Electronic Transactions on Numerical Analysis.
Volume 66, pp. 32–63, 2026.

ISSN 1068–9613.
DOI: 10.1553/etna_vol66s32

ETNA
Kent State University and
Johann Radon Institute (RICAM)

PARALLEL-IN-ITERATION OPTIMIZATION USING MULTIGRID REDUCTION IN TIME*

G. H. M. ARAÚJO†, O. A. KRZYSIK‡, AND H. DE STERCK†

Abstract. Standard gradient-based iteration algorithms for optimization, such as gradient descent and its various proximal-based extensions to nonsmooth problems, are known to converge slowly for ill-conditioned problems, sometimes requiring many tens of thousands of iterations in practice. Since these iterations are computed sequentially, they may present a computational bottleneck in large-scale parallel simulations. In this work, we present a "parallel-in-iteration" framework that allows one to parallelize across these iterations using multiple processors with the objective of reducing the wall-clock time needed to solve the underlying optimization problem. Our methodology is based on repurposing parallel-time integration algorithms for time-dependent differential equations, motivated by the fact that optimization algorithms often have interpretations as discretizations of time-dependent differential equations (such as gradient flow). Specifically in this work, we use the parallel-in-time method of multigrid reduction in time (MGRIT), but note that our approach permits in principle the use of any other parallel-in-time method. We numerically demonstrate the efficacy of our approach on two different model problems, including a standard convex quadratic problem and the nonsmooth elastic obstacle problem in one and two spatial dimensions. For our model problems, we observe fast MGRIT convergence analogous to its prototypical performance on partial differential equations of diffusion type. Some theory is presented to connect the convergence of MGRIT to the convergence of the underlying optimization algorithm. Theoretically predicted parallel speedup results are also provided.

Key words. parallel in time, MGRIT, convex optimization, gradient descent, proximal gradient, elastic obstacle

AMS subject classifications. 65F10, 65K10, 65M55

1. Introduction.

Consider the unconstrained optimization problem,

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad f(\mathbf{u}), \tag{1.1}$$

with $f : \mathbb{R}^N \to \mathbb{R}$ a convex and L-smooth function, i.e., a convex function with L-Lipschitz gradient:

$$\|\nabla f(\mathbf{u}) - \nabla f(\mathbf{v})\| \leqslant L\|\mathbf{u} - \mathbf{v}\| \quad \forall\, \mathbf{u}, \mathbf{v} \in \mathbb{R}^N.$$

Such optimization problems are ubiquitous throughout the sciences, with many theoretical properties established for them and applications in fields such as machine learning [44, 45], signal reconstruction [5, 8, 23], game theory [22], econometrics [2], and mathematical physics [39, 46]. Various iterative gradient-based optimization algorithms for solving unconstrained minimization problems, such as gradient descent and Nesterov's accelerated gradient descent [33], take the form of one- or two-step updates. Generally, the convergence rates of these algorithms for convex functions can be slow. For example, gradient descent has a convergence rate of $\mathcal{O}(1/k)$ for problem (1.1), where k denotes the number of iterations; while Nesterov's improved rate of $\mathcal{O}(1/k^2)$ was shown in [34] to be optimal among all methods that

*Received July 31, 2025. Accepted January 22, 2026. Published online on June 1, 2026. Recommended by Eran Treister. The work of the second author (O.A.K.) has been funded by the Los Alamos National Laboratory Advanced Simulation and Computation program through Contract No. 89233218CNA000001. The research was performed under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy at Los Alamos National Laboratory, managed by Triad National Security, LLC, under contract 89233218CNA000001. LA-UR-25-27245.

†Department of Applied Mathematics, University of Waterloo, Waterloo, Ontario, Canada ({ghmacieiradearaujo, hans.desterck}@uwaterloo.ca).

‡Theoretical Division, Los Alamos Laboratory, Los Alamos, NM, USA (okrzysik@lanl.gov, ORCID: 0000-0001-7880-6512).

ETNA
Kent State University and
Johann Radon Institute (RICAM)

have information only about the gradient of f for its updates. In practical terms, this means that many iterations are typically required to solve the problem; for example, tens of thousands of iterations are not uncommon [8, 35].

Related to (1.1) is the unconstrained problem of minimizing a sum of two functions

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad F(\mathbf{u}), \quad \text{with } F(\mathbf{u}) := f(\mathbf{u}) + g(\mathbf{u}), \tag{1.2}$$

where $f : \mathbb{R}^N \to \mathbb{R}$ is convex and L-smooth, and $g : \mathbb{R}^N \to (-\infty, +\infty]$ is continuous, convex, and possibly nondifferentiable. The minimization problem (1.2) is typically solved by splitting algorithms [28], in which the differentiable f and possibly nondifferentiable g are handled differently. Many classical splitting algorithms also take the form of one- or two-step updates. Namely, the proximal gradient algorithm and FISTA (fast iterative shrinkage-thresholding algorithm) [5] are generalizations of gradient descent and Nesterov's accelerated gradient, respectively, for solving (1.2), maintaining the convergence rates of $\mathcal{O}(1/k)$ and $\mathcal{O}(1/k^2)$ as their counterparts do for (1.1). The convergence rates of these optimization algorithms are also slow, in practice requiring many iterations to get close to the true optimum [5, 8]. If gradient evaluations take some nontrivial amount of compute time, we may have to wait for a long wall-clock time to get an accurate approximation to the optimum because the algorithms are sequential in k.

One can write one-step algorithms in the form $\mathbf{u}_{k+1} = \Phi(\mathbf{u}_k) + \mathbf{w}_{k+1}$, where $\Phi(\cdot)$ is a linear or nonlinear operator and $\mathbf{w}_{k+1}$ is a generic term that does not dpeend on $\mathbf{u}_k$, for example, to encode the initial condition or a forcing term. For example, we can write the gradient descent update for (1.1) given by

$$\mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_k), \quad k = 0, 1, \ldots \tag{1.3}$$

as

$$\mathbf{u}_{k+1} = (I - s\nabla f)(\mathbf{u}_k).$$

Suppose we need K iterations to approximate the true optimum within the desired accuracy. Then, taking $\Phi := (I - s\nabla f)$, computing the K gradient descent iterations in parallel is equivalent to solving the below "all-at-once" system in parallel:

$$\mathcal{A}(\mathbf{u}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_K \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_K \end{bmatrix} = \mathbf{w}, \tag{1.4}$$

where for this particular case $\mathbf{w}_0$ is the initial condition and $\mathbf{w}_1 = \cdots = \mathbf{w}_K = \mathbf{0}$. We note that some two-step methods, such as Nesterov's accelerated gradient descent [33] for solving (1.1), can be rewritten as one-step algorithms of the form

$$[\mathbf{u}_{k+1} \quad \mathbf{v}_{k+1}]^T = \Phi([\mathbf{u}_k \quad \mathbf{v}_k]^T)$$

for block vector inputs $[\mathbf{u}_k \quad \mathbf{v}_k]^T$, such that an analogous all-at-once system to (1.4) holds in such cases too.

The system (1.4) is exactly of the kind which parallel-in-time algorithms solve for time-dependent evolution equations [19, 36]. For example, the parareal method [27] is a two-level parallel-in-time method [21] which solves systems of differential equations by employing

ETNA
Kent State University and
Johann Radon Institute (RICAM)

sequential coarse-grid solves to speed up solving the original fine-grid problem in parallel. However, parareal is limited in concurrency, as the coarse-grid solve is still sequential. Thus, the multigrid reduction-in-time algorithm (MGRIT) [16] was developed as a multilevel generalization of parareal based on multigrid reduction (MGR) techniques [38].

In the linear case where $\Phi(\mathbf{u}_k)$ is computed by multiplying $\mathbf{u}_k$ by the matrix Φ, MGRIT solves the system (1.4) via a coarse-grid system derived from an approximation to the Schur complement of (1.4) in terms of the coarse-level variables, and has a factor of $m \in \mathbb{N}$ fewer variables on each successively coarser grid. Here, m is called the coarsening factor. The biggest challenge in applying MGRIT is developing a suitable coarse-grid operator $\Phi_\Delta \approx \Phi^m$, where Φ^m is the operator appearing in the Schur complement. That is, Φ_Δ should approximate m successive applications with Φ, but it should be computationally much less expensive than applying Φ m times; this cost reduction combined with a quickly converging method is what allows for speedup on a parallel machine. It has been well documented that parallel-in-time methods such as parareal and MGRIT often work very well for diffusion-dominated (parabolic-type) partial differential equation (PDE) problems [14, 16, 25]. Advection-dominated (hyperbolic-type) problems are more challenging, but careful construction of coarse-grid operators may facilitate fast convergence [12, 13, 25]. This provides motivation for the use of parallel-in-time methods for optimization problems such as (1.1) and (1.2).

For sufficiently small step lengths s as in (1.3), we can sometimes interpret optimization algorithms as discretizations of certain ordinary differential equations (ODEs), in which k acts as a "discrete time index," and the step length s is related to the time-scale of the ODE. For example, as $s \to 0$, gradient descent can be thought of as an explicit Euler discretization of the gradient flow ODE,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u} = -\nabla f(\mathbf{u}(t)), \quad t > 0,$$

and, as $s \to 0$, Nesterov's accelerated gradient descent can be thought of as an explicit discretization [43] of the ODE,

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{u} + \frac{3}{t}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u} + \nabla f(\mathbf{u}(t)) = 0, \quad t > 0.$$

With these connections between solving optimization problems in parallel and solving systems of differential equations using parallel-in-time methods, we present a framework for implementing MGRIT for solving optimization problems in the classes (1.1) and (1.2) in parallel. We describe two model problems belonging to the former and latter classes, and discuss the choices of fine- and coarse-grid operators for solving these problems with MGRIT. These operators are based on standard choices of algorithms for solving these problems sequentially, such that the MGRIT implementation can be interpreted as a sped-up version of these sequential algorithms. Parallel-in-time methods have been previously used in the context of solving PDE-constrained optimization problems either as preconditioners or for solving time-dependent subproblems that arise in the context of time-dependent PDE-constrained optimization, such as in optimal control [20, 29, 48, 49] or for computing the geometry of induction motors [24]. These previous applications differ from our approach, which is to use parallel-in-time methods to parallelize across iterations of some underlying sequential optimization method.

The rest of this paper is organized as follows. Sections 2 and 3 introduce the model problems and optimization algorithms that we use to solve them, respectively, with the latter elaborating on the interpretation of optimization methods as discretizations of time-dependent

ETNA
Kent State University and
Johann Radon Institute (RICAM)

ODEs. Section 4 reviews the MGRIT parallel-in-time method. Section 5 discusses our methodology for solving optimization problems in parallel with MGRIT, which explores the aforementioned connections between ODEs for which parallel-in-time methods have originally been designed and optimization problems, providing the key insight behind this paper. Section 6 provides numerical results, while Section 7 provides theoretical speedup estimates. Section 8 explores the direct application of the proposed MGRIT approach to momentum-accelerated algorithms and discusses the difficulties that were encountered to obtain fast convergence of MGRIT iterations for this case. Section 9 features concluding remarks and ideas for future work.

2. Model problems. We now introduce the two model problems considered throughout this paper. The first model problem (MP1) is relatively simple, and serves largely as a stepping stone towards the more challenging second model problem (MP2), which presents a more realistic application of our proposed algorithm. The optimization problems we consider have a connection to discretizations of time-dependent heat or diffusion PDEs, and thus help to motivate our use of parallel-in-time methods.

