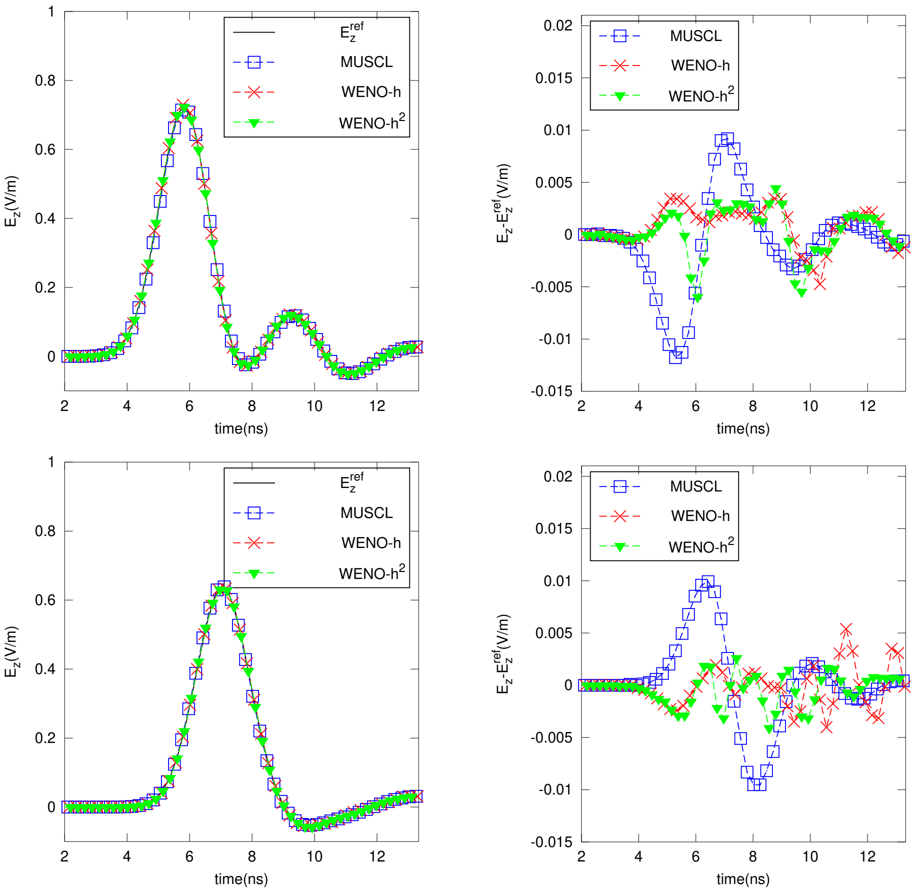


Figure 7. Glass prism in a waveguide: time-domain solution at observation points $P_1 = (-0.2, 0, 0)$ (top row) and $P_2 = (0, 0, 0)$ (bottom row) using MUSCL and WENO-h, h^2 schemes on a mesh with $h = 0.05$ compared to the reference solution by the MUSCL scheme on a mesh with $h = 0.0125$.

order polynomial scheme is unstable for this configuration. Therefore, we only compare numerical results obtained by WENO-h, h^2 schemes on a mesh of average linear cell size 0.05 to the result by the MUSCL scheme [7] on the same mesh. Numerical solution by the MUSCL scheme on a finer mesh with linear cell size 0.0125 is used as a reference solution. Figure 7 shows the time-domain solution for the E_z field as well as a pointwise error with reference solution at two observation points (before and inside the glass prism) shown in Figure 6. We find that while the polynomial scheme diverges for this problem, WENO schemes still converge with better accuracy than the MUSCL scheme. We notice higher level oscillations in the results by the WENO-h scheme compared to WENO-h^2. This suggests WENO-h^2 as a better choice for problems with dielectric contrasts.

6. Summary

In this paper we have successfully implemented a third order type II WENO scheme developed in [33] to solve the linear Maxwell's equations on tetrahedral meshes. An efficient implementation of the scheme is challenging due to its strong dependence on mesh geometry, mesh scale, and high computational cost. Because of the unstructured mesh, the least square solution of the system for finding linear weights almost always contains negative components which create unstable and inaccurate results. To solve this problem we used a hybridization with a third order polynomial scheme at quadrature points with very negative linear weights. Also due to irregular geometries a small number of small stencil matrices are singular. These stencils are removed at the initialization step to avoid polluting the linear scheme. We also implemented specific choices of ϵ dependent on cell sizes in the definition of nonlinear weights which allowed us to control both the accuracy and dissipation in the numerical solution. In our study we used a 1D accuracy analysis as a guideline for the 3D scheme and our numerical experiments confirmed its validity. As in earlier work for uniform meshes in 1D [2], we found that $e = h_i^2$ for each cell is optimal for solutions containing both smooth and singular parts. To reduce computational cost associated with WENO reconstruction, we implemented a criterion that determines which stencils could use cheaper polynomial reconstruction instead. The resulting WENOA-h^2 scheme is more efficient than lower order FV schemes such as MUSCL, and is nonoscillatory and more stable than linear schemes for EM problems with varying material properties and complex geometries.

References

[1] R. Abgrall, *On essentially non-oscillatory schemes on unstructured meshes: analysis and implementation*, J. Comput. Phys. **114** (1994), no. 1, 45–58. MR Zbl

[2] F. Aràndiga, A. Baeza, A. M. Belda, and P. Mulet, *Analysis of WENO schemes for full and global accuracy*, SIAM J. Numer. Anal. **49** (2011), no. 2, 893–915. MR Zbl

[3] C. A. Balanis, *Advanced engineering electromagnetics*, Wiley, 1989.

[4] T. J. Barth and P. O. Frederickson, *Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction*, 28th Aerospace Sciences Meeting, no. 90-0013, AIAA, 1990.

[5] P. Batten, C. Lambert, and D. M. Causon, *Positively conservative high-resolution convection schemes for unstructured elements*, Internat. J. Numer. Methods Engrg. **39** (1996), no. 11, 1821–1838. MR Zbl

[6] D. Baumann, C. Fumeaux, and R. Vahldieck, *Field-based scattering-matrix extraction scheme for the FVTD method exploiting a flux-splitting algorithm*, IEEE T. Microw. Theory **53** (2005), no. 11, 3595–3605.

[7] P. Bonnet, X. Ferrieres, F. Issac, F. Paladian, J. Grando, J. C. Alliot, and J. Fontaine, *Numerical modeling of scattering problems using a time domain finite volume method*, J. Electromagnet. Wave **11** (1997), no. 8, 1165–1189. Zbl

[8] R. Borges, M. Carmona, B. Costa, and W. S. Don, *An improved weighted essentially non-oscillatory scheme for hyperbolic conservation laws*, J. Comput. Phys. **227** (2008), no. 6, 3191–3211. MR Zbl

[9] A. Chatterjee and R.-S. Myong, *Efficient implementation of higher-order finite volume time-domain method for electrically large scatterers*, Prog. Electromagn. Res. B **17** (2009), 233–254.

[10] M. Dumbser and M. Käser, *Arbitrary high order non-oscillatory finite volume schemes on unstructured meshes for linear hyperbolic systems*, J. Comput. Phys. **221** (2007), no. 2, 693–723. MR Zbl

[11] M. Dumbser, M. Käser, V. A. Titarev, and E. F. Toro, *Quadrature-free non-oscillatory finite volume schemes on unstructured meshes for nonlinear hyperbolic systems*, J. Comput. Phys. **226** (2007), no. 1, 204–243. MR Zbl

[12] L. J. Durlofsky, B. Engquist, and S. Osher, *Triangle based adaptive stencils for the solution of hyperbolic conservation laws*, J. Comput. Phys. **98** (1992), no. 1, 64–73. Zbl

[13] D. Firsov, J. LoVetri, I. Jeffrey, V. Okhmatovski, C. Gilmore, and W. Chamma, *High-order FVTD on unstructured grids using an object-oriented computational engine*, ACES J. **22** (2007), no. 1, 71–82.

[14] R. F. Harrington, *Time-harmonic electromagnetic fields*, McGraw-Hill, 1961.

[15] A. Harten and S. R. Chakravarthy, *Multi-dimensional ENO schemes for general geometries*, Tech. Report 187637, NASA, 1991.

[16] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarthy, *Uniformly high-order accurate essentially nonoscillatory schemes, III*, J. Comput. Phys. **71** (1987), no. 2, 231–303. MR Zbl

[17] A. K. Henrick, T. D. Aslam, and J. M. Powers, *Mapped weighted essentially non-oscillatory schemes: achieving optimal order near critical points*, J. Comput. Phys. **207** (2005), no. 2, 542–567. Zbl

[18] P. Hillion, *Numerical integration on a triangle*, Internat. J. Numer. Methods Engrg. **11** (1977), no. 5, 797–815. MR Zbl

[19] G.-S. Jiang and C.-W. Shu, *Efficient implementation of weighted ENO schemes*, J. Comput. Phys. **126** (1996), no. 1, 202–228. MR Zbl

[20] X.-D. Liu, S. Osher, and T. Chan, *Weighted essentially non-oscillatory schemes*, J. Comput. Phys. **115** (1994), no. 1, 200–212. MR Zbl

[21] Y. Liu and Y.-T. Zhang, *A robust reconstruction for unstructured WENO schemes*, J. Sci. Comput. **54** (2013), no. 2–3, 603–621. MR Zbl

[22] C. Ollivier-Gooch and M. Van Altena, *A high-order-accurate unstructured mesh finite-volume scheme for the advection–diffusion equation*, J. Comput. Phys. **181** (2002), no. 2, 729–752. Zbl

[23] S. Piperno, M. Remaki, and L. Fezoui, *A nondiffusive finite volume scheme for the three-dimensional Maxwell's equations on unstructured meshes*, SIAM J. Numer. Anal. **39** (2002), no. 6, 2089–2108. MR Zbl

[24] T. Pringuey and R. S. Cant, *High order schemes on three-dimensional general polyhedral meshes — application to the level set method*, Commun. Comput. Phys. **12** (2012), no. 1, 1–41. MR Zbl

[25] M. Remaki, *A new finite volume scheme for solving Maxwell's system*, COMPEL **19** (2000), no. 3, 913–932. Zbl

[26] V. Shankar, W. F. Hall, and A. H. Mohammadian, *A time-domain differential solver for electromagnetic scattering problems*, Proc. IEEE **77** (1989), no. 5, 709–721.

[27] J. Shi, C. Hu, and C.-W. Shu, *A technique of treating negative weights in WENO schemes*, J. Comput. Phys. **175** (2002), no. 1, 108–127. Zbl

[28] C.-W. Shu and S. Osher, *Efficient implementation of essentially nonoscillatory shock-capturing schemes*, J. Comput. Phys. **77** (1988), no. 2, 439–471. MR Zbl

[29] T. Sonar, *On the construction of essentially non-oscillatory finite volume approximations to hyperbolic conservation laws on general triangulations: polynomial recovery, accuracy and stencil selection*, Comput. Methods Appl. Mech. Engrg. **140** (1997), no. 1–2, 157–181. MR Zbl

[30] J. L. Steger and R. F. Warming, *Flux vector splitting of the inviscid gasdynamic equations with application to finite-difference methods*, J. Comput. Phys. **40** (1981), no. 2, 263–293. MR Zbl

[31] P. Tsoutsanis, V. A. Titarev, and D. Drikakis, *WENO schemes on arbitrary mixed-element unstructured meshes in three space dimensions*, J. Comput. Phys. **230** (2011), no. 4, 1585–1601. MR Zbl

[32] M. Wirianto, W. A. Mulder, and E. C. Slob, *Applying essentially non-oscillatory interpolation to controlled-source electromagnetic modelling*, Geophys. Prospect. **59** (2011), no. 1, 161–175.

[33] Y.-T. Zhang and C.-W. Shu, *Third order WENO scheme on three dimensional tetrahedral meshes*, Commun. Comput. Phys. **5** (2009), no. 2–4, 836–848. MR Zbl

Received March 9, 2017. Revised January 3, 2018.