In what follows, we consider continuous optimization problems on d-dimensional domains $\Omega \subset \mathbb{R}^d$, $d \in \{1, 2\}$, with boundary denoted by $\partial\Omega$. We let $\mathbf{x} \in \mathbb{R}^d$ denote a coordinate on Ω, such that $\mathbf{x} = x$ when $d = 1$ and $\mathbf{x} = (x, y)$ when $d = 2$. The domain Ω is discretized with a mesh consisting of $n + 2$ total points, and n interior points, in each direction, equally separated by a distance of $h > 0$, with $h = \mathcal{O}(1/n)$. Of particular relevance to our model problems are finite-difference discretizations of (negative) Laplacian operators, which, on the interior mesh points, are defined as

$$A_1 = -\frac{1}{h^2}\begin{bmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \ddots & \ddots & \ddots \\ & & 1 & -2 \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{2.1a}$$

and

$$A_2 = -\frac{1}{h^2}\begin{bmatrix} -T & I & & \\ I & -T & I & \\ & \ddots & \ddots & \ddots \\ & & I & -T \end{bmatrix} \in \mathbb{R}^{n^2\times n^2} \tag{2.1b}$$

for $d = 1$ and $d = 2$, respectively. Here, $T \in \mathbb{R}^{n\times n}$ is a tridiagonal matrix with elements -1, 4, and -1 on the sub-, main, and super-diagonals, respectively, and $I \in \mathbb{R}^{n\times n}$ is the identity matrix. The 2-norm of A_1 is

$$\|A_1\|_2 = \frac{1}{h^2}\, 4\sin^2\left(\frac{n\pi}{2(n+1)}\right) \approx \frac{4}{h^2} \quad \text{for } h \ll 1,$$

while the 2-norm of A_2 is

$$\|A_2\|_2 = \frac{1}{h^2}\, 8\sin^2\left(\frac{n\pi}{2(n+1)}\right) \approx \frac{8}{h^2} \quad \text{for } h \ll 1.$$

Throughout the paper, we refer to A_1 and A_2 as one-dimensional (1D) and two-dimensional (2D) Laplacian matrices, respectively.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

2.1. Quadratic minimization problem (MP1). For our first model problem, we consider the straightforward problem of quadratic minimization, formulated as

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad \frac{1}{2}\langle A_1\mathbf{u},\mathbf{u}\rangle - \langle \mathbf{b},\mathbf{u}\rangle, \tag{2.2}$$

where A_1 is the 1D Laplacian matrix (2.1a), such that $N = n$, and $\mathbf{b} \in \mathbb{R}^N$ is some prescribed vector. Since A_1 is symmetric positive definite (SPD), the objective function here, $f(\mathbf{u}) = \langle A_1\mathbf{u},\mathbf{u}\rangle - \langle \mathbf{b},\mathbf{u}\rangle$, is strongly convex and L-smooth, where L denotes the smallest Lipschitz constant of ∇f, which in this case is $L = \|A_1\|$. Since A_1 is SPD, and hence invertible, the unique solution of (2.2) is given by $\mathbf{u} = A_1^{-1}\mathbf{b}$, which is attained when $\nabla f(\mathbf{u}) = \mathbf{0}$.

2.2. Elastic obstacle problem (MP2). The elastic obstacle problem (EOP) [7] consists of finding the equilibrium state of an elastic membrane under a nonpenetration constraint. The EOP we consider was initially motivated by mathematical physics [39] such as steady-state fluid interaction, thin-plate solid dynamics, and elastostatics [46]. Various strategies have been proposed for numerically solving the EOP, including multigrid approaches such as those in [1, 6, 30, 50]. The formulation we present here follows an L^1 penalty approach [46].

Given an obstacle function $\phi(\mathbf{x}) : \Omega \to \mathbb{R}$, the EOP seeks the location of a membrane $\widehat{u}(\mathbf{x}) : \Omega \to \mathbb{R}$ with the lowest elastic potential energy. Assuming the elastic potential energy of the membrane is proportional to its surface area, the EOP can be written as

$$\min_{\widehat{u}\in\Omega} \quad \int_\Omega \sqrt{1 + \|\nabla\widehat{u}\|_2^2}\,\mathrm{d}\mathbf{x} \quad \text{s.t. } \widehat{u} \geq \phi \text{ on } \Omega,\ \widehat{u} = 0 \text{ on } \partial\Omega. \tag{2.3}$$

Here, $\nabla\widehat{u} : \Omega \to \mathbb{R}^d$ is the gradient field of $\widehat{u}$, and the norm $\|\cdot\|_2$ induced by the inner product $\langle\cdot,\cdot\rangle$ is the standard 2-norm for functions over Ω. Note that we assume $\phi \leq 0$ on $\partial\Omega$, so the boundary condition $\widehat{u} = 0$ on $\partial\Omega$ is well posed.

An approximation of (2.3) can be obtained by linearization [50], based on the Maclaurin series $\sqrt{1+x^2} = 1 + \frac{1}{2}x^2 + o(x^4)$. Assuming that $\|\nabla\widehat{u}\|_2^2$ is sufficiently small such that we can ignore higher-order terms, we have $\sqrt{1 + \|\nabla\widehat{u}\|_2^2} \approx 1 + \frac{1}{2}\|\nabla\widehat{u}\|_2^2$; further dropping the constant 1 gives the linearized obstacle problem

$$\min_{\widehat{u}\in\Omega} \quad \int_\Omega \frac{1}{2}\|\nabla\widehat{u}\|_2^2\,\mathrm{d}\mathbf{x} \quad \text{s.t. } \widehat{u} \geq \phi,\ \widehat{u} = 0 \text{ on } \partial\Omega. \tag{2.4}$$

In order to solve the linearized problem (2.4) we discretize it with finite differences on the aforementioned equispaced meshes.

Consider first the 1D case. Let $x_i,\ i \in \{1,\dots,n\}$, denote an interior mesh point, and let $\widehat{u}_i \approx \widehat{u}(x_i)$ be an approximation of $\widehat{u}$ at this mesh point. Then the integral is discretized as

$$\int_\Omega \frac{1}{2}\|\nabla\widehat{u}(x)\|_2^2\,\mathrm{d}x \approx \sum_{i=1}^{n} \frac{1}{2}\|\nabla\widehat{u}(x_i)\|_2^2 h \approx \frac{h}{2}\langle A_1\widehat{\mathbf{u}},\widehat{\mathbf{u}}\rangle,$$

where $\widehat{\mathbf{u}} = (\widehat{u}_1,\dots,\widehat{u}_n) \in \mathbb{R}^n$, and $A_1 \in \mathbb{R}^{n\times n}$ is the 1D Laplacian matrix (2.1a), which assumes zero boundary conditions.

The approximation proceeds analogously in 2D, where now we let $(x_i, y_j),\ i,j \in \{1,\dots,n\}$, denote an interior mesh point, and $\widehat{u}_{i,j} \approx \widehat{u}(x_i,y_j)$. In this case, the integral is approximated as

$$\int_\Omega \frac{1}{2}\|\nabla\widehat{u}(x,y)\|_2^2\,\mathrm{d}x\,\mathrm{d}y \approx \sum_{i,j=1}^{n} \frac{1}{2}\|\nabla\widehat{u}(x_i,y_j)\|_2^2 h^2 \approx \frac{h^2}{2}\langle A_2\widehat{\mathbf{u}},\widehat{\mathbf{u}}\rangle,$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

where $\widehat{\mathbf{u}} = (\widehat{u}_{1,1}, \ldots, \widehat{u}_{n,n}) \in \mathbb{R}^{n^2}$ and $A_2 \in \mathbb{R}^{n^2 \times n^2}$ is the 2D Laplacian matrix (2.1b).

Ignoring the leading h or h^2 factors, since this does not change the solution of the minimization problem, our discretized versions of (2.4) take the form

$$\min_{\widehat{\mathbf{u}} \in \mathbb{R}^N} \quad \frac{1}{2}\langle A_d \widehat{\mathbf{u}}, \widehat{\mathbf{u}} \rangle \quad \text{s.t.} \quad \widehat{\mathbf{u}} \geq \phi, \tag{2.5}$$

where $N = n$ for $d = 1$, and $N = n^2$ for $d = 2$. Moreover, $\phi \in \mathbb{R}^N$ denotes the discretized version of $\phi(\mathbf{x})$ on the underlying mesh, and A_d the Laplacian matrix on this mesh. By making a change of variable $\mathbf{u} := \widehat{\mathbf{u}} - \phi \geq \mathbf{0}$, we can write (2.5) in its shifted form:

$$\min_{\mathbf{u} \in \mathbb{R}^N} \quad \frac{1}{2}\langle A_d \mathbf{u}, \mathbf{u} \rangle + \langle A_d \phi, \mathbf{u} \rangle + \frac{1}{2}\langle A_d \phi, \phi \rangle \quad \text{s.t.} \quad \mathbf{u} \geq \mathbf{0}. \tag{2.6}$$

Dropping the $\frac{1}{2}\langle A_d \phi, \phi \rangle$ term (since it does not influence the minimizer), choosing $\mathbf{p} = -A_d \phi$, and using an indicator function $i_+ : \mathbb{R}^N \to \mathbb{R}$, where $i_+(\mathbf{a}) = 0$ if $(\mathbf{a})_q \geq 0$ for all q and $i_+(\mathbf{a}) = +\infty$ if $(\mathbf{a})_q < 0$ for any q, allows (2.6) to be written in unconstrained form as

$$\min_{\mathbf{u} \in \mathbb{R}^N} \quad \frac{1}{2}\langle A_d \mathbf{u}, \mathbf{u} \rangle - \langle \mathbf{p}, \mathbf{u} \rangle + i_+(\mathbf{u}). \tag{2.7}$$

However, we choose to work with an alternative unconstrained formulation, given by

$$\min_{\mathbf{u} \in \mathbb{R}^N} \quad \frac{1}{2}\langle A_d \mathbf{u}, \mathbf{u} \rangle - \langle \mathbf{p}, \mathbf{u} \rangle + \lambda \|(-\mathbf{u})_+\|_1, \tag{2.8}$$

where $\lambda > 0$ is a predefined penalty parameter and $\sum_i (-(\mathbf{u})_i)_+ = \|(-\mathbf{u})_+\|_1$, where $(a)_+ = \max\{0, a\}$ and $\|\cdot\|_1$ denotes the L^1-norm. For sufficiently large λ, the solution of (2.8) is the same as that of (2.7) [18, 31], hence approximating the solution of (2.4) after reversing the change of variables $\widehat{\mathbf{u}} = \mathbf{u} + \phi$.

Throughout the rest of this paper, we refer to the shifted approximate obstacle problem in penalty form (2.8) as MP2-1D and MP2-2D, in the 1D and 2D cases, respectively. In our MP2-1D numerical experiments, we consider $x \in \Omega := [0, 3\pi]$ and the obstacle $\phi(x) = \max\{0, \sin x\}$, for which (2.4) has the exact solution given by $\widehat{u}(x) = \sin x$ for $x \in [0, \frac{1}{2}\pi]$ or $x \in [\frac{5}{2}\pi, 3\pi]$ and $\widehat{u}(x) = 1$ for $x \in [\frac{1}{2}\pi, \frac{5}{2}\pi]$; see Figure 2.1. For MP2-2D, we consider $(x, y) \in \Omega := [0, 3\pi]^2$ and $\phi(x, y) = \max\{0, \sin x\} \max\{0, \sin y\}$; again, see Figure 2.1. We take $\lambda = 900$ in both the 1D and 2D cases, which is large enough such that solving (2.8) is equivalent to solving (2.7).