MARINA KOTOVSHCHIKOVA: m.a.kotovshchikova@gmail.com
Department of Mathematics, University of Manitoba, Winnipeg, Manitoba, Canada

DMITRY K. FIRSOV: d.k.firsov@gmail.com
San Jose, CA, United States

SHIU HONG LUI: luish@cc.umanitoba.ca
Department of Mathematics, University of Manitoba, Winnipeg, Manitoba, Canada

COMM. APP. MATH. AND COMP. SCI.
Vol. 13, No. 1, 2018
dx.doi.org/10.2140/camcos.2018.13.107

ON A SCALABLE NONPARAMETRIC DENOISING OF TIME SERIES SIGNALS

LUKÁŠ POSPÍŠIL, PATRICK GAGLIARDINI,
WILLIAM SAWYER AND ILLIA HORENKO

Denoising and filtering of time series signals is a problem emerging in many areas of computational science. Here we demonstrate how the nonparametric computational methodology of the finite element method of time series analysis with H_1 regularization can be extended for denoising of very long and noisy time series signals. The main computational bottleneck is the inner quadratic programming problem. Analyzing the solvability and utilizing the problem structure, we suggest an adapted version of the spectral projected gradient method (SPG-QP) to resolve the problem. This approach increases the granularity of parallelization, making the proposed methodology highly suitable for graphics processing unit (GPU) computing. We demonstrate the scalability of our open-source implementation based on PETSc for the Piz Daint supercomputer of the Swiss Supercomputing Centre (CSCS) by solving large-scale data denoising problems and comparing their computational scaling and performance to the performance of the standard denoising methods.

1. Introduction

Time series signals (i.e., data measured in intervals over a period of time) are typical for many practical areas such as econometrics (e.g., movement of stock prices [17]), climatology (e.g., temperature changes [39]), or molecular dynamics (e.g., in conformational changes of the molecule [19]). The analysis of time series signals aims to extract meaningful characteristics and understand the process which has generated those data. Such an analysis is the key ingredient in forecasting the process beyond the observed and measured time. However, one of the main difficulties in the analysis of real measurements is the presence of measurement/experimental noise. Additionally, an almost exponentially growing amount of collected data in many practical applications requires a development of better and faster data-driven denoising, modeling, and classification tools suitable for high performance computing (HPC).

MSC2010: 37M10, 62-07, 62H30, 65Y05, 90C20.
Keywords: time series analysis, quadratic programming, SPG-QP, regularization.

Suppose we observed time series signal $x_t \in \mathbb{R}^n$, $t = 1, \dots, T$, (where n is the data dimension and T is the length of time series) and those data are appropriately described by the model function $\mu(t, \Theta)$ with parameters $\Theta \in \mathbb{R}^m$ and an additive noise ε, i.e.,

$$x_t = \mu(t, \Theta) + \varepsilon_t, \quad t = 1, \dots, T, \tag{1}$$

where $\{\varepsilon_t\}$ is a family of independent and identically distributed (i.i.d.) random variables with zero expectation. The explicit model function $\mu(t, \Theta)$ is chosen a priori based on our knowledge of the particular application. In general, the aim of the modeling process is to determine optimal parameters $\bar{\Theta}$ such that the observed data x_t are described by (1) in the most optimal way, for example using maximum likelihood estimation (MLE) or minimizing mean-square error. Finally, the denoised signal can be obtain as an output of (1) with known $\bar{\Theta}$ and without the presence of the (eliminated) noise term ε_t.

In the first part of the introduction, we shortly review the general classification of time series modeling methodologies based on the choice of μ. In the second part, we investigate methods from the point of computational cost and highlight the importance of developing the optimization algorithms for effective solution. The final part of the introduction presents the finite element method of time series analysis with H_1 regularization (FEM-H1) methodology used in the approach presented in this paper.

1.1. ***General model classification.*** In the simple case, the form of the model function μ is known and, for instance, it can be expressed as some a priori defined function dependent on time. If the dimension of the underlying parameters Θ is finite, then this method is called *parametric*.

For example, in the case of a linear regression

$$\mu(t, \theta_0, \theta_1) = \theta_1 t + \theta_0 \tag{2}$$

the unknown model parameters $\theta_0, \theta_1 \in \mathbb{R}$ can be found using MLE (or minimizing least-square error) as a solution of an appropriate optimization problem

$$[\bar{\theta}_0, \bar{\theta}_1] = \arg \min_{\theta_0, \theta_1} \sum_{t=1}^{T} \|x_t - \mu(t, \theta_0, \theta_1)\|^2. \tag{3}$$

The recovered signal is obtained as values of $\mu(t, \bar{\theta}_0, \bar{\theta}_1)$, $t = 1, \dots, T$. However, the model used (2) is valid only if the original data was generated by a linear model — and if no nonlinear effects had a significant impact on the underlying process.

Another example of parametric methods are hidden Markov models (HMMs). Here, it is a priori assumed that there exist regimes such that data in each regime is distributed according to some explicit parametric distribution from a known and fixed family of parametric distributions (e.g., Gaussian, Poisson, etc.). The aim is then to search for an optimal regime-switching homogeneous Markov process represented by unknown components of transition matrix and initial states [1].

In general, parametric methods are usually based on rather strong explicit assumptions about the problem structure. These assumptions help create a tractable finite-dimensional formulation of the problem, which can be solved analytically or numerically and efficiently. In general, the more model assumptions are imposed, the less general the model is — and the simpler the numerical optimization problem to be solved is. On the other hand, imposing an unspecific parametric structure leads to models that incorrectly describe the problem under consideration.

The way to avoid the (possibly inappropriate) restrictive a priori explicit parametric assumptions about the dynamics of the model parameters is to use *nonparametric* models. In the case of nonparametric models, the dimension of the underlying parameters Θ is infinite and we assume that optimal parameters are represented as functions from an a priori restricted class of feasible functions. However, the larger generality and complexity of the nonparametric models used also imposes a much higher computational cost on the resulting infinite-dimensional optimization problem. Therefore, the nonparametric models are much more challenging in numerical implementation and execution.

An example of a nonparametric method is a generalized additive model (GAM) [25]. In comparison to linear regression (2)–(3), the model function μ in GAM can be defined as an arbitrary, nonlinear, and nonparametric smooth function from the Sobolev space on an interval $[t_1, t_T] = [1, T]$

$$W^2([t_1, t_T]) = \left\{ \mu(\cdot) \in \mathscr{C}^{(2)}([t_1, t_T]) : \int_{t_1}^{t_T} [\mu''(\hat{t})]^2 \, \mathrm{d}\hat{t} < \infty \right\}. \tag{4}$$

The optimal (i.e., sufficiently smooth) modeling function is then given by solving the optimization problem

$$\bar{\mu} = \arg\min \sum_{t=1}^{T} (x_t - \mu(t)) + \lambda \int_{t_1}^{t_T} [\mu''(\hat{t})]^2 \, \mathrm{d}\hat{t}. \tag{5}$$

Here $\lambda > 0$ represents the regularization parameter which has to be estimated [49].

1.2. ***Computational cost.*** Choosing the "most optimal" tools in every particular application is not a trivial task and is made more difficult by the interplay of many factors [36], most of all by the following two factors: (i) the amount of bias that is introduced by the analysis method (for example, coming from the eventually wrong a priori assumptions about the linearity, Gaussianity, and homogeneity of the underlying processes) and (ii) the computational scalability of different data-driven algorithms — as well as the possibility to deploy these algorithms in an HPC setting — to be able to process the data en masse.

Experience shows that the analysis algorithms that introduce the highest-potential bias (e.g., standard linear Fourier filtering methods based on the fast Fourier transform (FFT) or parametric Bayesian methods like HMM with Gaussian or Poisson outputs and a time-homogenous Markov model assumption [36; 37; 47]) demonstrate the best HPC scaling performance whereas the nonlinear and non-Gaussian approaches like convolutional neural networks (CNNs) need more communication and scale worse — so only the deployment of massively parallel GPU architectures helped to reach the scale-up that was necessary to apply these methods to large realistic problems [11; 46].

From the mathematical perspective, essentially all of the data analysis and classification methods currently available in the standard analysis packages can be formulated as the numerical algorithms for solutions of large optimization problems.

To give some examples, the standard Fourier, wavelet, and kernel filtering algorithms for denoising the signals $x_t \in \mathbb{R}^n$ — as well as the parameter identification methods for support vector machine (SVM) classification, linear discriminant analysis, and linear autoregressive models (AR) — can be formulated and implemented as solution algorithms for the same type of unconstrained quadratic minimization problem (QP)

$$\bar{y} = \arg\min_y \|x - \Phi y\|_2,$$

where $\|\cdot\|_2$ denotes the Euclidean norm, $y \in \mathbb{R}^r$ (r is typically much less then n), and $\Phi \in \mathbb{R}^{n\times r}$ is a known filtering matrix. Variational methods (like regularized kernel filtering, compressed sensing, and regularized model inference) [8; 50; 48] can be obtained by adding the regularizing inequality constraints to this convex problem: for example, a very popular compressed sensing algorithm in a dual formulation can be obtained by adding the linear inequality constraint $\|y\|_1 \leq \epsilon$ (where $\|\cdot\|_1$ denotes the L_1 norm and ϵ is some a priori fixed “sparsity” parameter) to the above unconstrained QP. Neural networks can be straightforwardly approached as parametric nonconvex optimization problems of the type

$$\bar{y} = \arg\min_y \|x - \Phi(y)\|_2,$$

where Φ is some a priori fixed nonlinear operator characterizing the network topology and y are unknown network parameters.

Finally, the nonstationary and nonparametric denoising and modeling methods based on regularized nonconvex clustering algorithms (like the FEM-H1 methodology developed in [20]) can be implemented as the solution of a minimization problem of a regularized clustering functional with equality and inequality constraints. In following we briefly review this approach.

1.3. ***Nonparametric nonstationary FEM-H1 methodology.*** We will consider observed data as time series $x_t \in \mathbb{R}^n$, $t = 1, \dots, T$. Our aim is to find coefficients $\Theta(t)$ of some model function $\mu(t, \Theta)$ such that this function in some sense fits the data in the best way. This fitting condition (i.e., the distance between observed data and values of the model function) is measured by a metric g, which can be for example defined by means of the function

$$g(x_t, \Theta(t)) = (x_t, \mathbb{E}[\mu(t, \Theta(t))])^2. \tag{6}$$

Therefore, the most appropriate parameters $\Theta(t)$ can be obtained by solving the variational problem

$$\bar{\Theta} = \arg \min_{\Theta(\,\cdot\,) \in \Omega_\Theta} L(\Theta), \quad L(\Theta) = \sum_{t=1}^{T} g(x_t, \Theta(t)), \tag{7}$$

where L refers to a model distance function and Ω_Θ represents the space of all feasible parameters of parameter functions $\Theta(\,\cdot\,)$ for the considered model. However, this problem is ill posed if only one sequence of data $\{x_1, \dots, x_T\}$ is available (this is a typical situation for many practical applications, e.g., computational finance or climatology, where only one historical sequence of data is available for each particular time series). One option of regularizing this problem and making it well posed is based on the clustering of $\Theta(t)$; i.e., one can assume that there exist K different stationary parameters $\Theta = [\Theta_1, \dots, \Theta_K]$ such that the fitness function (6) can be expressed as a convex combination

$$g(x_t, \Theta(t)) = \sum_{i=1}^{K} \gamma_i(t) g(x_t, \Theta_i),$$

where $\gamma_k(t) \in \{1, \dots, T\} \to [0, 1]$, $k = 1, \dots, K$, are so-called model indicator functions [38; 29]. These functions define the *activeness* of an appropriate i-th cluster at a given time t; if $\gamma_i(t) = 1$, then the data are modeled by the i-th model in time t. These properties can be written in the form of constraints

$$\sum_{i=1}^{K} \gamma_i(t) = 1 \quad \text{for all } t, \quad 0 \le \gamma_i(t) \le 1 \quad \text{for all } t, i. \tag{8}$$

Hence, model indicator functions could be considered switching functions between individual models on clusters. Additionally, one can incorporate additional information about observed processes, for example by assuming that switching between clusters is in some sense slower than the changes of the signal caused by the presence

```
Set feasible initial approximation $\Gamma^0 \in \Omega_\Gamma$
    while $\| L_\varepsilon(\Theta^k, \Gamma^k) - L_\varepsilon(\Theta^{k-1}, \Gamma^{k-1}) \| \geq \varepsilon$
        Solve $\Theta^k = \arg\min_{\Theta \in \Omega_\Theta} L_\varepsilon(\Theta, \Gamma^{k-1})$ (with fixed $\Gamma^{k-1}$)
        Solve $\Gamma^k = \arg\min_{\Gamma \in \Omega_\Gamma} L_\varepsilon(\Theta^k, \Gamma)$ (with fixed $\Theta^k$)
        $k = k + 1$
    end while
Return approximation of model parameters $\Theta^k$ and approximation of model indicator functions $\Gamma^k$
```

Algorithm 1. Outer optimization algorithm.

of the modeling error or the noise in data. In our notation, this approach means that model indicator functions γ_i are smooth in some appropriately chosen function space. For example, one can enforce the smoothness in H_1 space by introducing the Tikhonov-based penalization term

$$\begin{gathered}[\bar{\Theta}, \bar{\Gamma}] = \arg \min_{\substack{\Theta \in \Omega_\Theta \\ \Gamma \in \Omega_\Gamma}} L_\varepsilon(\Theta, \Gamma), \\ L_\varepsilon(\Theta, \Gamma) = \sum_{t=1}^{T} \sum_{i=1}^{K} \gamma_i(t) g(x_t, \Theta_i) + \varepsilon^2 \sum_{i=1}^{K} \sum_{t=2}^{T} (\gamma_i(t-1) - \gamma_i(t))^2,\end{gathered} \tag{9}$$

where Ω_Γ is a feasible set defined by conditions (8) and ε^2 denotes the regularization parameter.