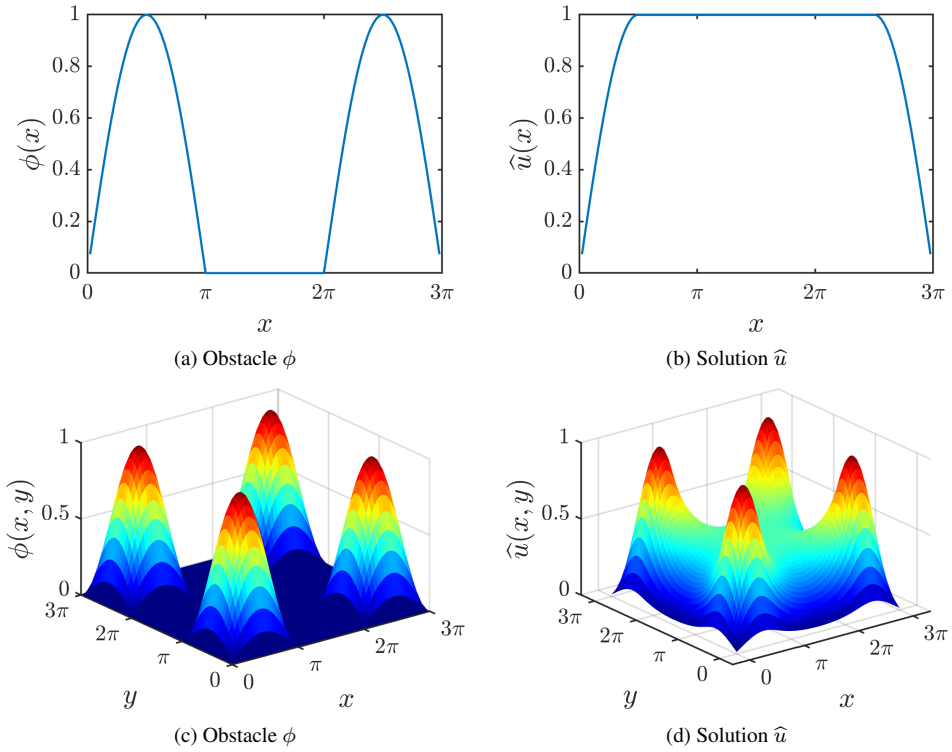
Figure 2.1 presents visualizations of the obstacles and corresponding solutions of MP2-1D and MP2-2D. Note that the solutions pictured in panels (b) and (d) are those on the interior of the domain, and hence with the boundary points $\widehat{u} = 0$ eliminated.

3. Optimization algorithms. The algorithms we employ are based on gradients and proximals of functions; in the case of nondifferentiable convex functions, gradients are replaced by subgradients. Let us recall the definition of the proximal operator P_f of a function $f : \mathbb{R}^N \to \mathbb{R}$:

$$P_f(\mathbf{v}) = \left\{ \mathbf{u} \in \mathbb{R}^N : \mathbf{u} = \operatorname*{arg\,min}_{\mathbf{z} \in \mathbb{R}^N} \left(f(\mathbf{z}) + \frac{1}{2}\|\mathbf{z} - \mathbf{v}\|^2 \right) \right\}. \tag{3.1}$$

Under the assumption that f is convex and continuous, which holds throughout this paper, $(f(\cdot) + \frac{1}{2}\|\cdot - \mathbf{v}\|^2)(\mathbf{z})$ is strongly convex, such that $P_f(\mathbf{v})$ is single-valued.

ETNA
Kent State University and
Johann Radon Institute (RICAM)



(a) Obstacle ϕ (b) Solution $\widehat{u}$

(c) Obstacle ϕ (d) Solution $\widehat{u}$

FIG. 2.1. *(a) Obstacle ϕ and (b) solution $\widehat{u}$ of MP2-1D (2.8), and (c) obstacle ϕ and (d) solution $\widehat{u}$ of MP2-2D (2.8), computed numerically using the proximal gradient algorithm (3.7).*

Recall the definition of ∂f, the subdifferential of f:

$$\partial f(\mathbf{u}) = \{\mathbf{z} \in \mathbb{R}^N : \forall\, \mathbf{v} \in \mathbb{R}^N,\ f(\mathbf{u}) + \langle \mathbf{v} - \mathbf{u}, \mathbf{z} \rangle \leq f(\mathbf{v})\},$$

where we call $\mathbf{z} \in \partial f(\mathbf{u})$ a subgradient of f at $\mathbf{u}$; if f is differentiable at $\mathbf{u}$, then $\partial f(\mathbf{u}) = \{\nabla f(\mathbf{u})\}$. We note that $\mathbf{u} = P_f(\mathbf{v})$ if and only if $\mathbf{v} - \mathbf{u} \in \partial f(\mathbf{u})$ or, equivalently, $\mathbf{v} \in \mathbf{u} + \partial f(\mathbf{u}) = (I + \partial f)(\mathbf{u})$. Then, it follows that $\mathbf{u} = (I + \partial f)^{-1}(\mathbf{v})$, such that we can characterize the proximal operator as the resolvent of the subdifferential operator:

$$P_f = (I + \partial f)^{-1}. \tag{3.2}$$

We now discuss the optimization algorithms that we apply to the discrete model problems from Section 2. The main algorithms we consider are gradient descent for MP1, (2.2), and proximal gradient descent for MP2, (2.8), whose convergence we attempt to speed up by means of parallelization in time. In our method, we also make use of two auxiliary algorithms on the coarse grid, namely proximal point for MP1 and alternating proximal mappings for MP2, which are closely related to gradient descent and proximal gradient descent, respectively. We refer the reader to [4, Chapters 26 and 29–31] for further details on the convergence properties of these algorithms.

3.1. Optimization algorithms for MP1. Consider the problem of minimizing a convex and L-smooth function f,

$$\min_{\mathbf{u} \in \mathbb{R}^N} \quad f(\mathbf{u}). \tag{3.3}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

The first of the two methods we consider for solving (3.3) is the classical gradient descent method. Given a starting point $\mathbf{u}_0 \in \mathbb{R}^N$ and a fixed positive scalar $s > 0$, the gradient descent iteration is given by

$$\mathbf{u}_{k+1} = (I - s\nabla f)(\mathbf{u}_k) =: G_{sf}(\mathbf{u}_k), \quad k = 0, 1, \ldots. \tag{3.4}$$

The gradient descent sequence converges to a minimizer of function f for $0 < s < 2/L$ with a convergence rate of $\mathcal{O}(1/k)$. This means that if we want $|f(\mathbf{u}_k) - f(\mathbf{u}^*)| \leq \epsilon$, where $\mathbf{u}^*$ is a minimizer of f and $\epsilon > 0$, we need $\mathcal{O}(1/\epsilon)$ iterations of gradient descent. Under strong convexity, linear convergence is attained [34] and $\mathcal{O}(\log(1/\epsilon))$ are necessary asymptotically.

We note that MP1 (2.2) is a case of (3.3) for the strongly convex function $f(\mathbf{u}) = \langle A_1\mathbf{u}, \mathbf{u}\rangle - \langle \mathbf{b}, \mathbf{u}\rangle$, with A_1 the Laplacian (2.1a). Moreover, $\partial f(\mathbf{u}) = \{\nabla f(\mathbf{u})\}$, with $\nabla f(\mathbf{u}) = A_1\mathbf{u} - \mathbf{b}$, is L-Lipschitz with $L = \|A_1\|_2$, such that f is L-smooth. Therefore, the objective function f defined by (2.2) satisfies the conditions required for convergence of both (3.4) and (3.5).

When it is necessary to take steps of size larger than the $2/L$ limit for gradient descent, we substitute gradient descent steps by proximal point steps, our second method, which is stable for all $s > 0$. Given some starting point $\mathbf{u}_0$ and a positive scalar $s > 0$, the proximal point algorithm update for a continuous and convex f is given by

$$\mathbf{u}_{k+1} = (I + s\partial f)^{-1}(\mathbf{u}_k) := P_{sf}(\mathbf{u}_k) \quad k = 0, 1, \ldots, \tag{3.5}$$

where we note that P_{sf} is the proximal of sf (3.1); since $s > 0$, we have $s\partial f(\cdot) = \partial(sf)(\cdot)$. The proximal point iteration (3.5) converges to a minimizer of f; note that L-smoothness of f is not required for this convergence, unlike in the case of gradient descent. We note that for simple problems, such as MP1, the proximal point method can converge in one step. However, our interest here is not in the proximal point method itself, but in using proximal steps to approximate a fixed number of gradient descent steps, as discussed in Section 5.2.

3.2. Optimization algorithms for MP2. Consider the possibly nonsmooth problem of minimizing the sum of functions

$$\min_{\mathbf{u}\in\mathbb{R}^N} \quad F(\mathbf{u}), \quad \text{with } F(\mathbf{u}) = f(\mathbf{u}) + g(\mathbf{u}), \tag{3.6}$$

where f is convex and L-smooth, and g is continuous, convex, and possibly nondifferentiable. We also assume that the set of minimizers of F is nonempty. Minimizers of F can be computed by splitting algorithms such as the proximal gradient method, which takes the form

$$\mathbf{u}_{k+1} := P_{sg} \circ G_{sf}(\mathbf{u}_k) = (I + s\partial g)^{-1}((I - s\nabla f)(\mathbf{u}_k)), \quad k = 0, 1, \ldots, \tag{3.7}$$

where $0 < s < 2/L$, where P_{sg} is the proximal of sg (3.1) and G_{sf} is the gradient descent step for f of size s (3.4). The proximal gradient method is a composition of a gradient descent step over f and a proximal point step over g; we note that proximal gradient inherits the stability constraint of gradient descent, namely that $s < 2/L$. We also note that for $f = 0$ in (3.6), iteration (3.7) reduces to the proximal point algorithm (3.5), while for $g = 0$ in (3.6) the iteration reduces to gradient descent (3.4).

For MP2 (2.8) written in the form of (3.6), we have $f = \frac{1}{2}\langle A_d\mathbf{u}, \mathbf{u}\rangle - \langle \mathbf{p}, \mathbf{u}\rangle$ and $g = \lambda\|(-\mathbf{u})_+\|_1$. Here, f is strongly convex and L-smooth, with $L = \|A_d\|_2$, where A_d is given by (2.1a) for $d = 1$ and by (2.1b) for $d = 2$, and g is convex and continuous. Therefore, f and g satisfy the requirements for convergence of (3.7), which converges to a solution of (3.6) as desired.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

When it is necessary to take steps of size $s > 2/L$, we substitute proximal gradient descent by the alternating proximal mappings method, with iteration given by

$$\mathbf{u}_{k+1} := P_{sg} \circ P_{sf}(\mathbf{u}_k) = (I + s\partial g)^{-1}((I + s\nabla f)^{-1}(\mathbf{u}_k)), \tag{3.8}$$

where P_{sg} and P_{sf} are proximals (3.1) of sg and sf, respectively, and we note that $\partial f = \{\nabla f\}$, since f is differentiable by assumption. This method (3.8) is a composition of two proximal steps over f and g. In particular, the sequence (3.8) converges to a minimizer of $\mathrm{env}(f) + g$ [4, Proposition 31.1], where

$$\mathrm{env}(f(\mathbf{u})) := \min_{\mathbf{u}\in\mathbb{R}^N} \left(f(P_f(\mathbf{u})) + \frac{1}{2}\|\mathbf{u} - P_f(\mathbf{u})\|^2 \right)$$

is the Moreau envelope of f at $\mathbf{u}$. This means that (3.8) does not generally converge to a minimizer of (3.6).

3.3. Optimization methods as ODE discretizations. Here, we describe the connection between optimization algorithms and ODE discretizations, which provides key motivation for the parallel-in-iteration framework we outline in the next section. The connection between these has been well established in the literature; see [3, 37, 40, 43].

First, consider the gradient flow ODE of a convex and L-smooth function f, given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = -\nabla f(\mathbf{u}(t)), \quad \mathbf{u}(0) = \mathbf{u}_0. \tag{3.9}$$

Applying a forward finite-difference approximation of this equation yields the forward Euler discretization

$$\mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_k), \quad k = 0, 1, \ldots, \tag{3.10}$$

where $\mathbf{u}_{k+1} \approx \mathbf{u}(t_k + \Delta t)$, $\mathbf{u}_k \approx \mathbf{u}(t_k)$, and $s = \Delta t$. Observe that this forward Euler discretization is identical to the gradient descent iteration (3.4). Similarly, applying a backward finite-difference approximation to (3.9) yields the backward Euler discretization

$$\mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_{k+1}), \quad k = 0, 1, \ldots. \tag{3.11}$$

Let us recall the proximal point method (3.5) and note that

$$\begin{aligned}
\mathbf{u}_{k+1} = P_{sf}(\mathbf{u}) &\iff \mathbf{u}_{k+1} = \operatorname*{argmin}_{\mathbf{u}} \left(sf(\mathbf{u}) + \frac{1}{2}\|\mathbf{u} - \mathbf{u}_k\|^2 \right) \\
&\iff \mathbf{u}_{k+1} = \operatorname*{argmin}_{\mathbf{u}} \left(f(\mathbf{u}) + \frac{1}{2s}\|\mathbf{u} - \mathbf{u}_k\|^2 \right) \\
&\iff \mathbf{0} = \nabla f(\mathbf{u}_{k+1}) + \frac{1}{s}(\mathbf{u}_{k+1} - \mathbf{u}_k) \\
&\iff \mathbf{u}_{k+1} = \mathbf{u}_k - s\nabla f(\mathbf{u}_{k+1}).
\end{aligned}$$

Therefore, the proximal point algorithm (3.5) is equivalent to a backward Euler discretization of (3.9). Furthermore, the proximal point algorithm can be interpreted as an implicit variant of the gradient descent method (3.4).

In the context of ODEs and PDEs, the forward Euler method as in (3.10) is known to have poor stability properties for stiff equations, putting strict requirements on the step size Δt. On the other hand, the backward Euler discretization as in (3.11) has relatively improved stability properties for stiff equations, typically permitting much larger time step sizes. These stability

ETNA
Kent State University and
Johann Radon Institute (RICAM)

properties influence the design of our parallel-in-iteration methodology described in the next section. In particular, the methodology requires us to take large steps on coarse grids, and we chose to use the proximal point method (3.5) when these steps are larger than $s > 2/L$ so as to avoid the instability of gradient descent (3.4).

Now, consider the gradient flow of $f + g$, assuming, for simplicity, that both f and g are differentiable. The associated gradient flow ODE is

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = -\nabla f(\mathbf{u}(t)) - \nabla g(\mathbf{u}(t)). \tag{3.12}$$

Applying a forward Euler step on the ∇f term and a backward Euler step on the ∇g term results in the implicit–explicit (IMEX) discretization

$$\mathbf{u}_{k+1} = (I + s\nabla g)^{-1}((I - s\nabla f)(\mathbf{u}_k)), \quad k = 0, 1, \ldots. \tag{3.13}$$

Observe that (3.13) is identical to (3.7), such that the proximal gradient method can be interpreted as an IMEX discretization of the corresponding gradient flow equation (3.12). Similarly, the alternating proximal mappings methods (3.8) is equivalent to a split fully backward-Euler-based discretization of (3.12).

4. Parallel-in-time methods. In this section we outline our parallel-in-iteration strategy for the solution of the discrete optimization problems outlined in the previous section. As outlined in Section 1, the motivation for parallelizing over iterations is that the convergence rates of these methods tend to be slow, and thus, in practice, they typically require many iterations to reach convergence. By repurposing techniques for the parallel-in-time solution of ODEs and PDEs to parallelize over iterations of optimization algorithms, we aim to reduce the wall-clock time required to solve discrete optimization problems. Specifically in this work we focus on the parallel-in-time method known as MGRIT (multigrid reduction in time) [16]. However, we note that other suitable parallel-in-time methods can also be considered: for example, the parareal method [27] or the parallel full approximation scheme (FAS) in space-time (PFASST) method [15, 32].

4.1. The MGRIT algorithm. MGRIT [16] was developed as a multilevel generalization of the two-level parareal method [27] for solving time-dependent systems of differential equations in parallel. To outline this method, let us consider a general initial-value ODE problem of the form

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) = \mathcal{L}(t, \mathbf{u}(t)), \quad \mathbf{u}(0) = \mathbf{u}_0, \quad t \in (0, T].$$

Discretizing this system of ODEs using a one-step method (e.g., any Runge–Kutta method) results in a system of equations of the form

$$\begin{aligned} \mathbf{u}_0 &= \mathbf{w}_0, \\ \mathbf{u}_i &= \Phi_i(\mathbf{u}_{i-1}) + \mathbf{w}_i, \quad i = 1, 2, \ldots, N_t, \end{aligned} \tag{4.1}$$

where $\mathbf{w}_0 \equiv \mathbf{u}_0$, and $t_i = i\Delta t$, $i = 0, 1, \ldots, N_t$, is a temporal mesh with spacing $\Delta t = T/N_t$, which we assume is constant for simplicity. Furthermore, $\mathbf{u}_i \approx \mathbf{u}(t_i)$ is the discrete approximation to $\mathbf{u}(t)$ at time t_i, and Φ_i is the *time-stepping operator*, responsible for advancing the solution at t_{i-1} to t_i; for simplicity, we further assume Φ_i is a time-independent operator, such that $\Phi_i(\cdot) = \Phi(\cdot)$ for all $i = 1, 2, \ldots, N_t$. The vector $\mathbf{w}_i$ represents solution-independent forcing terms in the discretized problem.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

We can then rewrite (4.1) as the following "all-at-once" system:

$$\mathcal{A}(\mathbf{u}) = \begin{bmatrix} I & & & \\ -\Phi(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi(\cdot) & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{N_t} \end{bmatrix} = \mathbf{w}. \tag{4.2}$$

For our purposes, it is also pertinent to introduce the associated residual vector, which, given some $\mathbf{v} \approx \mathbf{u}$, is defined as

$$\mathbf{r}(\mathbf{v}) = \mathbf{w} - \mathcal{A}(\mathbf{v}) = \begin{bmatrix} \mathbf{w}_0 - \mathbf{v}_0 \\ \mathbf{w}_1 - [\mathbf{v}_1 - \Phi(\mathbf{v}_0)] \\ \vdots \\ \mathbf{w}_{N_t} - [\mathbf{v}_{N_t} - \Phi(\mathbf{v}_{N_t-1})] \end{bmatrix}. \tag{4.3}$$

Instead of solving (4.1) sequentially with time-stepping, MGRIT aims to solve it iteratively and in parallel via the introduction of a coarser approximation of the problem. To this end, given an integer coarsening factor $m > 1$, we introduce a coarse temporal mesh $T_j = j\,\Delta T$, $j = 0, 1, \ldots, N_T$, where $N_T = N_t/m$, with constant spacing $\Delta T = m\,\Delta t$. This partitions the $N_t + 1$ points of the fine grid into two sets: (1) *C-points*, which are the $N_T + 1$ points appearing exclusively on the coarse grid, and (2) *F-points*, which are the remaining $N_t - N_T$ points that do not appear on the coarse grid. Figure 4.1 illustrates the fine- and coarse-grid temporal meshes.

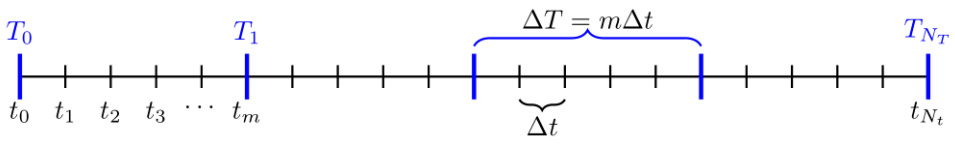


FIG. 4.1. *Fine- and coarse-grid discretization meshes with coarsening factor m. F-points (in black) are points t_i for $i = 1, \ldots, N_t$ such that $i \neq jm$ for $j = 1, \ldots, N_T = N_t/m$, and C-points (in blue) are points $T_j = t_{jm}$ for $j = 1, \ldots, N_T = N_t/m$.*

As is standard for a multilevel method, an MGRIT iteration consists of a coarse-grid correction step sandwiched between pre- and post-relaxation steps. The standard relaxation, as used here, is known as FCF-relaxation [16], and combines so-called F- and C-relaxations, both of which can be performed in parallel. An F-relaxation is the process where, starting from a C-point, the time-stepping operator is used to update the values of the $m - 1$ F-points that follow it (that is, until the F-point that immediately precedes the next C-point). C-relaxation is the update of each C-point by stepping from the preceding F-point using Φ. FCF-relaxation is simply the application of F-, C-, and F-relaxations in this order. The post-relaxation is taken to be an F-relaxation.

In between the pre- and post-relaxations, a coarse-grid correction problem is solved to provide a correction at C-points, as we now describe. MGRIT has both linear and FAS variants [25], suitable for linear and nonlinear Φ, respectively, and these variants differ in the way that coarse-grid correction is performed. Let us first explain how this works in the linear case, and to this end let us assume that Φ_i is a time-independent, linear operator, such that $\Phi_i(\mathbf{u}_{i-1}) = \Phi\mathbf{u}_{i-1}$ corresponds to a matrix–vector product with matrix Φ. We also write the action of the space-time operator $\mathcal{A}(\mathbf{u}) = \mathcal{A}\mathbf{u}$ as a matrix–vector product.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

From (4.1), we can then write, for coarse indices $j = 1, 2, \ldots, N_T$,

$$\mathbf{u}_{jm} = \Phi \mathbf{u}_{jm-1} + \mathbf{w}_{jm} = \cdots = \Phi^m \mathbf{u}_{(j-1)m} + \widetilde{\mathbf{w}}_{jm},$$

where $\widetilde{\mathbf{w}}_{jm} = \mathbf{w}_{jm} + \Phi \mathbf{w}_{jm-1} + \cdots + \Phi^{m-1}\mathbf{w}_{(j-1)m+1}$. Therefore, the fine-grid system (4.2) is equivalent to the following system written only in terms of the C-points:

$$\begin{aligned} \mathbf{u}_0 &= \mathbf{w}_0, \\ \mathbf{u}_{jm} &= \Phi^m \mathbf{u}_{(k-1)m} + \widetilde{\mathbf{w}}_{jm}, \quad j = 1, 2, \ldots, N_T, \end{aligned}$$

where Φ^m plays the role of time-stepping operator for this coarse system. Or, in block-matrix form

$$\mathcal{A}_\Delta \mathbf{u}_\Delta = \begin{bmatrix} I & & & \\ -\Phi^m & I & & \\ & \ddots & \ddots & \\ & & -\Phi^m & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_m \\ \vdots \\ \mathbf{u}_{N_T m} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \widetilde{\mathbf{w}}_m \\ \vdots \\ \widetilde{\mathbf{w}}_{N_T m} \end{bmatrix} = \mathbf{w}_\Delta. \tag{4.4}$$

While system (4.4) has a factor of m fewer equations than (4.2), it is no less expensive to solve because (naively) computing the action of Φ^m is m times as expensive as computing that of Φ. So, instead, MGRIT solves an approximation to the *ideal* coarse-grid system (4.4) arising from replacing the *ideal* coarse-grid time-stepping operator Φ^m with an approximation:

$$\Phi_\Delta \approx \Phi^m.$$

That is, rather than inverting $\mathcal{A}_\Delta$ in (4.4), one inverts the approximation $\mathcal{B}_\Delta$ given by

$$\mathcal{B}_\Delta = \begin{bmatrix} I & & & \\ -\Phi_\Delta & I & & \\ & \ddots & \ddots & \\ & & -\Phi_\Delta & I \end{bmatrix}.$$

Let us now explain in more detail how MGRIT uses the approximation $\mathcal{B}_\Delta \approx \mathcal{A}_\Delta$. Consider the coarse system $\mathcal{A}_\Delta \mathbf{u}_\Delta = \mathbf{w}_\Delta$ (4.4). Let $\mathbf{v}_\Delta \approx \mathbf{u}_\Delta$ be an approximate solution of (4.4) with error $\mathbf{e}_\Delta = \mathbf{u}_\Delta - \mathbf{v}_\Delta$. Then, note that

$$\mathcal{A}_\Delta \mathbf{e}_\Delta = \mathcal{A}_\Delta \mathbf{u}_\Delta - \mathcal{A}_\Delta \mathbf{v}_\Delta = \mathbf{w}_\Delta - \mathcal{A}\mathbf{v}_\Delta = \mathbf{r}_\Delta,$$

where $\mathbf{r}_\Delta$ denotes the coarse-grid residual. Since $\mathcal{A}_\Delta$ is invertible, it follows that $\mathbf{e}_\Delta = \mathcal{A}_\Delta^{-1}\mathbf{r}_\Delta$, such that

$$\mathbf{u}_\Delta = \mathbf{v}_\Delta + \mathbf{e}_\Delta = \mathbf{v}_\Delta + \mathcal{A}_\Delta^{-1}\mathbf{r}_\Delta.$$