This methodology was called FEM-H1, and it was introduced and developed in [27; 28; 31; 29; 30; 32; 33]. For the unified and simplified derivation of the method, as well as its relation to classical methods of unsupervised learning, please see [38]. Moreover, the method was extended for spatial regularization using the network information in the graph-based form of the regularization matrix [20]. In this case, the only difference appears in the formulation of the smoothing term.

From a numerical point of view, the problem (9) can be solved as a sequence of split optimization problems; see Algorithm 1.

Please notice that the first optimization problem in Algorithm 1 is strongly connected to the type of modeling problem and model used. However, if we are able to solve the stationary variant of the problem, then this clustered problem includes only one modification represented by the multiplication by constant coefficients $\gamma_i(t)$. Beyond that, this problem can be reduced into K completely independent problems; for each cluster we are solving the stationary problem. And also the size of this problem is typically small since we suppose that the number of clusters is reasonably small.

One of the main challenges in applying this framework to the analysis of real time series data (e.g., in computational finance, climatology, or neuroscience) is the high computational cost of the second optimization subproblem in this algorithm. Due to this limitation, published applications of these methods are confined to relatively small data sets [27; 28; 31; 29; 30; 32; 33]. This second optimization subproblem is completely independent of the type application, i.e., independent of the choice of fitness function (6). Nevertheless, the size of this problem is given by KT and cannot be separated because of the conditions (8) and the form of regularization term in (9). In optimization theory, the problem of this form (quadratic cost function with a feasible set formed by linear equality and inequality constraints) is called a quadratic programming problem (QP) [42; 14]. Therefore, if we develop the efficient solver to deal with this main computational bottleneck of the FEM-H1 data analysis framework, then we will be able to apply the framework to very large realistic data sets from different application areas (finance, image processing, bioinformatics, etc.). A central goal of this paper is to provide an algorithmic solution to this fundamental problem of the FEM-H1 framework.

Therefore, we will subsequently concentrate on the HPC solution of the problem of unknown model indicator functions Γ. For practical reasons, we define a column vector with all (unknown) model indicator functions by

$$\gamma := [\gamma_1, \dots, \gamma_K] \in \mathbb{R}^{KT}$$

and problem (9) for constant Θ can be written in the form of block-structured QP problem

$$\begin{aligned}
\bar{\gamma} &:= \arg\min_{\gamma \in \Omega_\Gamma} L_\varepsilon(\gamma), \\
\gamma &:= [\gamma_1, \dots, \gamma_K] \in \mathbb{R}^{KT}, \\
\gamma_i &:= [\gamma_i(1), \dots, \gamma_i(T)] \in \mathbb{R}^{T}, \\
L_\epsilon(\gamma) &:= \frac{1}{T} b_\Theta^\intercal \gamma + \frac{\epsilon^2}{T} \gamma^\intercal H \gamma, \\
\Omega_\Gamma &:= \left\{ \gamma \in \mathbb{R}^{KT} : \gamma \geq 0 \wedge \sum_{k=1}^{K} \gamma_k(t) = 1 \text{ for all } t = 1, \dots, T \right\},
\end{aligned} \tag{10}$$

where $H \in \mathbb{R}^{KT \times KT}$ is a block-diagonal matrix, whose blocks $H_i \in \mathbb{R}^{T \times T}$ are formed by Laplace matrices, and

$$b_\Theta := [g(x_t, \Theta_1), \dots, g(x_t, \Theta_K)] \in \mathbb{R}^{KT}$$

denotes the column block-structured vector of modeling errors [29]. Notice that we scaled the cost function by positive coefficient $1/T$ to control the scale of the function values for the cases with large T.

The paper is organized as follows. In Section 2, we examine the solvability and properties of the QP problem (10). Afterwards in Section 3, we present the modification of the spectral projected gradient method for QP problems suitable for an HPC implementation and discuss its advantages in comparison to other standard algorithms. In our project, we are interested in the HPC implementation of the FEM-H1 methodology to be able to deal with very long time series. From the beginning, we consider a situation when even the input data cannot be stored and operated on on one computational node; therefore, the distributed layout of the vectors and matrices has to be introduced and considered during the whole solution process. In Section 4 we briefly introduce our parallel implementation approach. Section 5 presents the performance of our algorithm on a data denoising problem, which is constructed to mimic the main features (like the very high noise-to-signal ratios and non-Gaussianity of the noise) that are typical for time series from practical applications. In contrast to the analysis of the "real life" practical data (where the underlying "true signal" is hidden in the noise and is not known a priori), analysis of this test data that we propose allows for a direct comparison of the introduced method to different standard denoising algorithms. It also allows the assessment of the denoising performance of the methods for various ratios of signal-to-noise — an assessment that cannot be achieved for the "real life" data. We also present the scalability results of our implementation on the Piz Daint supercomputer. In this section, we show the efficiency of FEM-H1 methodology in comparison with other standard denoising approaches. We show that our method outperforms other standard denoising methods in terms of the denoising quality in the situations when the signal-to-noise ratio of the data becomes small.

Finally, Section 6 concludes the paper and presents some ideas for our future research.

2. Solvability of inner QP

For the simplicity of our analysis, we rewrite the problem (10) using the convenient notation

$$\min_{x \in \Omega} f(x), \qquad f(x) := \tfrac{1}{2} x^\mathsf{T} A x - b^\mathsf{T} x, \tag{11}$$

where $A := (\epsilon^2/T) H \in \mathbb{R}^{KT \times KT}$ is a symmetric positive semidefinite (SPS) Hessian matrix of a quadratic cost function $f : \mathbb{R}^{KT} \to \mathbb{R}$ and $b := -(1/T) b_\Theta^\mathsf{T}$ is the so-called right-hand side vector. This name came from the necessary optimality condition for the unconstrained problem $\Omega = \mathbb{R}^{KT}$, which is given by [42; 7; 14]

$$\nabla f(x) = Ax - b = 0 \iff Ax = b. \tag{12}$$

Since Hessian matrix A of the quadratic function f is SPS, this cost function is continuous and (not strictly) convex. Moreover, the null space (kernel) is given by

$$\operatorname{Ker} A = \operatorname{span}\{[\mathbf{1}^\mathsf{T}, \mathbf{0}, \dots, \mathbf{0}]^\mathsf{T}, [\mathbf{0}, \mathbf{1}^\mathsf{T}, \mathbf{0}, \dots, \mathbf{0}]^\mathsf{T}, \dots, [\mathbf{0}, \dots, \mathbf{0}, \mathbf{1}^\mathsf{T}]^\mathsf{T}\} \subset \mathbb{R}^{KT}, \quad (13)$$

where we denote $\mathbf{1} := [1, \dots, 1]^\mathsf{T} \in \mathbb{R}^T$ and $\mathbf{0} := [0, \dots, 0]^\mathsf{T} \in \mathbb{R}^T$.

The feasible set $\Omega \subset \mathbb{R}^{KT}$ is a (nonempty) bounded closed convex set, and it can be equivalently defined by

$$\Omega = \{\gamma \in \mathbb{R}^{KT} : \gamma \geq 0 \wedge B\gamma = c\},$$

where $B := [I, \dots I] \in \mathbb{R}^{T \times KT}$, $c := \mathbf{1}$, and $I \in \mathbb{R}^{T \times T}$ denotes the identity matrix. Using this notation, we can easily conclude that the optimization problem (11) is a QP problem with the SPS Hessian matrix, linear equality constraints, and bound constraints

$$\min \tfrac{1}{2} x^\mathsf{T} A x - b^\mathsf{T} x \quad \text{subject to } Bx = c,\ x \geq 0. \quad (14)$$

The existence of a solution of (14) is implied by the Weierstrass extreme value theorem: the real-valued cost function is continuous, and the nonempty feasible set is bounded.

However, the uniqueness of this solution is not so straightforward. It is given by the relationship between the null space (kernel) of Hessian matrix A, linear term b, and the feasible set Ω, since the differences between solutions lies in this vector space. For instance, if A is symmetric positive definite (SPD), then the cost function is strictly convex and the solution is unique on any nonempty closed convex feasible set [14]. Unfortunately, in our case the Hessian matrix is only SPS and the solution is not unique for an arbitrary feasible set. For example in the unconstrained case, if $\Omega := \mathbb{R}^n$, then the problem could possibly be nonsolvable; if $b \notin \operatorname{Im} A$, then the linear system (12) has no solution. If $b \in \operatorname{Im} A$, then the system of all solutions of the unconstrained problem is given by $\bar{x} = A^+ b + d$, where A^+ denotes the Moore–Penrose pseudoinverse of the singular matrix A and the vector d represents an arbitrary vector from (in this case nontrivial) $\operatorname{Ker} A$. Therefore, all solutions of the problem differ by the vector from $\operatorname{Ker} A$.

At first, we present the generalization of previous observations from the unconstrained case to solutions of a problem (14).

Lemma 1. *Let $\bar{x}_1, \bar{x}_2$ be two solutions of problem* (14). *Then*

$$\bar{x}_1 - \bar{x}_2 \in \operatorname{Ker} A \cap \operatorname{Ker} B.$$

Proof. Let us denote $d := \bar{x}_2 - \bar{x}_1$. Then using the definition of a quadratic cost function f and simple manipulations, we obtain

$$f(\bar{x}_2) = f(\bar{x}_1 + d) = f(\bar{x}_1) + d^\mathsf{T} \nabla f(\bar{x}_1) + \tfrac{1}{2} d^\mathsf{T} A d. \quad (15)$$

We suppose that both $\bar{x}_1$ and $\bar{x}_2$ are minimizers (both $f(\bar{x}_1)$ and $f(\bar{x}_2)$ are minimal values of f on the feasible set); therefore, $f(\bar{x}_1) = f(\bar{x}_2)$. Using this and comparing sides of equality (15), we can write

$$\tfrac{1}{2} d^\mathsf{T} A d = -d^\mathsf{T} \nabla f(\bar{x}_1). \tag{16}$$

The left side of this equation is always nonnegative because A is SPS. Moreover, the right side is nonpositive because $\bar{x}_1$ is a solution of the convex optimization problem with differentiable f and the necessary optimality condition is given by [7]

$$\nabla f(\bar{x}_1)^\mathsf{T}(y - \bar{x}_1) \geq 0 \quad \text{for all } y \in \Omega.$$

Combining these two inequalities, we obtain

$$d^\mathsf{T} A d = 0 \;\wedge\; d^\mathsf{T} \nabla f(\bar{x}_1) = 0. \tag{17}$$

The first equality implies $d \in \operatorname{Ker} A$.

We suppose that both of the solutions belong to the feasible set; therefore, they satisfy constraint conditions. From equality conditions for $\bar{x}_2$ and using $B\bar{x}_1 = c$, we get

$$c = B\bar{x}_2 = B(\bar{x}_1 + d) = B\bar{x}_1 + Bd = c + Bd;$$

therefore, $Bd = 0$ or equivalently $d \in \operatorname{Ker} B$. □

In the proof of the previous lemma, we did not yet use one important property, which appears in (17). Using $d \in \operatorname{Ker} A$ from the first equality of (17), we get equivalent condition

$$d^\mathsf{T} \nabla f(\bar{x}_1) = d^\mathsf{T}(A\bar{x}_1 - b) = -d^\mathsf{T} b = 0$$

or equivalently (using $d \in \operatorname{Ker} A \cap \operatorname{Ker} B$; see Lemma 1)

$$b \perp \operatorname{Ker} A \cap \operatorname{Ker} B.$$

This condition forms the sufficient condition for the possible existence of two different solutions of the general QP problem (14).

However, these solutions could be constrained by additional inequality constraints and the full system of necessary conditions is more complicated. Let us introduce a Lagrange function [42; 14] corresponding to the problem (14):

$$\mathscr{L}(x, \lambda_E, \lambda_I) := \tfrac{1}{2} x^\mathsf{T} A x - b^\mathsf{T} x + \lambda_E^\mathsf{T}(Bx - c) - \lambda_I^\mathsf{T} x, \tag{18}$$

where λ_I and λ_E are Lagrange multipliers corresponding to the equality and inequality constraints. So-called Karush–Kuhn–Tucker (KKT) optimality conditions are given by

$$\begin{aligned} Ax - b + B^\top \lambda_E - \lambda_I &= 0, \\ Bx - c &= 0, \\ x, \lambda_I &\geq 0, \\ [x]_j [\lambda_I]_j &= 0 \quad \text{for all } j = 1, \dots, KT, \end{aligned} \tag{19}$$

where we use the notation $[v]_j$ to denote the j-th component of vector v.

Additionally, we can utilize the block-diagonal structure of our specific problem (10) given by the decomposition into clusters. Let us denote the block of matrix A by $\hat{A} \in \mathbb{R}^{T \times T}$ and corresponding blocks of vectors $x_k, b_k, \lambda_{Ik} \in \mathbb{R}^T$, $k = 1, \dots, K$. Then we can write the first KKT system in a form

$$\hat{A} x_k - b_k + \lambda_E + \lambda_{Ik} = 0, \quad k = 1, \dots, K.$$

Now we can sum all these equations to get

$$\hat{A} \left(\sum_{k=1}^K x_k \right) - \sum_{k=1}^K (b_k + \lambda_{Ik}) + K \lambda_E = 0,$$

and since from the equality constraint we have $\sum_{k=1}^K x_k = \mathbf{1}$ and $\mathbf{1} \in \operatorname{Ker} \hat{A}$ (see (13)), we can write

$$\lambda_E = \frac{1}{K} \sum_{k=1}^K (b_k + \lambda_{Ik}) \tag{20}$$

and substitute back into first KKT condition (19). Using the definition of a matrix B, we obtain

$$Ax - Qb - Q\lambda_I = 0, \quad Q := I - \frac{1}{K} B^\top B. \tag{21}$$

Here the orthogonal matrix $Q \in \mathbb{R}^{KT \times KT}$ represents the projector onto $\operatorname{Ker} B$.

Using KKT optimality conditions and the block structure of the problem, we are able to prove the following lemma, which gives the relationship between Lagrange multipliers corresponding to different solutions.

Lemma 2. *Let $\bar{x}_1, \bar{x}_2$ be two different solutions of the problem* (11) *and let $\bar{\lambda}_{1I}$, $\bar{\lambda}_{1E}$, $\bar{\lambda}_{2I}$, $\bar{\lambda}_{2E}$ be corresponding Lagrange multipliers in KKT system* (19). *Then*

$$\bar{\lambda}_{1I} = \bar{\lambda}_{2I} \quad \text{and} \quad \bar{\lambda}_{1E} = \bar{\lambda}_{2E}.$$

Proof. We have already shown that the Lagrange multipliers corresponding to equality constraints are uniquely given by the values of the Lagrange multipliers corresponding to the inequality constraints (20). Therefore, in the proof we will focus on a relationship between $\bar{\lambda}_{1I}$ and $\bar{\lambda}_{2I}$.

Let us consider two different solutions $\bar{x}_1$ and $\bar{x}_2$. These solutions satisfy all KKT conditions (19) and (21):

$$\begin{aligned} A\bar{x}_1 - Qb - Q\bar{\lambda}_{1I} &= 0, & A\bar{x}_2 - Qb - Q\bar{\lambda}_{2I} &= 0,\\ \bar{x}_1, \bar{\lambda}_{1I} &\geq 0, & \bar{x}_2, \bar{\lambda}_{2I} &\geq 0,\\ [\bar{x}_1]_j[\bar{\lambda}_{1I}]_j &= 0, & [\bar{x}_2]_j[\bar{\lambda}_{2I}]_j &= 0. \end{aligned}$$

Let us denote

$$\bar{x}_2 - \bar{x}_1 =: d, \qquad \bar{\lambda}_{2I} - \bar{\lambda}_{1I} =: p \tag{22}$$

and substitute this into the first KKT condition for $(\bar{x}_2, \bar{\lambda}_{2I})$:

$$A(\bar{x}_1 + d) - Qb - Q(\bar{\lambda}_{1I} + p) = 0. \tag{23}$$

Since $d \in \operatorname{Ker} A$ (see Lemma 1) and using the first KKT condition for $(\bar{x}_1, \bar{\lambda}_{1I})$, we can write (23) in the form

$$Qp = 0 \implies p \in \operatorname{Im} B^\top.$$

Now we focus on the inequality conditions. We use our notation (22) and substitute into KKT conditions (for all $j = 1, \dots, KT$):

$$\begin{aligned} [\bar{x}_2]_j - [d]_j &\geq 0, & [\bar{x}_1]_j + [d]_j &\geq 0,\\ [\bar{\lambda}_{2I}]_j - [p]_j &\geq 0, & [\bar{\lambda}_{1I}]_j + [p]_j &\geq 0. \end{aligned}$$

We multiply these inequalities by nonnegative numbers $[\bar{\lambda}_{2I}]_j$, $[\bar{\lambda}_{1I}]_j$, $[\bar{x}_2]_j$, $[\bar{x}_1]_j$ and use complementarity KKT conditions. We get

$$\begin{aligned} [d]_j[\bar{\lambda}_{2I}]_j &\leq 0, & [d]_j[\bar{\lambda}_{1I}]_j &\geq 0,\\ [p]_j[\bar{x}_2]_j &\leq 0, & [p]_j[\bar{x}_1]_j &\geq 0. \end{aligned}$$

Adding complementarity conditions with substitution (22) and using original complementarity conditions

$$\begin{aligned} -[d]_j[\bar{\lambda}_{2I}]_j - [p]_j[\bar{x}_2]_j + [d]_j[p]_j &= 0,\\ [d]_j[\bar{\lambda}_{1I}]_j + [p]_j[\bar{x}_1]_j + [d]_j[p]_j &= 0, \end{aligned}$$

we end up with

$$[d]_j[p]_j = 0 \quad \text{for all } j = 1, \dots, KT. \tag{24}$$

Let us remark that this condition is much stronger than $d^\top p = 0$, which could be obtained directly from

$$\begin{aligned} &d \in \operatorname{Ker} A \cap \operatorname{Ker} B,\\ &p \in \operatorname{Im} B^\top, \end{aligned}$$

using $\operatorname{Ker} B \perp \operatorname{Im} B^T$ [35].

Now we are ready to prove that $p = 0$. Since $p \in \operatorname{Im} B^\top$, there exists $\alpha \in \mathbb{R}^T$ such that

$$p_k = \alpha \quad \text{for all } k = 1, \dots, K.$$

Suppose by contradiction that there exists an index $i \in \{1, \dots, T\}$ such that $[\alpha]_i \neq 0$. Due to (24) corresponding components of d have to be zero, i.e.,

$$[d]_i = [d]_{i+T} = \cdots = [d]_{i+(K-1)T} = 0. \tag{25}$$

However, if we suppose that $d \neq 0$ (solutions $\bar{x}_1$ and $\bar{x}_2$ are different), then the vector d with property (25) is not from $\operatorname{Ker} A$ (see (13)), which is a contradiction. Therefore, $p = 0$. □

In the general case, we still cannot prove the uniqueness conditions. The following presents the situation when our problem (11) has an infinite number of solutions:

Lemma 3. *Let us consider the problem* (11) *with*

$$b \in \operatorname{Im} B^\top$$

and $K \geq 2$. Then this problem has an infinite number of solutions. Moreover, one of these solutions is given by

$$\bar{x} = \frac{1}{K}\mathbf{1}. \tag{26}$$

Proof. At first, we prove that (26) is a solution. This point is not on the boundary of a feasible set; therefore, we can ignore the inequality constraints — all of them are satisfied and correspondingly $\bar{\lambda}_I = 0$. The first KKT condition (21) is given by

$$Ax = Qb.$$

Since Q is an orthogonal projector onto $\operatorname{Ker} B$ and we suppose $b \perp \operatorname{Ker} B$, the right-hand side of this equation is equal to 0. Notice that $\bar{x} \in \operatorname{Ker} A$; therefore, the left-hand side is also equal to zero and the first KKT condition is satisfied. Moreover, the equality constraint could be also easily checked; therefore, $\bar{x}$ is solution.

Now it remains to show that there exists at least one additional solution, i.e., that there exists a nonzero vector $d \in \operatorname{Ker} A \cap \operatorname{Ker} B$ such that

$$\bar{x} + d \in \Omega.$$

Since we suppose $K \geq 2$, the vector space $\operatorname{Ker} A \cap \operatorname{Ker} B$ is nontrivial and it is possible to choose a nonzero vector from this space. Additionally, $\bar{x}$ does not belong to the boundary of Ω; therefore, there exists a nonzero vector in any *direction*. □

3. Spectral projected gradient method for QP problems

There exist several types of algorithms for solving a general QP problem (14). Since the manipulation with both constraint types is usually difficult, these algorithms are based on elimination of one type of constraint in the outer loop, and then they solve the sequence of inner problems with the remaining type of constraint.

To be more specific, one can use popular interior-point (IP) methods [42; 51]. These methods enforce the inequality constraints using a barrier function:

$$\min_{Bx=c,\, x\geq 0} f(x) \approx \min_{Bx=c} f(x) + \mu \sum_{i=1}^{T} \log x_i .$$

Here $\mu > 0$ represents the barrier parameter. Using this approach, the algorithm transforms the original KKT system with inequalities (19) to the system of nonlinear equations with so-called duality gap μ. The new problem is not equivalent to the original, but as the duality gap approaches zero, it becomes a better and better approximation. The barrier function increases all function values near the boundary of the feasible set to create an impenetrable barrier for a step-based algorithm which solves the inner problem with the remaining equality constraints. Usually μ is not implemented as a constant but is a sequence $\mu_k \to 0$, and the solution of a previous inner problem is used as an initial approximation of the inner algorithm for a new μ_{k+1}. Since the nonquadratic term (logarithm) is added to the original quadratic function f, the corresponding KKT system of this inner problem is nonlinear. To solve this system, one can use Newton-type methods with step size control to be sure that the new step will not jump through the barrier. Newton-type methods for solving nonlinear systems introduce the sequence of linear equations which have to be solved. For more details, see [42; 51].

Another approach is to deal with equality constraints first. This can be performed using augmented Lagrangian methods [42; 14]. The algorithm enforces the equality constraints using a penalty term:

$$\min_{Bx=c,\, x\geq 0} f(x) \approx \min_{x\geq 0} f(x) + \rho \|Bx - c\|^2 .$$

Again, the new problem is not equivalent to the original one, although if $\rho \to \infty$, then $\|Bx - c\| \to 0$. Therefore, the penalty parameter ρ is usually implemented as the increasing sequence. The main advantage of this approach is the structure of the inner problem with inequalities — this new problem is again a QP. However, the Hessian matrix of the new problem is given by

$$\nabla^2 (f(x) + \rho \|Bx - c\|^2) = A + \rho B^{\mathsf{T}} B;$$

therefore, since the condition number depends on the value ρ and while $\rho \to \infty$, the problem becomes harder and harder to solve. Similar to the interior-point methods, there exists extensive theory about the connection between the stopping criterion for solving ill-conditioned inner problems and the value of a penalty parameter; see the semimonotonic augmented Lagrangian algorithm of [14] or [16]. The inner QP problem can be solved using the interior-point method, active-set algorithms, or projected gradient methods. In the case of bound constraints, the projection onto a feasible set is trivial. Therefore, in the case of the active-set algorithm and projected gradient methods, the inequality constraints are satisfied accurately, which is not the case for the interior-point methods. The barrier function makes it impossible to find the solution on the boundary of a feasible set, because in that case the value of a barrier term is equal to infinity.

This property brings us to the main disadvantage of both approaches. In the case of interior-point methods, it is impossible to satisfy inequality constraints accurately. In the case of the penalization technique, it is impossible to satisfy equality constraints exactly.

Fortunately, our QP problem (10) is not a general QP (14). To be more specific, the feasible set in our case is a separable set composed of T simplexes of size K. There exists an efficient algorithm for computing the projection of a general point onto simplex

$$\begin{aligned}
&P_{\Omega_t}(y) := \arg\min_{x \in \Omega_t} \| y - x \|, \\
&\Omega_t := \left\{ \gamma_t \in \mathbb{R}^K : \gamma_t \geq 0 \wedge \sum_{k=1}^{K} [\gamma_t]_k = 1 \right\}, \quad t = 1, \dots T, \\
&\gamma_t := [\gamma_1(t), \dots, \gamma_K(t)]^\top.
\end{aligned}$$

The algorithm was presented in [10], and it computes the projection onto the simplex of dimension K in at most K steps; see Algorithm 2.

If we use these projections in our algorithm, all constraint conditions will be satisfied accurately. Moreover, computation of the projection can be performed as T independent processes, and since T is the largest parameter of problem dimension, this approach increases the granularity of the overall solution process — making it suitable for GPU computation. We can also use the value of the original cost function to stop our algorithm with respect to sufficient decrease given by demands from Algorithm 1.