Assuming that $\Phi_\Delta \approx \Phi^m$ such that $\mathcal{B}_\Delta^{-1} \approx \mathcal{A}_\Delta^{-1}$, then

$$\mathbf{u}_\Delta \approx \mathbf{v}_\Delta + \mathcal{B}_\Delta^{-1}\mathbf{r}_\Delta.$$

Let $\{\mathbf{u}_\Delta^k\}_{k=0,1,\ldots}$ be a sequence of approximate solutions generated by the $k = 0, 1, \ldots$ MGRIT iterations. Letting $\mathbf{u}_\Delta^{k+1}$ and $\mathbf{u}_\Delta^k$ play the roles of $\mathbf{u}_\Delta$ and $\mathbf{v}_\Delta$, respectively, where we think of $\mathbf{u}_\Delta^{k+1}$ as a better approximation of $\mathbf{u}_\Delta$ than $\mathbf{u}_\Delta^k$, we obtain the MGRIT residual correction scheme:

$$\mathbf{u}_\Delta^{k+1} = \mathbf{u}_\Delta^k + \mathcal{B}_\Delta^{-1}(\mathbf{w}_\Delta - \mathcal{A}_\Delta \mathbf{u}_\Delta^k), \quad k = 0, 1, \ldots.$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Component-wise, this may be written with Φ and Φ_Δ as

$$\mathbf{u}_{(j+1)m}^{k+1} = \Phi_\Delta \mathbf{u}_{jm}^{k+1} + \Phi^m \mathbf{u}_{jm}^k - \Phi_\Delta \mathbf{u}_{jm}^k + \widetilde{\mathbf{w}}_{(j+1)m},$$

where $\mathbf{u}_0^k = \mathbf{w}_0$.

Now let us consider the MGRIT coarse-grid correction in the more general FAS case, where $\Phi(\cdot)$ in (4.2) is not (necessarily) a linear operator. Let $\mathbf{v} \approx \mathbf{u}$ be an approximate solution of $\mathcal{A}(\mathbf{u}) = \mathbf{w}$ in (4.2). After performing the FCF-relaxation step, the fine-grid approximation $\mathbf{v}$ and nonlinear residual $\mathbf{w} - \mathcal{A}(\mathbf{v})$ are injected to the coarse grid:

$$\mathbf{v}_\Delta = R_I(\mathbf{v}), \quad \mathbf{r}_\Delta = R_I(\mathbf{w} - \mathcal{A}(\mathbf{v})),$$

where R_I stands for the injection operator. The *ideal* FAS coarse-grid problem is then to solve the system

$$\mathcal{A}_\Delta(\mathbf{u}_\Delta) = \mathcal{A}_\Delta(\mathbf{v}_\Delta) + \mathbf{r}_\Delta \tag{4.5}$$

for the coarse-grid solution $\mathbf{u}_\Delta$, where similarly to (4.4),

$$\mathcal{A}_\Delta(\cdot) = \begin{bmatrix} I & & & \\ -\Phi^m(\cdot) & I & & \\ & \ddots & \ddots & \\ & & -\Phi^m(\cdot) & I \end{bmatrix}.$$

The coarse-grid system (4.5) results from the elimination of F-point variables from the residual equation $\mathcal{A}(\mathbf{u}) - \mathcal{A}(\mathbf{v}) = \mathbf{r}(\mathbf{v})$. Similarly to the linear case, the MGRIT iteration solves an approximation of (4.5),

$$\mathcal{B}_\Delta(\mathbf{u}_\Delta) = \mathcal{B}_\Delta(\mathbf{v}_\Delta) + \mathbf{r}_\Delta,$$

where $\mathcal{B}_\Delta(\cdot) \approx \mathcal{A}_\Delta(\cdot)$ replaces the $\Phi^m(\cdot)$ terms in $\mathcal{A}_\Delta(\cdot)$ by a computationally cheaper $\Phi_\Delta(\cdot) \approx \Phi^m(\cdot)$. Letting $\mathbf{u}_\Delta^{k+1}$ and $\mathbf{u}_\Delta^k$ play the roles of $\mathbf{u}_\Delta$ and $\mathbf{v}_\Delta$, respectively, the component-wise FAS residual correction scheme is then given by

$$\mathbf{u}_{(j+1)m}^{k+1} = \Phi_\Delta(\mathbf{u}_{jm}^{k+1}) + \Phi(\mathbf{u}_{(j+1)m-1}^k) - \Phi_\Delta(\mathbf{u}_{jm}^k) + \mathbf{w}_{(j+1)m}, \quad j = 1, 2, \ldots, N_T.$$

We note that when $\Phi(\cdot)$ and $\Phi_\Delta(\cdot)$ are linear operators Φ and Φ_Δ, the FAS variant simplifies to the linear variant of MGRIT.

While we have described only two-level MGRIT methods, multilevel MGRIT schemes can be obtained by recursively applying the two-level schemes. In our numerical tests, we apply the standard MGRIT scheme for MP1 (2.2) and the FAS-MGRIT scheme for MP2 (2.8). For both MP1 and MP2, MGRIT is iterated until the 2-norm of the residual (4.3) is reduced by eight orders of magnitude from its initial value. We note that since the residual (4.3) is measured after performing the FCF-relaxations, the residual at the F-points will be zero, such that it is sufficient to measure the residual at the C-points.

5. MGRIT for solving optimization problems parallel in iteration. In this section, we discuss our methodology for using MGRIT to solve optimization problems in a novel parallel-in-iteration framework, using our first model problem, MP1 (2.2), as an example. First, we show the equivalence between solving MP1 using the gradient descent algorithm (3.4) and solving the 1D heat equation, in terms of the all-at-once system in (4.2). Inspired by well-known established practices for solving ODEs, we then present our heuristic for choosing a suitable coarse-grid operator for MGRIT applied to gradient descent optimization of MP1. Finally, we present theoretical results relating the convergence of approximate solutions obtained for the all-at-once system solved by MGRIT and the convergence of the original, sequential optimization algorithm.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

5.1. Links between solving ODEs using MGRIT and parallel-in-iteration optimization. Recall that the ith gradient descent iteration (3.4) for MP1, (2.2), is given by

$$\mathbf{u}_i = (I - s\nabla f)(\mathbf{u}_{i-1}) = (I - sA_1)\mathbf{u}_{i-1} + s\mathbf{b}. \tag{5.1}$$

Therefore, denoting $\Phi := (I - sA_1)$ and $\mathbf{w}_i = s\mathbf{b}$ for $i = 1, \ldots, N_t$, we can write N_t gradient descent iterations as

$$\mathcal{A}\mathbf{u} = \begin{bmatrix} I & & & \\ -\Phi & I & & \\ & \ddots & \ddots & \\ & & -\Phi & I \end{bmatrix} \begin{bmatrix} \mathbf{u}_0 \\ \mathbf{u}_1 \\ \vdots \\ \mathbf{u}_{N_t} \end{bmatrix} = \begin{bmatrix} \mathbf{w}_0 \\ \mathbf{w}_1 \\ \vdots \\ \mathbf{w}_{N_t} \end{bmatrix} = \mathbf{w}, \tag{5.2}$$

where $\mathbf{u}_0 = \mathbf{w}_0$ is the initial condition. Note that this is the same kind of system as in (4.2); in particular, it is the same system one would obtain from the forward Euler discretization of the 1D heat equation with Dirichlet boundary conditions and constant source term:

$$\begin{aligned} \frac{\partial}{\partial t}u(x,t) - \frac{\partial^2}{\partial x^2}u(x,t) &= b, \qquad (x,t) \in \Omega \times [0,T], \\ u(x,0) &= u_0(x), \qquad x \in \Omega, \end{aligned} \tag{5.3}$$

where $\Omega \subset \mathbb{R}$.

While previous research on solving (5.3) with MGRIT, namely [14, 16], certainly informs our investigation, differences in objective and approach justify a new exploration of this model problem, regardless of the similarities. First, both [14, 16] are strongly concerned with matching the accuracy of spatial and temporal discretizations of (5.3), with [16] testing different ratios between time step lengths and spatial mesh sizes and [14] employing higher-order time discretizations in order to match the spatial and temporal discretization orders. Second, with this accuracy concern in mind, [14, 16] mostly employ implicit discretizations, which are unconditionally stable in regards to time step lengths, on all levels. These objectives differ from ours significantly, as we are concerned with providing speedup over sequential solves for problems to which explicit optimization methods are typically applied, and in which the time step lengths follow a standard choice for the chosen method. Furthermore, the aforementioned accuracy concerns in both space and time are not present here, as our accuracy requirement is that the MGRIT solution at the final time points approximates the solution of the model problem to a similar tolerance accuracy as the sequential solve which we are trying to speed up. Therefore, it is necessary for us to provide our own numerical experiments and speedup estimates for this model problem, despite the similarities with [14, 16].

5.2. Choice of Φ_Δ for MP1 using gradient descent. Inspired by the similarities between our model problem (2.2) and the 1D heat equation (5.3), in this paper we follow a simple heuristic inspired by [16] of simply replacing forward Euler steps on the finest level with backward Euler steps on the coarse s. Recall from (5.1) that $\Phi := I - sA_1$. We further refine this by choosing the step size to be $s = 1/L$, which is the classical choice for gradient descent. As discussed in Section 3.3, the natural implicit candidate for the coarse-grid operator is the one in (3.5):

$$\Phi_\Delta := (I + msA_1)^{-1}. \tag{5.4}$$

That is, we use the approximation

$$\Phi_\Delta := (I + msA_1)^{-1} \approx (I - sA_1)^m = \Phi^m. \tag{5.5}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Intuitively, we are approximating m forward Euler time steps, each with step size s, with a single backward Euler time step of size ms. We recall that the proximal point method (3.5) converges for any $s > 0$, such that $ms = m/L$ is a feasible step-size choice for any coarsening factor m. In a multilevel MGRIT setting of $\widehat{\ell} \geq 2$ levels, we take $\ell = 1$ to refer to the fine grid. For $2 \leq \ell \leq \widehat{\ell}$, the ℓth level employs the operator

$$\Phi_{\Delta}^{\ell} := (I + m^{\ell-1} s A_1)^{-1}, \tag{5.6}$$

an implicit operator similar to (5.4) with the appropriate time step coarsening; when $\ell = 2$, we omit the superscript. We note that the computation of the inverse matrices in (5.4) and (5.6) amounts to solving a linear system, which we do by means of LU-factorization; the L and U factors are stored, such that the factorization is computed only once per level.

5.3. Choice of Φ_{Δ} for MP2 using proximal gradient descent. The choice of coarse-grid operator for MP2, (2.8), is analogous to that for MP1 in that we choose Φ to be the proximal gradient descent operator (3.7) and Φ_{Δ} to be its implicit analogue, the alternating proximals operator (3.8), with coarsened step size. Namely, we choose

$$\Phi := P_{sg}(I - sA_d) \quad \text{and} \quad \Phi_{\Delta}^{\ell} = P_{m^{\ell-1}sg}(I + m^{\ell-1} s A_d)^{-1}, \quad 2 \leq \ell \leq \widehat{\ell}, \tag{5.7}$$

where $g =: \lambda \|(-\bullet)_+\|$, as in (2.8). The proximal operator P_g in (5.7) is then given entry-wise by

$$(P_{\lambda\|(-\bullet)_+\|_1}(\mathbf{u}))_i = \begin{cases} (\mathbf{u})_i + \lambda, & (\mathbf{u})_i + \lambda < 0, \\ 0, & (\mathbf{u})_i \leq 0 \leq (\mathbf{u})_i + \lambda, \\ (\mathbf{u})_i, & (\mathbf{u})_i > 0, \end{cases}$$

where the i subscript denotes the ith component of the corresponding vector. The coarse-grid operators in (5.7) are chosen under the assumption that, on the first coarse grid (i.e., $\ell = 2$),

$$\Phi_{\Delta} := P_{msg}(I + msA_d)^{-1} \approx (P_{sg}(I - sA_d))^m = \Phi^m$$

is a sufficiently accurate approximation, which is consistent with the assumption in (5.5).