Therefore, we are mainly interested in the projected gradient descent methods, i.e., in the algorithms which use $x^0 \in \Omega$ as an initial approximation and suitable step lengths $\alpha_k > 0$ to generate the approximations of the solution by

$$x^{k+1} = P_\Omega(x^k - \alpha_k \nabla f(x^k)), \quad k = 0, 1, \dots. \tag{27}$$

```
Given arbitrary $y \in \mathbb{R}^K$
    **if** $K = 1$ **then** set $P(y) := 1$ and **stop**
    Sort $y$ in ascending order and set $k := K - 1$
    **while** $k > 0$
        $\alpha := \big(\sum_{j=k+1}^{K} [y]_j - 1\big)/(K - k)$
        **if** $\alpha > [y]_k$ **then** $\hat{\alpha} := \alpha$ and $k := -1$
        **else** $k := k - 1$
    **end while**
    **if** $k = 0$ **then** $\hat{\alpha} := \big(\sum_{j=1}^{K} [y]_j - 1\big)/K$
    Set $[P(y)]_k := \max\{[y]_k - \hat{\alpha}, 0\}$ for all $k = 1, \ldots, K$
Return $P(y)$
```

Algorithm 2. Projection onto simplex [10].

In this case, the feasibility of generated approximations is enforced by using the projections, and the descent of the object function is induced by using $-\nabla f(x)$, which is generally the best local decrease direction. One of the most efficient projection gradient methods is a spectral projected gradient method (SPG) [5]. The first part of one SPG iteration is based on generating the point using (27) with step size defined by the Barzilai–Borwein algorithm (BB) [3]. However, the projected variant of BB is not convergent in general and the original proof of convergence cannot be applied [12] and an additional line-search technique has to be implemented.

In this section, we shortly review both components of SPG, i.e., the projected BB method and additional generalized Armijo condition. The method was developed to solve general optimization problems, and in this paper, we add our own modification for solving QP problems using the properties of the quadratic objective function.

Let us follow [45]. BB is the nonmonotone gradient descent method for solving unconstrained convex optimization problems. These methods are based on a construction of a sequence of solution approximations using the recursive formula

$$x^{k+1} = x^k - \alpha_k g^k, \quad k = 0, 1, \ldots, \tag{28}$$

with a step size $\alpha_k \in \mathbb{R}^+$ and a vector of steepest descent $-g^k := -\nabla f(x^k)$. The most popular gradient descent method is the steepest descent method (SD, first presented by Cauchy [9]). This method uses the step length, which minimizes the function $f(x^{k+1})$ using locally optimal step size

$$\alpha_k = \arg\min_{\alpha \in \mathbb{R}} f(x^k - \alpha \nabla f(x^k)) = \frac{\langle g^k, g^k \rangle}{\langle A g^k, g^k \rangle}, \tag{29}$$

which leads to the monotone descent of the objective function.

However, the step size of BB is based on a different idea. To briefly review the relation of BB for the solution of unconstrained problems to Newton's method for solving a scalar nonlinear equation

$$g(x) = 0, \quad g : \mathbb{R} \to \mathbb{R},$$

let us replace the derivative $g'(x^k)$ in Newton's method by its secant approximation to get

$$x^{k+1} = x^k - \frac{1}{g'(x^k)} g(x^k) \approx x^k - \frac{x^k - x^{k-1}}{g(x^k) - g(x^{k-1})} g(x^k). \tag{30}$$

Denoting $g^k = g(x^k) = f'(x^k) = \nabla f(x^k)$ and

$$\alpha_k = \frac{x^k - x^{k-1}}{g^k - g^{k-1}}, \tag{31}$$

we can see that the secant method (30) can be considered as a gradient descent method (28). If $g(x) : \mathbb{R}^n \to \mathbb{R}^n$, then we cannot evaluate α_k by (31), but we can assemble the *secant equation*

$$\frac{1}{\alpha_k}(x^k - x^{k-1}) = g^k - g^{k-1}. \tag{32}$$

After denoting

$$s^k = x^k - x^{k-1}, \qquad g^k - g^{k-1} = As^k$$

and solving (32) in the least-square sense

$$\alpha_k = 1/\arg\min_{\beta \in \mathbb{R}} (\langle s^k, s^k \rangle \beta^2 - 2\langle As^k, s^k \rangle \beta + \langle As^k, As^k \rangle)$$

and some simplifications, we get

$$\alpha_k = \frac{\langle s^k, s^k \rangle}{\langle As^k, s^k \rangle}. \tag{33}$$

This is the step size of BB. The proof of convergence with estimates for solving the unconstrained QP problem (i.e., (14) with $\Omega = \mathbb{R}^n$) was presented in [13].

However, the projected variant of BB (i.e., (27) with (33)) is not convergent in general and the original proof of convergence cannot be applied [12]. One option how to enforce the convergence of the method is to use an additional line-search. Let us denote the difference

$$g^P_{\alpha_k}(x^k) = P_\Omega(x^k - \alpha_k \nabla f(x^k)) - x^k \tag{34}$$

as a *projected gradient* at point $x^k \in \Omega$ with the step length $\alpha_k > 0$. To enforce the convergence, the SPG algorithm uses an additional line-search step

$$x^{k+1} = x^k + \beta_k d_k \tag{35}$$

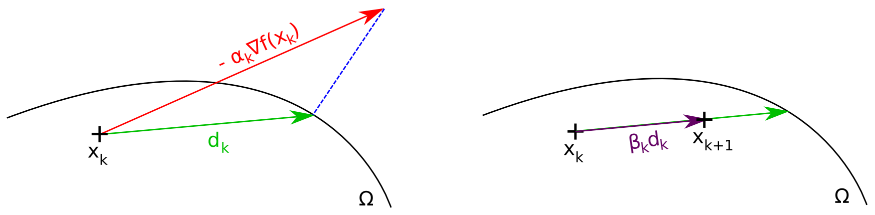


Figure 1. Projected gradient descent method with the additional line-search in two steps. Left: computation of projected gradient. Right: an additional line-search.

with $d_k := g^P_{\alpha_k}(x^k)$ and an *appropriate* choice of the step size $\beta_k \in (0, 1]$. The method with these two steps (computation of the projected gradient and an additional line-search) is demonstrated in Figure 1.

The next lemma demonstrates the reason for using the projected gradient as a search direction in (35): it is a descent direction.

Lemma 4. *Let* $x \in \Omega$, $\alpha > 0$, *and* $g^P_\alpha(x) = P_\Omega(x - \alpha \nabla f(x)) - x \neq 0$. *Then*

$$\langle g^P_\alpha(x), \nabla f(x) \rangle < 0. \tag{36}$$

Proof. We suppose $\Omega \subset \mathbb{R}^n$ is a closed convex set; therefore [4],

$$\langle P_\Omega(y) - P_\Omega(z), y - z \rangle \geq \| P_\Omega(y) - P_\Omega(z) \|^2 \quad \text{for all } y, z \in \mathbb{R}^n.$$

If we choose $y = x - \alpha \nabla f(x)$ and $z = x = P_\Omega(x)$, then we can estimate

$$\begin{aligned} -\alpha \langle g^P_\alpha(x), \nabla f(x) \rangle &= \langle g^P_\alpha(x), (x - \alpha \nabla f(x)) - x \rangle \\ &\geq \| g^P_\alpha(x) \|^2 > 0. \end{aligned}$$ □

The SPG algorithm uses the Grippo–Lampariello–Lucidi method (GLL) [21] to find appropriate step size β_k. This algorithm is based on a bisection method to satisfy the so-called *generalized Armijo condition*

$$f(x^k + \beta_k d^k) < f_{\max} + \tau \beta_k \langle \nabla f(x^k), d^k \rangle. \tag{37}$$

Here $\tau \in (0, 1)$ represents a safeguarding parameter and

$$f_{\max} := \max\{ f(x^{k-j}) : 0 \leq j \leq \min\{k, m-1\}\}.$$

The main difference between this generalized version and the original Armijo conditions [42] is in the utilization of function values in m previous approximations instead of using only the previous $f(x^{k-1})$. This approach supplies the nonmonotonic behavior of BB and at the same time controls the descent. The proof of convergence of SPG is based on satisfying the generalized Armijo condition in every step [5]. We present SPG by Algorithm 3 and GLL by Algorithm 4.

```
Given cost function $f : \mathbb{R}^n \to \mathbb{R}$, initial approximation $x^0 \in \Omega$, projection onto feasible set
$P_\Omega(x)$, safeguarding parameters $0 < \alpha_{\min} \ll \alpha_{\max}$, precision $\varepsilon > 0$, and initial step size
$\alpha_0 > 0$
    $k := 0$
    while $\| P_\Omega(x^k - \nabla f(x^k)) - x^k \| > \varepsilon$
        $d^k := P_\Omega(x^k - \alpha_k \nabla f(x^k)) - x^k$
        Compute step size $\beta_k$ using GLL
        $x^{k+1} := x^k + \beta_k d^k$
        $s^k := x^{k+1} - x^k$
        $y^k := \nabla f(x^{k+1}) - \nabla f(x^k)$
        if $\langle s^k, y^k \rangle \leq 0$ then
            $\alpha_{k+1} := \alpha_{\max}$
        else
            $\alpha_{k+1} := \min\{\alpha_{\max}, \max\{\alpha_{\min}, \langle s^k, s^k \rangle / \langle s^k, y^k \rangle\}\}$
        end if
        $k := k + 1$
    end while
Return approximation of solution $x^{k+1}$
```

Algorithm 3. The original SPG method [6].

The main bottleneck of GLL is the computational complexity, which cannot be estimated in general. To be more specific, it is hard to say when the bisection will be finished.

The SPG was developed to solve more general optimization problems on convex sets. In our problems, the cost function is a quadratic function. We can use its particular form and its properties to simplify the GLL algorithm, obtaining an algorithm with fewer cost function evaluations, i.e., with a smaller number of the most time-consuming operations — multiplications by the Hessian matrix A. The motivation came from the other well known algorithms for solving QP, like the steepest descent method and the conjugate gradient method [26].

This modification was initially presented in [43], and it reduces the bisection in GLL to a simple formula.

First, let us present the basic equality in QP [14]: (for all $x, d \in \mathbb{R}^n$ and $b \in \mathbb{R}$)

$$f(x + \beta d) = f(x) + \beta \langle Ax - b, d \rangle + \tfrac{1}{2}\beta^2 \langle Ad, d \rangle. \tag{38}$$

We start the simplification of SPG for solving QP problems with the most obvious simplifications. Notice that in the Algorithm 3 we can write

$$\begin{aligned} y^k &= \nabla f(x^{k+1}) - \nabla f(x^k) = (Ax^{k+1} - b) - (Ax^k - b) \\ &= A(x^{k+1} - x^k) = As^k. \end{aligned}$$

```
Given cost function $f : \mathbb{R}^n \to \mathbb{R}$, parameter $m \in \mathbb{N}$, approximation and direction
$x^k, d^k \in \mathbb{R}^n$, parameter $\tau \in (0,1)$, safeguarding parameters $0 < \sigma_1 < \sigma_2 < 1$ for $\sigma_1, \sigma_2 \in \mathbb{R}$
    $f_{\max} := \max\{f(x^{k-j}) : 0 \le j \le \min\{k, m-1\}\}$
    $x_{\text{temp}} := x^k + d^k$
    $\delta := \langle \nabla f(x^k), d^k \rangle$
    $\beta := 1$
    while $f(x_{\text{temp}}) > f_{\max} + \delta\beta\delta$
        $\beta_{\text{temp}} := -\frac{1}{2}\beta^2\delta/(f(x_{\text{temp}}) - f(x^k) - \beta\delta)$
        if $\beta_{\text{temp}} \in \langle \sigma_1, \sigma_2\beta \rangle$ then
            $\beta := \beta_{\text{temp}}$
        else
            $\beta := \beta/2$
        end if
        $x_{\text{temp}} := x^k + \beta d^k$
    end while
Return step size $\beta$
```

Algorithm 4. GLL line-search [21].

Since matrix A is SPS, we can write for any $s^k \in \mathbb{R}^n \setminus \operatorname{Ker} A$

$$\langle s^k, y^k \rangle = \langle s^k, As^k \rangle > 0,$$

and the condition in SPG (Algorithm 3) is always satisfied.

Moreover, the BB step length (33) is the inverse Rayleigh quotient (with $s^k \notin \operatorname{Ker} A$) and it can be bounded by

$$\frac{1}{\lambda_{\max}} \le \alpha_{k+1} \le \frac{1}{\hat{\lambda}_{\min}},$$

where $\hat{\lambda}_{\min}$ is the smallest nonzero eigenvalue and $\lambda_{\max}$ is the largest eigenvalue of the matrix A (in our case, the Hessian matrix is SPS; see (9) and (13)). Therefore, we can omit the safeguarding parameters $\alpha_{\min}$ and $\alpha_{\max}$ in Algorithm 3.

Let us take a better look into GLL line-search (Algorithm 4). The computation of β_{temp} can be simplified using (38). We obtain

$$\begin{aligned}\beta_{\text{temp}} &:= -\frac{\beta^2\delta}{2(f(x^k+\beta d^k) - f(x^k) - \beta\delta)} \\ &= -\frac{\beta^2\delta}{2\beta\delta + \beta^2\langle Ad^k, d^k\rangle - 2\beta\delta} = -\frac{\langle \nabla f(x^k), d^k\rangle}{\langle Ad^k, d^k\rangle} =: \bar{\beta}.\end{aligned}$$

This is a simple Cauchy step size (29). Since the vector d^k is the descent direction (36) and A is SPS, our optimal $\bar{\beta}$ is positive. Obviously, the computation of a new β_{temp} is independent of the previous value and the original GLL method solely

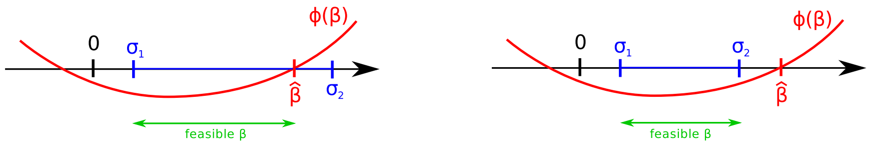


Figure 2. Possible situations in a GLL condition for QP.

performs the bisection method, i.e., tries to halve the coefficient β and verify the generalized Armijo condition. Furthermore, the value of a step size β has to be from the interval $[\sigma_1, \sigma_2] \subseteq [0, 1]$, because a smaller or larger value may cause a departure from the feasible set.

The division of step size β by 2 now modifies only the generalized Armijo condition. This condition can be also simplified as

$$\begin{aligned}
0 &> f(x^k + \beta d^k) - f_{\max} - \tau\beta\delta \\
&= f(x^k) + \beta\langle \nabla f(x^k), d^k\rangle + \tfrac{1}{2}\beta^2\langle Ad^k, d^k\rangle f_{\max} - \tau\beta\langle \nabla f(x^k), d^k\rangle \\
&= \tfrac{1}{2}\beta^2\langle Ad^k, d^k\rangle + (1-\tau)\beta\langle \nabla f(x^k), d^k\rangle + f(x^k) - f_{\max}, \\
0 &> \tfrac{1}{2}\beta^2 + (1-\tau)\beta\frac{\langle \nabla f(x^k), d^k\rangle}{\langle Ad^k, d^k\rangle} + \frac{1}{\langle Ad^k, d^k\rangle}(f(x^k) - f_{\max}).
\end{aligned}$$

Afterwards, we denote the function on the right-hand side and the constant term by

$$\Phi(\beta) := \tfrac{1}{2}\beta^2 - (1-\tau)\bar{\beta}\beta - \xi, \qquad \xi := \frac{1}{\langle Ad^k, d^k\rangle}(f_{\max} - f(x^k)).$$

We are interested in β such that the generalized Armijo condition in a form

$$\Phi(\beta) < 0 \tag{39}$$

is satisfied. The positive root of $\Phi(\beta)$ is given by

$$\hat{\beta} := (1-\tau)\bar{\beta} + \sqrt{(1-\tau)^2\bar{\beta}^2 + 2\xi}.$$

There exist only two possible situations; see Figure 2. Therefore, we can conclude that the feasible step size in the second step of SPG could be defined as

$$\beta_k \in [\sigma_1, \min\{\sigma_2, \hat{\beta}\}].$$

This simple interval can replace GLL; i.e., any β_k from this interval satisfies the generalized Armijo condition.

The computation of the function values can be also simplified using (38):

$$f(x^{k+1}) = f(x^k) + \beta_k\langle g^k, d^k\rangle + \tfrac{1}{2}\beta_k^2\langle Ad^k, d^k\rangle.$$

```
Given cost function $f : \mathbb{R}^n \to \mathbb{R}$, initial approximation $x^0 \in \Omega$, projection onto feasible set
$P_\Omega(x)$, parameters $m \in \mathbb{N}$, $\tau \in (0, 1)$, safeguarding parameters $0 < \sigma_1 < \sigma_2 < 1$ for
$\sigma_1, \sigma_2 \in \mathbb{R}$, precision $\varepsilon > 0$, and initial step size $\alpha_0 > 0$
    $k := 0$
    $g^0 := Ax^0 - b$
    $f^0 := \frac{1}{2}\langle g^0 - b, x^0 \rangle$
    **for** $k = 0, 1, \dots$
        $d^k := P_\Omega(x^k - \alpha_k g^k) - x^k$
        Compute matrix-vector multiplication $Ad^k$
        Compute multiple dot-product $\langle d^k, \{d^k, Ad^k, g^k\} \rangle$
        **if** $\sqrt{\langle d^k, d^k \rangle} \le \varepsilon$ **then stop**
        $f_{\max} := \max\{f(x^{k-j}) : 0 \le j \le \min\{k, m-1\}\}$
        $\xi := (f_{\max} - f^k)/\langle d^k, Ad^k \rangle$
        $\bar{\beta} := -\langle g^k, d^k \rangle / \langle d^k, Ad^k \rangle$
        $\hat{\beta} := \tau\bar{\beta} + \sqrt{\tau^2\bar{\beta}^2 + 2\xi}$
        Choose $\beta_k \in \langle \sigma_1, \min\{\sigma_2, \hat{\beta}\} \rangle$
        $x^{k+1} := x^k + \beta_k d^k$
        $g^{k+1} := g^k + \beta_k Ad^k$
        $f^{k+1} := f^k + \beta_k \langle d^k, g^k \rangle + \frac{1}{2}\beta_k^2 \langle d^k, Ad^k \rangle$
        $\alpha_{k+1} := \langle d^k, d^k \rangle / \langle d^k, Ad^k \rangle$
        $k := k + 1$
    **end for**
Return approximation of solution $x^k$
```

Algorithm 5. SPG for QP problems (SPG-QP).

Finally, we can simplify the computation of the BB step length using (35) to

$$\alpha_{k+1} = \frac{\langle s^k, s^k \rangle}{\langle s^k, y^k \rangle} = \frac{\langle s^k, s^k \rangle}{\langle s^k, As^k \rangle} = \frac{\langle \beta_k d^k, \beta_k d^k \rangle}{\langle \beta_k d^k, \beta_k Ad^k \rangle} = \frac{\langle d^k, d^k \rangle}{\langle d^k, Ad^k \rangle}$$

and using the recursive formula for the computation of a new gradient:

$$g^{k+1} := Ax^{k+1} - b = A(x^k + \beta_k d^k) - b = g^k + \beta_k Ad^k.$$

We use all these simplifications to form Algorithm 5. For the sake of simplicity, we relabel the coefficient $\tau := 1 - \tau \in (0, 1)$.

Notice that the most time-consuming operation — multiplication by Hessian matrix A — is performed only once per iteration. Moreover, all scalar products in every iteration can be performed as a single operation. This feature decreases the amount of global communication during the solution process.

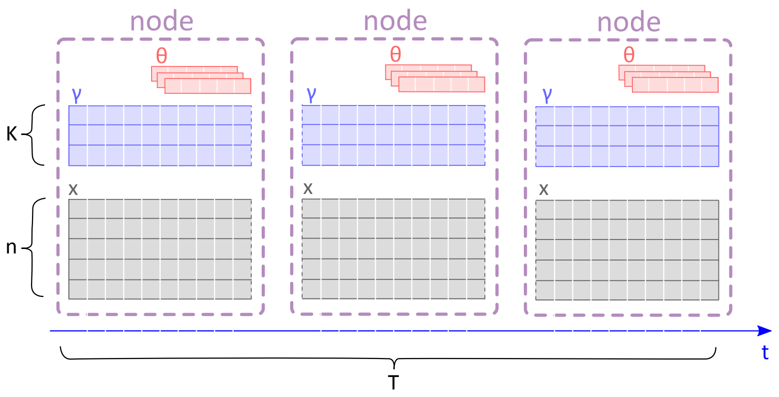


Figure 3. Large global vector of time series data X is distributed into nodes in a natural time-splitting way. Moreover, each node owns the part of global vector γ which corresponds to the time part of local data. Since the number of model parameters Θ is small, each node owns its own local copy.

4. HPC implementation

We are developing and maintaining a new HPC library [44] for nonstationary time series analysis in C++ using PETSc [2]. This library supports the manipulation of vectors distributed on multiple nodes. These data can be used during computation on CPUs or GPUs. Therefore, the user of our library can decide which architecture will be used for computation.

The solution of our problem consists of two different types of parallelization. The first type is straightforward: the problem (9) has to be solved for several values of regularization parameters ϵ^2 and various numbers of clusters K. Moreover, the solution obtained by the iterative process depends on an initial approximation. Each combination of these parameters can be used to run a completely independent instance of Algorithm 1. The parallelization in this case is *embarrassingly parallel*.

A more complicated parallelization scheme has to be used in a case where one instance of Algorithm 1 cannot be run on a single node, because of the size of the input data and/or the size of the unknowns, especially the size of vector γ. To deal with this problem, we naturally distribute the given long time series data $X \in \mathbb{R}^{T \times n}$ into nodes as successive disjoint time subintervals with approximately the same size; see Figure 3.

Decomposition in time plays a key role in the effective computation of projections used in SPG-QP; see Algorithm 5. Using our approach, all data of one simplex are stored in one particular node; therefore, the projection is computed on each node fully independently. However, this decomposition in time brings new difficulties like disruption of diagonal-block structure of A defined in analysis in form (10); therefore, we had to implement an additional local-to-global index recomputation.

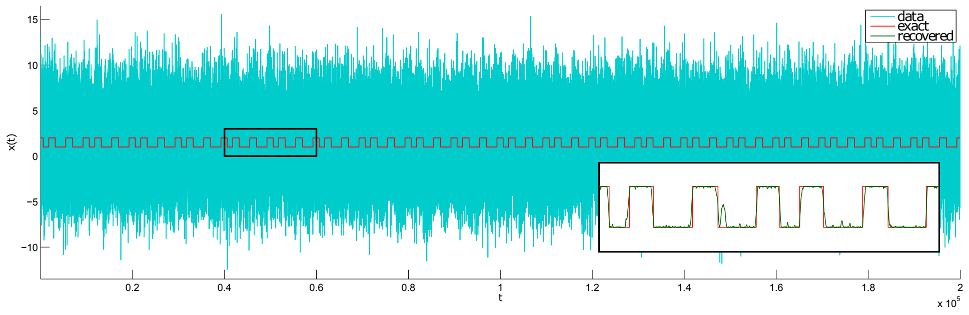


Figure 4. The beginning part of the benchmark signal of total length $T = 10^7$. Here the black box represents a part of the signal which is further enlarged to present the difference between the original signal (red) and the solution, namely the denoised signal (green). This denoised signal was obtained using optimal penalty parameter $\varepsilon^2 = 3000$ (obtained with the standard L-curve method [22]) with norm of absolute error 0.04.

5. Numerical experiments

To demonstrate the scalability of our QP solver, we consider a time series K-means clustering problem. This problem is characterized by the most basic modeling functions: the piecewise-constant functions. We are trying to model the given data using the constant mean value in every cluster in least-square sense with the FEM-H1 regularization penalty in time:

$$\text{for all } t \in T_k, \quad x_t = \theta_k + \epsilon_t,$$

$$L_\varepsilon(\theta_1, \dots, \theta_K, \Gamma) = \sum_{t=0}^{T} \sum_{k=1}^{K} \gamma_k(t)(x_t - \theta_k)^2 + \varepsilon^2 \sum_{k=1}^{K} \sum_{t=0}^{T-1} (\gamma_{k,t+1} - \gamma_{k,t})^2.$$

As a benchmark, we take a short signal of length 10^4 and repeat this short signal to obtain long time series $X_{\text{exact}}(t)$. This long signal is considered to be an *exact* benchmark solution of our denoising algorithm. As an input of our problem we consider the signal with a variable noise ε:

$$X(t) = X_{\text{exact}} + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, 10). \tag{40}$$

Figure 4 presents a beginning part of the considered long signal of length $T = 10^7$.