We note that the heuristic presented in Sections 5.2 and 5.3, by which we replace gradient descent terms by proximals, can be applied to any gradient-based method. For nonquadratic objective functions, we may need to invert a nonlinear system to compute these proximals by means of (3.2). Alternatively, we can simply compute the proximals numerically by minimizing the function in the definition (3.1) using some iterative method; this computational approach is specially useful when the proximal for the relevant function does not have a closed form. We also note that replacing the explicit gradient steps by implicit proximal steps is only required when taking steps larger than the stability limit of the explicit method. In some machine learning contexts, for example, parameter training stages are performed using arbitrarily small steps well within the stability limit, such that (proximal) gradient descent could be used on coarse levels without drawbacks. Finally, we note that our heuristic still applies in principle to more complex methods with varying step sizes and/or additional parameters. However, determining effective Φ_{Δ} can become a quite complicated task depending on the complexity of the Φ algorithm.

5.4. Gradient convergence. Given the exact fine-grid trajectory $\{\mathbf{u}_i\}$ and the MGRIT approximation $\{\mathbf{v}_i\}$ of $\{\mathbf{u}_i\}$, the MGRIT convergence criterion in our numerical experiments are set in terms of the norm of the MGRIT coarse-grid residual $\mathbf{r}(\mathbf{v}) = \mathbf{w} - \mathcal{A}(\mathbf{v})$ over the entire space-iteration domain, where $\mathbf{v} \in \mathbb{R}^{N \cdot (N_t+1)}$ (see (5.2)), while the sequential

ETNA
Kent State University and
Johann Radon Institute (RICAM)

convergence is given in terms of the gradient of the objective function $F(\mathbf{u}_i)$ at iteration i, where $\mathbf{u}_i \in \mathbb{R}^N$.

The following lemma relates these two measures of convergence by establishing a bound for the (generalized) gradient of the MGRIT approximation $\mathbf{v}_i$ at iteration i in terms of the associated error $\mathbf{e}_i = \mathbf{u}_i - \mathbf{v}_i$ and the (generalized) gradient of the exact $\mathbf{u}_i$. The setting for the lemma and subsequent remarks is the general problem of minimizing $F = f + g$ as in (3.6), where f is a convex L-smooth function, and g is a continuous, convex, and possibly nondifferentiable function. More specifically, we consider the proximal gradient iteration for minimizing $F = f + g$, with its convergence measured in terms of the generalized gradient of F:

$$G_{sF}(\mathbf{u}_i) := \frac{1}{s}[\mathbf{u}_i - P_{sg}(\mathbf{u}_i - s\nabla f(\mathbf{u}_i))] = \frac{1}{s}(I - P_{sg} \circ G_{sf}(\mathbf{u}_i)). \tag{5.8}$$

Note that $P_{sg} \circ G_{sf}$ is the same as the proximal gradient operator (3.7); in particular, $P_{sg} \circ G_{sf} = I - sG_{sF}$. Furthermore, when $g = 0$, $F = f$ and $G_{sF} = G_{sf}$. Therefore, the problem of minimizing f in (3.3) using the gradient descent method (3.4) can be seen as a special case of the problem of minimizing $F = f + g$ with the proximal gradient method (3.7). We note that for both MP1 and MP2, we take $s = 1/L$. For MP1 and MP2-1D, we have that $L = \|A_1\|_2$, where A_1 is the 1D Laplacian matrix (2.1a), while for MP2-2D, we have that $L = \|A_2\|_2$, where A_2 is the 2D Laplacian matrix (2.1b). Moreover, we consider a general parallel-in-iteration method for an all-at-once system as in (4.2), such that the following results are not particular to the MGRIT method.

LEMMA 5.1. *Let $f : \mathbb{R}^N \to \mathbb{R}$ be a convex and L-smooth function, and $g : \mathbb{R}^N \to \mathbb{R}$ a continuous convex function. Consider a parallel-in-iteration method described by an all-at-once system of the form (4.2) with $\Phi := I - sG_{sF}$ the proximal gradient operator of $f + g$ from (3.7), with step size $s = 1/L$. Let $\mathbf{v}_i \approx \mathbf{u}_i$ be an approximation of the ith component of the solution of (4.2), with associated error $\mathbf{e}_i = \mathbf{u}_i - \mathbf{v}_i$. Then, the generalized gradient (5.8) of the approximate solution satisfies*

$$\|G_{sF}(\mathbf{v}_i)\| \le L\sqrt{2}\,\|\mathbf{e}_i\| + \|G_{sF}(\mathbf{u}_i)\|. \tag{5.9}$$

Proof. Since $\mathbf{e}_i = \mathbf{u}_i - \mathbf{v}_i$, we can write $\mathbf{v}_i = \mathbf{u}_i - \mathbf{e}_i$. By [4, Proposition 29.1], $\Phi := I - sG_{sF}$ is $2/3$-averaged. Therefore,

$$\begin{aligned}\frac{1}{2}\|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i)\|^2 &\le \|\mathbf{v}_i - \mathbf{u}_i\|^2 - \|(I - sG_{sF})(\mathbf{v}_i) - (I - sG_{sF})(\mathbf{u}_i)\|^2 \\ &\le \|\mathbf{v}_i - \mathbf{u}_i\|^2 = \|\mathbf{e}_i\|^2.\end{aligned}$$

Taking the square root on both sides, we get

$$\|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i)\| \le \sqrt{2}\,\|\mathbf{e}_i\|.$$

Using the triangle inequality, we get

$$\begin{aligned}\|sG_{sF}(\mathbf{v}_i)\| &= \|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i) + sG_{sF}(\mathbf{u}_i)\| \\ &\le \|sG_{sF}(\mathbf{v}_i) - sG_{sF}(\mathbf{u}_i)\| + \|sG_{sF}(\mathbf{u}_i)\| \\ &\le \sqrt{2}\,\|\mathbf{e}_i\| + s\|G_{sF}(\mathbf{u}_i)\|.\end{aligned}$$

Therefore,

$$\begin{aligned}\|G_{sF}(\mathbf{v}_i)\| &\le \frac{\sqrt{2}}{s}\,\|\mathbf{e}_i\| + \|G_{sF}(\mathbf{u}_i)\| \\ &= L\sqrt{2}\,\|\mathbf{e}_i\| + \|G_{sF}(\mathbf{u}_i)\|. \qquad \square\end{aligned}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

Bound (5.9) in Lemma 5.1 shows that, when $\mathbf{u}_i$ approximates a minimizer of $F(\mathbf{u})$ with accuracy $\|G_{sF}(\mathbf{u}_i)\|$, then our parallel-in-iteration method will approach the same accuracy $\|G_{sF}(\mathbf{v}_i)\|$ for $\mathbf{v}_i$ if we run sufficient iterations of the parallel-in-iteration scheme to make $\|\mathbf{e}_i\|$ small relative to $\|G_{sF}(\mathbf{u_i})\|/(L\sqrt{2})$. This can also be restated in terms of residuals rather than errors; we present the result for the linear case in the following remark.

REMARK 5.2. Recall the all-at-once system (4.2) and the related residual equation (4.3). If Φ is a linear operator such that Lemma 5.1 holds, then $\mathcal{A}\mathbf{e} = \mathbf{r}$, such that $\mathbf{e} = \mathcal{A}^{-1}\mathbf{r}$. Therefore, $\|\mathbf{e}_i\| \leq \|\mathbf{e}\| = \|\mathcal{A}^{-1}\mathbf{r}\| \leq \|\mathcal{A}^{-1}\|\|\mathbf{r}\|$, which together with (5.9) implies

$$\|G_{sF}(\mathbf{v}_i)\| \leq L\sqrt{2}\,\|\mathcal{A}^{-1}\|\|\mathbf{r}\| + \|G_{sF}(\mathbf{u}_i)\|. \tag{5.10}$$

As such, considering large $i \approx N_t$, we find that a sufficiently small space-time residual implies accurate approximation of the minimizer.

In particular, let $\mathbf{u}_i^k$ be the approximate solution at time i and iteration k, and let $\mathbf{e}_i^k = \mathbf{u}_i - \mathbf{u}_i^k$ be the error of the approximation. Similarly, let $\mathbf{u}^k$ be the approximate solution vector at iteration k. Next, we consider the assumption of linear convergence of the all-at-once system. For MGRIT, linear convergence has been proven for certain linear problems as in [13, 14, 42], and widely observed numerically even for nonlinear problems [11, 17], including in the results we present in Section 6.2.

REMARK 5.3. Suppose the parallel-in-iteration method described in Lemma 5.1 converges linearly in error for the problem of minimizing $F = f + g$ with convergence factor $\rho_{\mathbf{e}}$, that is,

$$\|\mathbf{e}^k\| \leq \rho_{\mathbf{e}}\|\mathbf{e}^{k-1}\| \leq (\rho_{\mathbf{e}})^k\|\mathbf{e}^0\|.$$

for $\rho_{\mathbf{e}} \in (0,1)$ some constant independent of k. Then, from (5.9) we get linear convergence of the gradient at any point i in time:

$$\|G_{sF}(\mathbf{u}_i^k)\| \leq (L\sqrt{2}\,\|\mathbf{e}^0\|)(\rho_{\mathbf{e}})^k + \|G_{sF}(\mathbf{u}_i)\|.$$

In particular, suppose that after $k = k_*$ iterations $(L\sqrt{2}\,\|\mathbf{e}^0\|)(\rho_{\mathbf{e}})^{k_*} := \epsilon_*$ is some sufficiently small number. Then,

$$\|G_{sF}(\mathbf{u}_i^k)\| \leq \epsilon_* + \|G_{sF}(\mathbf{u}_i)\|.$$

If we have linear convergence in the residual, that is,

$$\|\mathbf{r}^k\| \leq \rho_{\mathbf{r}}\|\mathbf{r}^{k-1}\| \leq (\rho_{\mathbf{r}})^k\|\mathbf{r}^0\|,$$

for $\rho_{\mathbf{r}} \in (0,1)$ some constant independent of k, then from (5.10) we obtain an analogous result for convergence of the gradient in terms of the residual for a linear Φ:

$$\|G_{sF}(\mathbf{u}_i^k)\| \leq (L\sqrt{2}\,\|\mathcal{A}^{-1}\|\|\mathbf{r}^0\|)(\rho_{\mathbf{r}})^k + \|G_{sF}(\mathbf{u}_i)\|. \tag{5.11}$$

In particular, we conclude that when using MGRIT with the gradient descent fine-grid operator (3.4) for MP1 (2.2), and the proximal gradient descent fine-grid operator (3.7) for MP2 (2.8), minimizing the MGRIT residual implies minimizing the gradient of $\mathbf{u}_i^k$ to a similar accuracy as that of $\mathbf{u}_i$, with (5.11) providing a bound that can inform the number of iterations required to make $\|G_{sF}(\mathbf{u}_i^k)\|$ close to $\|G_{sF}(\mathbf{u}_i)\|$. For example, assuming a desired accuracy of the minimizer of $\|G_{sF}(\mathbf{u}_{N_t})\| = \tau$, one could guarantee accuracy 2τ of the MGRIT approximation $\mathbf{u}_{N_t}^{k_*}$ with at most k_* iterations, where k_* is determined according to $\tau = L\sqrt{2}\,\|\mathcal{A}^{-1}\|\|\mathbf{r}^0\|(\rho_r)^{k_*}$.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

6. Numerical results. Here, we present results of our parallel-in-iteration algorithm applied to the optimization problems MP1 (2.2) and MP2 (2.8) discretized with the optimization methods discussed in Sections 3.1 and 3.2, respectively. It is important to note that we are not currently implementing MGRIT in parallel, which will be done in future work. The goal of the current paper is to present the method for optimization problems and investigate its convergence speed (the number of MGRIT iterations required to obtain an accurate solution) numerically, which is done using a serial implementation. Later in Section 7, we also provide estimates for speedup in a parallel setting. We note that while MGRIT by construction approximates the entire trajectory $\{\mathbf{u}_i\}$, $0 \leq i \leq N_t$, of the sequential solve, we are for optimization problems really interested in calculating the final approximation $\mathbf{u}_{N_t}$, which is the MGRIT approximate solution at the final iteration point N_t, such that we give special importance to $\mathbf{u}_{N_t}$.

First, we solve the problem sequentially using gradient descent or proximal gradient descent. Next, we solve the same problem with MGRIT, i.e., we use MGRIT aiming to accelerate the computation of the fine-grid trajectory $\{\mathbf{u}_i\}$ in parallel. On coarse levels in the MGRIT hierarchy we use an implicit variant of the optimization method used on the finest level. We illustrate convergence results for a particular set of choices for the parameters N (dimension of the Laplacian matrix), m (coarsening factor), and $\widehat{\ell}$ (number of levels). Although we only present results for the fine-level step-size choice $s = 1/L$, further tests (not shown here for brevity) indicate that MGRIT performs similarly well for larger step sizes up to the stability limit of $2/L$. For our initial examples, we chose $\widehat{\ell} = 2$ and $m = 4$. Later, we present results for different values of m, $\widehat{\ell}$, and N to analyze how MGRIT's performance is affected by the choices of these parameters. We choose 10^{-8} as the tolerance criterion for all iterative methods. For sequential solves, this applies to the reduction in the 2-norm of the gradient relative to its original value. For MGRIT solves, the tolerance criterion refers to the reduction on the 2-norm of the MGRIT residual, (4.3). For MP1, (2.2), we measure the gradient of the quadratic function f, while for MP2, (2.8), we measure the generalized gradient, (5.8), of the EOP function of the form $F := f + g$. The initial MGRIT guess $\mathbf{u}^0 \in \mathbb{R}^N$ is chosen to be random with components in $[0, 1]$, after which the condition $\mathbf{u}_0 = \mathbf{g}_0$ is enforced.

We note that the sequential solve provides us with N_t, exactly the number of iterations required for convergence within the given tolerance threshold, which we then use to specify the MGRIT fine-grid time domain to be $[0, (N_t - 1)/L]$, with N_t fine-grid time points. However, solving the problem sequentially before solving it with MGRIT is of course not recommended in practice. Instead, the final time should be chosen adaptively, for example, using the following strategy. Given some tolerance threshold to which we want to reduce the gradient, we take an educated guess at the final time required to achieve this reduction. This guess can possibly be informed by theoretical or practically observed convergence rates of the underlying optimization method. Setting the MGRIT time window with this guess for the final time, we run MGRIT until either the gradient at the final point reaches the prescribed tolerance, or the gradient at the final point stalls.

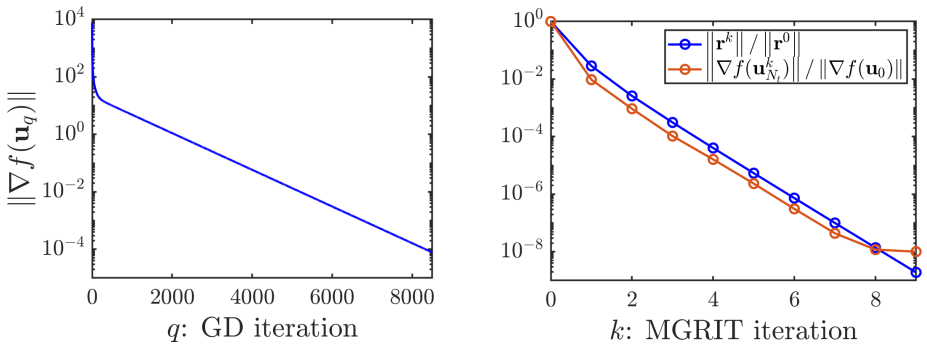
In the case where the gradient reaches the prescribed tolerance, we either have precisely guessed the correct final time, such that the number of time points is exactly what we would get by solving the problem sequentially first, or we have overestimated the final time such that we have too many time points. Otherwise, if the gradient at the final time point stalls, then we have underestimated the final time and do not have enough time points to achieve the desired gradient reduction. Iterating MGRIT past this point is redundant, since MGRIT will only reduce the residual of the underlying algebraic system (4.3), but without reducing the gradient. In that case, we must start a new MGRIT solve across a new time interval, starting from the final time point of the halted solve and setting the solution at the final time point of

ETNA
Kent State University and
Johann Radon Institute (RICAM)

the halted solve as the initial guess for the new solve's all-at-once system (4.2); essentially, this is the same as extending the original trajectory. Then, we take a new educated guess for the final time, which could be estimated based on the approximate convergence rate of the optimization method observed from the halted solve. Having set the new time horizon, we run MGRIT again and repeat this process until the desired gradient reduction is obtained.

Following this strategy, it may be that underestimating or overestimating the final time does not lead to performing any more work than when perfectly guessing the final time. However, we might lose some parallel speedup efficiency, as in both cases we do more sequential work than when we work with the ideal number of time points, in which case the parallelization is perfectly optimized. For our discussion on parallel speedup efficiency; see Section 7.

6.1. MGRIT for gradient descent applied to MP1. We solve (2.2) using gradient descent, (3.4). The 1D Laplacian matrix $A_1 \in \mathbb{R}^{n\times n}$ in (2.1a) with 2-norm L is taken with $n = 40$, the coarsening factor is chosen to be $m = 4$, and the number of levels to be $\widehat{\ell} = 2$. We also choose $\mathbf{b} \in \mathbb{R}^n$ to be random with components in $[0, 1]$. For MGRIT, we choose the proximal point operator (3.5) as the coarse-grid operator. First, we solve the problem sequentially, with $N_t = 8503$ iterations required for the convergence of the gradient descent algorithm (3.4). We then generate the MGRIT fine-grid $t_i = i\Delta t$, $i = 0, \dots, N_t$, with $\Delta t = 1/L$. Next, we generate the coarse-grid $T_j = j\Delta t$, $j = 0, 1, \dots, N_T$, with $\Delta t = m\Delta t = 4/L$, such that the coarse grid has $N_T = 2126$ points. MGRIT then solves (4.2). Convergence for the sequential and MGRIT solves is illustrated in Figure 6.1.



(a) Sequential convergence

(b) MGRIT convergence

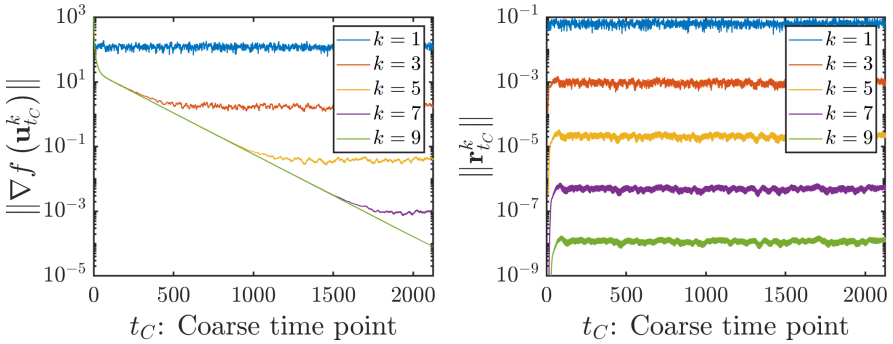
FIG. 6.1. *(a) Convergence of MP1, (2.2), with a standard sequential solve using gradient descent method (3.4); $\nabla f(\mathbf{u}_q)$ denotes the gradient of the qth sequential iterate. (b) Convergence of MP1, (2.2), with MGRIT using gradient descent operator (3.4) on the fine grid and proximal point operator (3.5) on the coarse grid; $\mathbf{r}^k$ denotes residual (4.3) at iteration k, $\nabla f(\mathbf{u}^k_{N_t})$ denotes the gradient of the approximate solution at the final time point N_t and MGRIT iteration k, $\nabla f(\mathbf{u}_0)$ denotes the gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^0$ denotes the initial MGRIT residual.*

In Figure 6.1, we see the MGRIT convergence in terms of both the MGRIT residual and the gradient at the final time point N_t. We recall that the MGRIT convergence criterion is given in terms of space-time residual (4.3), but using the gradient of the solution at the final time point $\mathbf{u}^k_{N_t}$ of each MGRIT iteration as reference, we see that the gradient of the function is also reduced by a similar factor. Recalling Remark 5.3, we can interpret (5.11) as follows. If we look at a semi-log plot of $\|\nabla f(\mathbf{u}^k_i)\|$ as a function of k, as in Figure 6.1(b), then it will be upper-bounded by a straight line with slope equal to $\rho_{\mathbf{r}} \in (0, 1)$, which is the slope of the residual curve, up until the point where $\|\nabla f(\mathbf{u}_i)\| \approx (L\sqrt{2}\,\|\mathcal{A}^{-1}\|\|\mathbf{r}^0\|)(\rho_{\mathbf{r}})^k$, at which

ETNA
Kent State University and
Johann Radon Institute (RICAM)

point the curve flattens out to a constant value of $\|\nabla f(\mathbf{u}_i)\|$. Therefore, the numerical results presented here are consistent with our theoretical predictions in Section 5.4.

In Figure 6.2, we further explore the behavior of the gradient and the residual (4.3) as MGRIT converges. As expected, we see that, as k increases, the gradient curve for the kth iteration in Figure 6.2(a) approximates the curve in Figure 6.1(a) increasingly closely. We note that in both Figure 6.2(a) and (b), the gradient and MGRIT residual are reduced by a consistent factor for each iteration. We also note that the behavior on the initial time points of each iteration in Figure 6.2(b) is consistent with the exactness property of MGRIT with FCF-relaxations; for $m = 4$, at the kth MGRIT iteration, the first $2mk = 8k$ fine points are solved exactly. Figure 6.2(b) shows that MGRIT converges in a small number of iterations, which is crucial for obtaining parallel speedup; see Section 7.



(a) Gradient convergence by iteration

(b) Residual convergence by iteration

FIG. 6.2. *(a) The 2-norm of $\nabla f(\mathbf{u}^k_{t_C})$, where ∇f is the gradient of f and $\mathbf{u}^k_{t_C}$ is the approximate solution at coarse time point t_C and iteration k. (b) The 2-norm of $\mathbf{r}^k_{t_C}$, where $\mathbf{r}^k_{t_C}$ is the residual (4.3) at coarse time point t_C and iteration k.*

In Table 6.1, we present MGRIT convergence results for MP1 (2.2) with different coarsening factors m and number of MGRIT levels $\widehat{\ell}$. We note that the larger the coarsening factor, the smaller the maximum number of allowed levels.

TABLE 6.1
MGRIT convergence results for different values of m and $\widehat{\ell}$, using gradient descent operator (3.4) on the fine grid and proximal point operator (3.5) on the coarse grid.