We provide the exact parameter solution $K = 2$, $\theta_1 = 1$, $\theta_2 = 2$ and solve the pure QP problem that represents the computational bottleneck of this time series denoising procedure. In SPG-QP, we choose $\tau = 0.9$ and $m = 20$ with respect to original SPG recommendations [5]. As a stopping criterion we choose the Euclidean norm of a projected gradient:

$$\|g^P(x)\| := \|P(x - \alpha \nabla f(x)) - x\| < 10^{-6}.$$

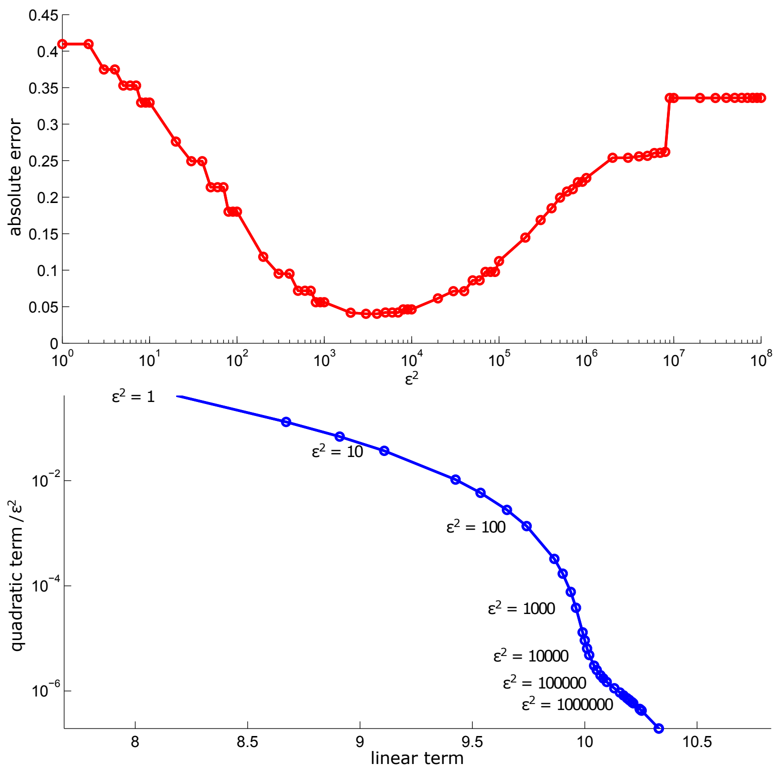


Figure 5. The mean value of L_1 norm of absolute error for several values of penalty parameter (top). The left part of the graph represents the error which arises due to insufficient regularization, and the right part is caused by too much regularization, for the beginning part of benchmark signal of total length $T = 10^7$. The L-curve (bottom) presents the relation between values of the linear term (modeling error) and quadratic term (regularization) which depends on the choice of regularization parameter.

We implement Algorithms 1 and 5 in the PETSc framework and solve this problem for several values of penalty parameter ε; see Figure 5. Standard methods like L-curve [22] are then used to identify the optimal values for ε. Based on these results, we choose value $\varepsilon = 3000$ for the following scalability tests.

To demonstrate the scalability of our implementation, we solve the abovementioned problem of parameters $T = 10^7$, $K = 2$, $\varepsilon^2 = 3000$ with $\theta_1 = 1$ and $\theta_2 = 2$ on CSCS Piz Daint using $N \in \{1, 2, 4, 8, 16, 32, 64\}$ nodes. For complete specifications of this machine, see http://www.cscs.ch/computers/piz_daint/. For GPU computation, we run one MPI process per hybrid node (Intel Xeon E5-2690 v3 with NVIDIA Tesla P100), which uses the GPU for computation. In the case of CPU, we use pure CPU-nodes ($2 \times$ Intel Xeon E5-2695, each with 18 cores) and run 36 MPI processes per node. PETSc deliberately chose not to support a multithreaded model, but rather only a multitask model (i.e., multiple MPI processes). Since we

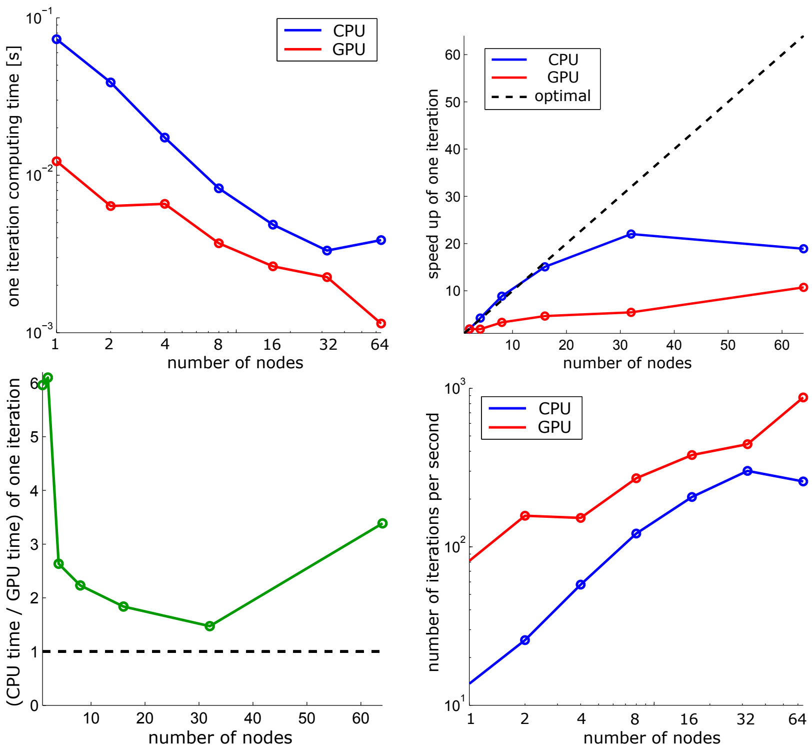


Figure 6. The strong scalability results: solution of QP problem of size $2 \cdot 10^7$ on Piz Daint using different architectures — the computation time of one iteration (upper right), relative speed-up of computation time of one iteration (upper left), the number of iterations per second (lower left), and computation times on CPU and GPU (lower right).

are using PETSc, we are left with no choice and we do not use any CPU-thread parallelization (e.g., OpenMP) in our implementation.

We generated a time series signal of the length 10^7 and denoise it on different numbers of nodes. For statistical reasons, we decided to focus on the average numbers of QP iterations — where averaging was performed over different numbers of involved nodes. Please see the computation time of one iteration and number of iterations per second provided in Figure 6. Here we can observe good scalability of CPU for a small number of nodes. However, the problem is too small for larger number of CPUs or GPUs; therefore, the speedup is rapidly decreasing due to MPI communication. In all cases, the GPU computation is faster, but in this case, we should also consider the energy consumption; see Figure 7. These values were computed using the technique presented in [18].

Next, we would like to compare the introduced SPG-QP method for solving the inner QP problem with other existing HPC open-source implementations. The most straightforward choice is to use methods already implemented in PETSc,

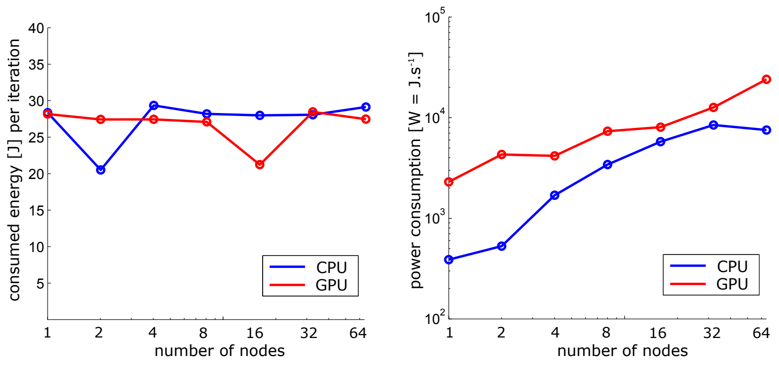


Figure 7. The energy consumption for one iteration of SPG-QP (left) and the power consumption (right).

for instance the Toolkit for Advance Optimization (TAO) [40]. However, for the combination of linear equalities and bound constraints (8), one can only use the interior-point method. Unfortunately, the actual manual pages say that the state-of-the-art implementation of the interior-point method in TAO is more of a placeholder for future constrained optimization algorithms and should not yet be used for large problems or production code. The most promising implementation of QP solvers in PETSc is PERMON (http://permon.it4i.cz/). The results for solving QP problems arising in linear elasticity contact problems [24] suggest good scalability performance and efficiency in the case of massively parallel CPU computation. However, our problem (10) has particular properties, especially the null spaces of the Hessian matrix and the matrix of linear equality constraints being not disjoint; see Lemma 1. Therefore, in our case the QP problem could have more then one solution. Besides that, we have tried to solve our problem with PERMON, but the algorithm is not able to solve problems with larger dimensions $T > 10^4$. Also the parameters of the algorithm have to be more precisely investigated and their tuning is nontrivial. Luckily, authors are working on generalization of the inner solver for solving the problems with singular Hessian matrices [15] and this approach should be implemented soon. Moreover, the GPU implementation is also future work. Summarizing these experiences, we were not able to find any QP solver that would be applicable to the particularly structured problem (10) of time series analysis — when the time series data is big (i.e., when $T \gg 10^4$). For a complete survey of existing HPC QP libraries, see [23].

Next, we also compare the introduced FEM-H1 with implementations of standard denoising methods. We were not able to find any HPC open-source library which includes the HPC implementation of the denoising methods for time series on hybrid architectures. In our implementation, we decided to use PETSc for basic vector/matrix operations; therefore, we are directly able to switch between CPU/GPU

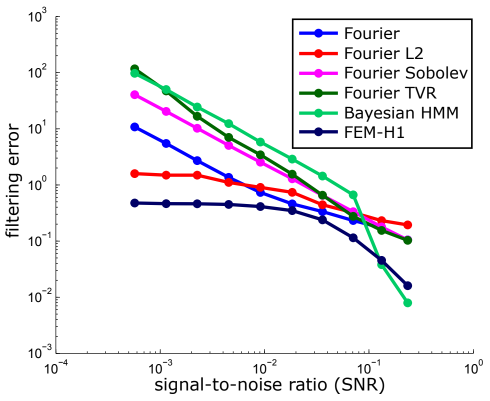


Figure 8. Dependence of the mean filtering error from the original signal-to-noise ratio (SNR). Results were obtained for the signal with Gaussian noise by averaging 100 independent realizations of the noisy signal data for each of the methods.

computations. Existing denoising libraries (e.g., Dlib C++ [34]) usually implement these operations from scratch using their own code, including basic vector/matrix operations, or (in the best case) use only sequential external BLAS libraries and cannot run on distributed memory architectures. Therefore, we decided to compare the denoising efficiency only for relatively short signals using the uniform sequential implementations in MATLAB [41]. To be more specific, we compare the following denoising algorithms.

- *Fourier* uses the MATLAB implementation of fft/ifft and has one parameter s, which defines the size of the window.
- *Fourier L_2* is Fourier filtering with an additional L_2 regularization term. There are two parameters: s as a size of the window and λ as a regularization parameter.
- *Fourier Sobolev* is Fourier filtering with an additional Sobolev prior penalty. This algorithm uses two parameters: size of the window s and regularization parameter λ.
- *Fourier TVR* is Fourier transformation with total variation regularization. The method leads to the system of nonlinear equations, which is typically solved using gradient method with constant step size and monotone descent of the solution error. The method uses three parameters: size of the window s and regularization parameters λ, ε.
- *Bayesian HMM* is the Bayesian hidden Markov model method [41]. This machine-learning algorithm uses random initial guesses; therefore, we run it 10 times and take the best solution with respect to the absolute error.

We have generated a large set of abovementioned testing signals of length $T = 1000$. For every standard deviation

$$\sigma \in 0.1 \cdot \{2, 8, 32, 128, 512, 2048, 8192, 32768, 131072, 524288\},$$

we generated 100 signals with random noise using formula (40) and then denoised the signal using the proposed algorithms. We set various algorithm parameters $s \in \{5, 20, 30, 40, 60, 80\}$, $\lambda \in \{0.1, 1, 10, 50, 100\}$, $\varepsilon \in \{0.001, 0.01, 0.1\}$ and consider only the best solution with respect to the absolute norm computed as a difference between the denoised signal and the exact signal X_{exact}. Similarly for FEM-H1, we solved the problem for various values of penalty parameter.

At the end of the solution process, we computed the average absolute error value through all random noises. The results are presented in Figure 8. Here, the signal-to-noise ratio (SNR) was computed as the ratio of the maximum variation of the true signal to the maximum variation of the noisy signal.

As can be seen from Figure 8, FEM-H1 methodology outperforms the standard methods when the signal-to-noise ratio becomes smaller (i.e., when the noise is becoming larger as compared to the true underlying signal).

6. Conclusion

In this paper, we introduced an extension of the nonparametric FEM-H1 framework allowing it to be applied to denoising of very large time series data sets. We investigated basic properties of the inner large-scale QP subproblem — being the most expensive part of the FEM-H1 nonstationary time series analysis methodology. To solve this problem with HPC, we presented a modification of the spectral projected gradient method for QP problems. This method is based on projections, enjoys high granularity of parallelization, and is suitable to run on GPU clusters, such as Piz Daint at the Swiss Supercomputing Centre (CSCS). We presented numerical results for solving a large-scale time series denoising problem.

In future work, we will compare SPG-QP with state-of-the-art parallel implementations of popular optimization methods for solving not only benchmarks, but also solving problems in practical applications, such as the inference of causality networks from multiscale economical data. Additionally, our code will be extended by spatial regularization to increase the number of data analysis applications which could be practically solved by our emerging open-source HPC library.

7. Acknowledgments

This work is supported by the Platform for Advanced Scientific Computing (PASC) and financed by the PASC project “Towards the HPC-inference of causality networks from multiscale economical data” by Horenko, Gagliardini, and Sawyer. The work of Horenko was partially supported by the Mercator Fellowship of the DFG SPP 1114 "Scaling cascades in complex systems". We would like to gratefully thank Olga Kaiser, Dimitri Igdalov, Anna Marchenko, Benjamin Cumming, Patrick Sanan, Karl Rupp, Susanne Gerber, and the PERMON team (David Horák, Václav Hapla, et al.) for their collaboration in the project.

References

[1] N. Agmon, Y. Alhassid, and R. D. Levine, *An algorithm for finding the distribution of maximal entropy*, J. Comput. Phys. **30** (1979), no. 2, 250–258. Zbl

[2] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, D. A. May, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, H. Zhang, and H. Zhang, *PETSc*, users manual ANL-95/11, rev. 3.7, Argonne National Laboratory, 2016.

[3] J. Barzilai and J. M. Borwein, *Two-point step size gradient methods*, IMA J. Numer. Anal. **8** (1988), no. 1, 141–148. MR Zbl

[4] D. P. Bertsekas, *Nonlinear programming*, 2nd ed., Athena Scientific, 1999. MR Zbl

[5] E. G. Birgin, J. M. Martínez, and M. Raydan, *Nonmonotone spectral projected gradient methods on convex sets*, SIAM J. Optim. **10** (2000), no. 4, 1196–1211. MR Zbl

[6] ———, *Algorithm 813: SPG — software for convex-constrained optimization*, ACM Trans. Math. Softw. **27** (2001), no. 3, 340–349. Zbl

[7] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge University, 2004. MR Zbl

[8] E. J. Candès and M. B. Wakin, *An introduction to compressive sampling*, IEEE Signal Proc. Mag. **25** (2008), no. 2, 21–30.

[9] A.-L. Cauchy, *Méthode générale pour la résolution des systèmes d'équations simultanées*, Comptes Rendus Hebd. Séances Acad. Sci. **25** (1847), 536–538.

[10] Y. Chen and X. Ye, *Projection onto a simplex*, preprint, 2011. arXiv

[11] A. Coates, B. Huval, T. Wang, D. J. Wu, A. Y. Ng, and B. Catanzaro, *Deep learning with COTS HPC systems*, Proceedings of the 30th International Conference on Machine Learning (S. Dasgupta and D. McAllester, eds.), Proceedings of Machine Learning Research, no. 28, 2013, pp. 1337–1345.

[12] Y.-H. Dai and R. Fletcher, *Projected Barzilai–Borwein methods for large-scale box-constrained quadratic programming*, Numer. Math. **100** (2005), no. 1, 21–47. MR

[13] Y.-H. Dai and L.-Z. Liao, *R-linear convergence of the Barzilai and Borwein gradient method*, IMA J. Numer. Anal. **22** (2002), no. 1, 1–10. MR Zbl

[14] Z. Dostál, *Optimal quadratic programming algorithms: with applications to variational inequalities*, Springer Optimization and Its Applications, no. 23, Springer, 2009. MR Zbl

[15] Z. Dostál and L. Pospíšil, *Minimizing quadratic functions with semidefinite Hessian subject to bound constraints*, Comput. Math. Appl. **70** (2015), no. 8, 2014–2028. MR

[16] ———, *Optimal iterative QP and QPQC algorithms*, Ann. Oper. Res. **243** (2016), no. 1–2, 5–18. Zbl

[17] W. Enders, *Applied econometric time series*, 4th ed., Wiley, 2015.

[18] G. Fourestey, B. Cumming, L. Gilly, and T. C. Schulthess, *First experiences with validating and using the Cray power management database tool*, Cray User Group proceedings, 2014.

[19] S. Gerber and I. Horenko, *On inference of causality for discrete state models in a multiscale context*, Proc. Natl. Acad. Sci. USA **111** (2014), no. 41, 14651–14656. MR Zbl

[20] ______, *Improving clustering by imposing network information*, Sci. Adv. **1** (2015), no. 7, 1500163.

[21] L. Grippo, F. Lampariello, and S. Lucidi, *A nonmonotone line search technique for Newton's method*, SIAM J. Numer. Anal. **23** (1986), no. 4, 707–716. MR Zbl

[22] P. C. Hansen and D. P. O'Leary, *The use of the L-curve in the regularization of discrete ill-posed problems*, SIAM J. Sci. Comput. **14** (1993), no. 6, 1487–1503. MR Zbl

[23] V. Hapla, *Massively parallel quadratic programming solvers with applications in mechanics*, Ph.D. thesis, Technical University of Ostrava, 2016.

[24] V. Hapla, D. Horák, L. Pospíšil, M. Čermák, A. Vašatová, and R. Sojka, *Solving contact mechanics problems with PERMON*, High performance computing in science and engineering: second international conference (T. Kozubek, R. Blaheta, J. Šístek, M. Rozložník, and M. Čermák, eds.), Lecture Notes in Computer Science, no. 9611, Springer, 2015, pp. 101–115.

[25] T. J. Hastie and R. J. Tibshirani, *Generalized additive models*, Monographs on Statistics and Applied Probability, no. 43, Chapman and Hall, 1990. MR Zbl

[26] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards **49** (1952), 409–436. MR Zbl

[27] I. Horenko, *On simultaneous data-based dimension reduction and hidden phase identification*, J. Atmos. Sci. **65** (2008), no. 6, 1941–1954.

[28] ______, *On robust estimation of low-frequency variability trends in discrete Markovian sequences of atmospheric circulation patterns*, J. Atmos. Sci. **66** (2009), no. 7, 2059–2072.

[29] ______, *Finite element approach to clustering of multidimensional time series*, SIAM J. Sci. Comput. **32** (2010), no. 1, 62–83. MR Zbl

[30] ______, *On clustering of non-stationary meteorological time series*, Dynam. Atmos. Oceans **49** (2010), no. 2–3, 164–187.

[31] ______, *On the identification of nonstationary factor models and their application to atmospheric data analysis*, J. Atmos. Sci. **67** (2010), no. 5, 1559–1574.

[32] ______, *Nonstationarity in multifactor models of discrete jump processes, memory, and application to cloud modeling*, J. Atmos. Sci. **68** (2011), no. 7, 1493–1506.

[33] ______, *On analysis of nonstationary categorical data time series: dynamical dimension reduction, model selection, and applications to computational sociology*, Multiscale Model. Simul. **9** (2011), no. 4, 1700–1726. MR Zbl

[34] D. E. King, *Dlib-ml: a machine learning toolkit*, J. Mach. Learn. Res. **10** (2009), 1755–1758.

[35] A. J. Laub, *Matrix analysis for scientists and engineers*, Society for Industrial and Applied Mathematics, 2005. MR Zbl

[36] F. Lotte, M. Congedo, A. Lécuyer, F. Lamarche, and B. Arnaldi, *A review of classification algorithms for EEG-based brain–computer interfaces*, J. Neural Eng. **4** (2007), no. 2, R1.

[37] J. H. Macke, M. Opper, and M. Bethge, *Common input explains higher-order correlations and entropy in a simple model of neural population activity*, Phys. Rev. Lett. **106** (2011), no. 20, 208102.

[38] P. Metzner, L. Putzig, and I. Horenko, *Analysis of persistent nonstationary time series and applications*, Commun. Appl. Math. Comput. Sci. **7** (2012), no. 2, 175–229. MR Zbl

[39] M. Mudelsee, *Climate time series analysis: classical statistical and bootstrap methods*, 2nd ed., Atmospheric and Oceanographic Sciences Library, no. 51, Springer, 2014. MR Zbl

[40] T. Munson, J. Sarich, S. Wild, S. Benson, and L. C. McInnes, *Toolkit for Advanced Optimization* (*TAO*), users manual ANL/MCS-TM-322, rev. 3.5, Argonne National Laboratory, 2014.

[41] K. Murphy, *Hidden Markov model* (*HMM*) *toolbox for Matlab*, 2005.

[42] J. Nocedal and S. J. Wright, *Numerical optimization*, Springer, 1999. MR Zbl

[43] L. Pospíšil, *Development of algorithms for solving minimizing problems with convex quadratic function on special convex sets and applications*, Ph.D. thesis, Technical University of Ostrava, 2015.

[44] L. Pospíšil, I. Horenko, P. Gagliardini, and W. Sawyer, *PASC-inference: open-source HPC library of nonparametric Bayesian causality inference methods*, 2017.

[45] M. Raydan M., *Convergence properties of the Barzilai and Borwein gradient method*, Ph.D. thesis, Rice University, 1991. MR

[46] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, preprint, 2015. arXiv

[47] F. Stimberg, M. Opper, G. Sanguinetti, and A. Ruttor, *Inference in continuous-time change-point models*, Neural Information Processing Systems Conference 2011 (J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, eds.), Advances in Neural Information Processing Systems, no. 24, 2011, pp. 2717–2725.

[48] L. A. Vese and C. Le Guyader, *Variational methods in image processing*, CRC, 2016. MR Zbl

[49] G. Wahba, *Spline models for observational data*, CBMS-NSF Regional Conference Series in Applied Mathematics, no. 59, Society for Industrial and Applied Mathematics, 1990. MR Zbl

[50] Y. Wiaux, L. Jacques, G. Puy, A. M. M. Scaife, and P. Vandergheynst, *Compressed sensing imaging techniques for radio interferometry*, Mon. Not. R. Astron. Soc. **395** (2009), no. 3, 1733–1742.

[51] S. J. Wright, *Primal-dual interior-point methods*, Society for Industrial and Applied Mathematics, 1997. MR Zbl

Received June 20, 2017.

LUKÁŠ POSPÍŠIL: lukas.pospisil@usi.ch
Università della Svizzera italiana, Lugano, Switzerland

PATRICK GAGLIARDINI: patrick.gagliardini@usi.ch
Università della Svizzera italiana, Lugano, Switzerland

WILLIAM SAWYER: wsawyer@cscs.ch
Swiss National Supercomputing Centre, Lugano, Switzerland

ILLIA HORENKO: horenkoi@usi.ch
Università della Svizzera italiana, Lugano, Switzerland

Guidelines for Authors

Authors may submit manuscripts in PDF format on-line at the Submission page at msp.org/camcos.

Originality. Submission of a manuscript acknowledges that the manuscript is original and and is not, in whole or in part, published or under consideration for publication elsewhere. It is understood also that the manuscript will not be submitted elsewhere while under consideration for publication in this journal.

Language. Articles in CAMCoS are usually in English, but articles written in other languages are welcome.

Required items. A brief abstract of about 150 words or less must be included. It should be self-contained and not make any reference to the bibliography. If the article is not in English, two versions of the abstract must be included, one in the language of the article and one in English. Also required are keywords and subject classifications for the article, and, for each author, postal address, affiliation (if appropriate), and email address.

Format. Authors are encouraged to use LaTeX but submissions in other varieties of TeX, and exceptionally in other formats, are acceptable. Initial uploads should be in PDF format; after the refereeing process we will ask you to submit all source material.

References. Bibliographical references should be complete, including article titles and page ranges. All references in the bibliography should be cited in the text. The use of BibTeX is preferred but not required. Tags will be converted to the house format, however, for submission you may use the format of your choice. Links will be provided to all literature with known web locations and authors are encouraged to provide their own links in addition to those supplied in the editorial process.

Figures. Figures must be of publication quality. After acceptance, you will need to submit the original source files in vector graphics format for all diagrams in your manuscript: vector EPS or vector PDF files are the most useful.

Most drawing and graphing packages (Mathematica, Adobe Illustrator, Corel Draw, MATLAB, etc.) allow the user to save files in one of these formats. Make sure that what you are saving is vector graphics and not a bitmap. If you need help, please write to graphics@msp.org with details about how your graphics were generated.

White space. Forced line breaks or page breaks should not be inserted in the document. There is no point in your trying to optimize line and page breaks in the original manuscript. The manuscript will be reformatted to use the journal's preferred fonts and layout.

Proofs. Page proofs will be made available to authors (or to the designated corresponding author) at a Web site in PDF format. Failure to acknowledge the receipt of proofs or to return corrections within the requested deadline may cause publication to be postponed.

Communications in Applied Mathematics and Computational Science

vol. 13 no. 1 2018

1559-3940(2018)13:1;1-C