$n = 40$, $N_t = 8503$

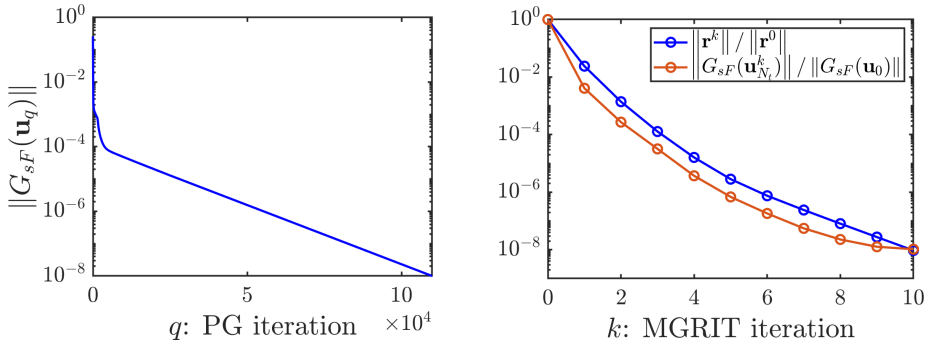
	$\widehat{\ell} = 2$	$\widehat{\ell} = 3$	$\widehat{\ell} = 4$	$\widehat{\ell} = 5$	$\widehat{\ell} = 6$	$\widehat{\ell} = 7$
$m = 4$	9	9	9	9	9	9
$m = 16$	8	8	8			
$m = 64$	8	8				
$m = 256$	7					

From Section 5.1 we recall the connection of MP1 (2.2) to the 1D heat equation, for which the MGRIT convergence behavior is well understood [14, 16]. For parabolic problems, MGRIT has been observed to have a desirable property of multigrid convergence known as h-independent convergence [47], which means that the convergence speed does not depend on the size of the fine grid.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

6.2. MGRIT for proximal gradient applied to MP2. We now consider solving the EOP problem, MP2 (2.8), using the proximal gradient method (3.7). For MGRIT, we choose the alternating proximals operator (3.8) to be the coarse-grid operator. We noted in Section 3.2 how the sequence generated by alternating proximals does not necessarily converge to the same set of minimizers as that of proximal gradient. However, the similarity between these operators justifies the use of the alternating proximal method as the coarse-grid method (recall discussions in Sections 3.3 and 5 on choosing coarse-grid operators).

6.2.1. Numerical results for MP2-1D. For the MP2-1D example, we consider MP2 in (2.8) with $d = 1$ and $N = n = 256$, with MGRIT parameters $m = 4$ and $\widehat{\ell} = 2$. We again start by presenting results for the sequential and MGRIT convergence in Figure 6.3. In this example, the proximal gradient method requires $N_t = 109\,888$ iterations to reach the prescribed halting tolerance, i.e., that the generalized gradient of F, G_{sF} in (5.8), be reduced by eight orders of magnitude. MGRIT reaches the prescribed residual tolerance in 10 iterations.



(a) Sequential convergence

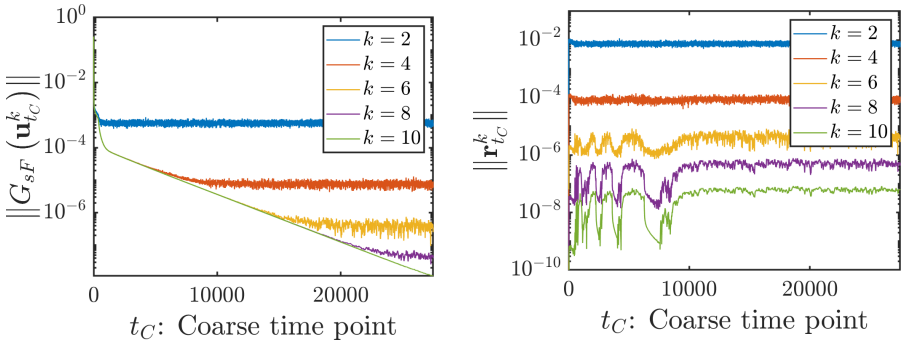
(b) MGRIT convergence

FIG. 6.3. *(a) Convergence of MP2-1D, (2.8), with a standard sequential solve using the proximal gradient descent method (3.7); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.8) of the qth sequential iterate. (b) Convergence of MP2-1D, (2.8), with MGRIT using proximal gradient operator (3.7) on the fine grid and alternating proximal mappings operator (3.8) on the coarse grid; $\mathbf{r}^k$ denotes residual (4.3) at iteration k, $G_{sF}(\mathbf{u}^k_{N_t})$ denotes the generalized gradient (5.8) of the approximate solution at the final time point N_t and MGRIT iteration k, $G_{sF}(\mathbf{u}_0)$ denotes the generalized gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^0$ denotes the initial MGRIT residual.*

As in Section 6.1, it is interesting to investigate the relation between the MGRIT residual (4.3) and the generalized gradient (5.8); see Figure 6.4. Much like in Figure 6.2(a), the gradient in Figure 6.4(a) is consistently reduced across iterations. The space-time residual in Figure 6.4(b), however, has an unusual behavior over the first coarse time points as the number of iterations increases. To further explore this, in Figure 6.5 we look at the spatial components of MGRIT residual vector $\mathbf{r}^k$ (4.3) at iteration k in order to relate the residual to the exact solution $\widehat{u}$ and obstacle ϕ of the 1D EOP in (2.4); see Figure 2.1.

With the solution $\widehat{u}$ and obstacle ϕ in Figure 2.1 overlaid on the residual (4.3) in Figure 6.5, we see that the residual decreases more slowly and less smoothly around the spatial coordinates where the membrane attaches to the obstacle and the problem is nonsmooth. This nonsmoothness causes the slope of the convergence curves in Figure 6.3(b) to taper off, compared to Figure 6.1(b). Looking back at Figure 6.4(b), we observe that for large k, when MGRIT approaches the exact trajectory $\{\mathbf{u}_i\}$, the MGRIT residual is initially nonsmooth as a function of time point. This reflects that, at early time points, the set of spatial points where

ETNA
Kent State University and
Johann Radon Institute (RICAM)



(a) Gradient convergence by iteration

(b) Residual convergence by iteration

FIG. 6.4. *(a) The 2-norm of $G_{sF}(\mathbf{u}^k_{t_C})$, where G_{sF} is the generalized gradient of F (5.8), and $\mathbf{u}^k_{t_C}$ is the approximate solution at coarse time point t_C and iteration k. (b) The 2-norm of $\mathbf{r}^k_{t_C}$, where $\mathbf{r}^k_{t_C}$ is the residual (4.3) at coarse time point t_C and iteration k.*

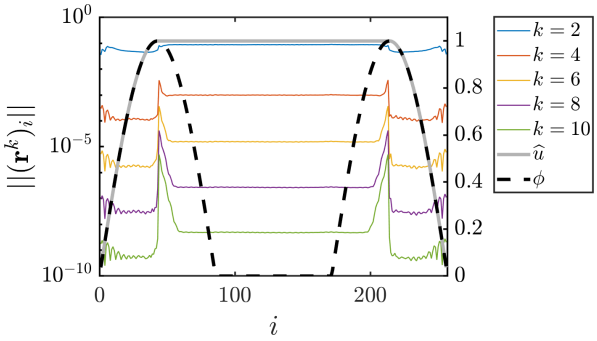


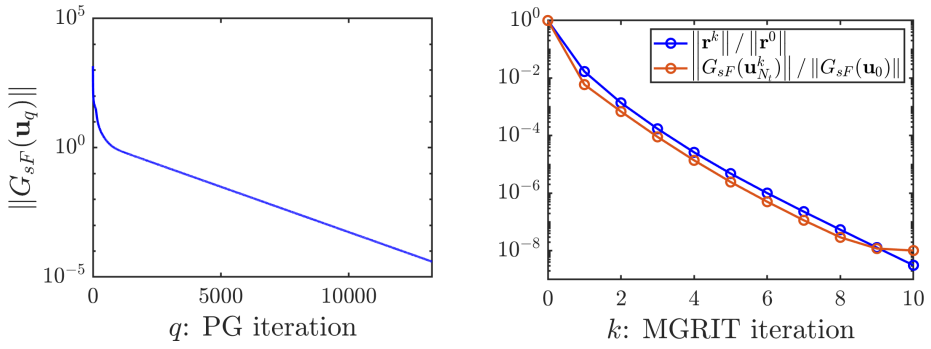
FIG. 6.5. *Left axis: the 2-norm of $(\mathbf{r}^k)_i$, where $(\mathbf{r}^k)_i$ is the ith spatial component of residual (4.3) at iteration k (colored lines). Right axis: $\widehat{u}$ (gray full curve) is the exact solution and ϕ (black dashed curve) is the obstacle of the 1D EOP, as in Figure 2.1. The i points form a 256-point discretization of the interior of the domain $\Omega = [0, 3\pi]$ of both $\widehat{u}$ and ϕ.*

the nonsmooth constraint is active is not fully determined yet. Once the active set is fixed, the residual behaves similarly to how it does for MP1 (2.2), as seen in Figure 6.2(b).

6.2.2. Numerical results for MP2-2D. For the MP2-2D example, we consider MP2 in (2.8) with $d = 2$ and $N = n^2 = 64^2$, with MGRIT parameters $m = 4$ and $\widehat{\ell} = 2$. As in Section 6.1, we start by presenting the sequential and MGRIT convergence results in Figure 6.6, noting that the sequential solver required $N_t = 13\,256$ iterations to converge. We then compare the generalized gradient (5.8) and the MGRIT residual (4.3) at the coarse time points in Figure 6.7. As with Figure 6.5 for the MP2-1D, we also look at the 2-norm of the spatial components of the residual (4.3) in Figure 6.8. This time, in order to present a 3D plot, we only present the residual at the final iteration $k = 10$.

Comparing Figure 6.8 to Figure 2.1, we see a similar relationship between the residual, the solution, and the obstacle as we did with Figures 6.5 and 2.1 for MP2-1D. To better illustrate this relationship, we present contour plots in Figure 6.9, from which we observe that the residual of the final MGRIT iteration is larger in the regions where the membrane attaches to the obstacle, as with the 1D case. Overall, we observe that MGRIT for MP2-2D

ETNA
Kent State University and
Johann Radon Institute (RICAM)



(a) Sequential convergence (b) MGRIT convergence

FIG. 6.6. *(a) Convergence of MP2-2D, (2.8), with a standard sequential solve using the proximal gradient descent method (3.7); $G_{sf}(\mathbf{u}_q)$ denotes the generalized gradient (5.8) of the qth sequential iterate. (b) Convergence of MP2-2D, (2.8), with MGRIT using proximal gradient operator (3.7) on the fine grid and alternating proximal mappings operator (3.8) on the coarse grid; $\mathbf{r}^k$ denotes the residual (4.3) at iteration k, $G_{sF}(\mathbf{u}^k_{N_t})$ denotes the generalized gradient (5.8) of the approximate solution at the final time point N_t and MGRIT iteration k, $G_{sF}(\mathbf{u}_0)$ denotes the generalized gradient of the initial guess $\mathbf{u}_0$, and $\mathbf{r}^0$ denotes the initial MGRIT residual.*

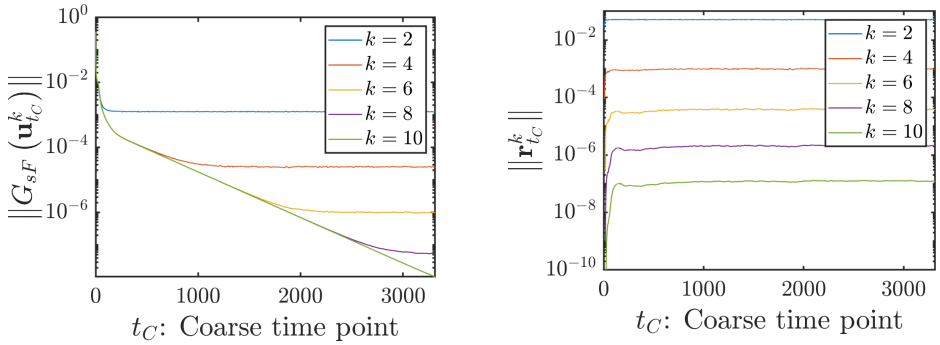


FIG. 6.7. *(a) The 2-norm of $G_{sF}(\mathbf{u}^k_{t_C})$, where G_{sF} is the generalized gradient of F, (5.8), and $\mathbf{u}^k_{t_C}$ is the approximate solution vector at coarse time point t_C and iteration k. (b) The 2-norm of $\mathbf{r}^k_{t_C}$, where $\mathbf{r}^k_{t_C}$ is residual (4.3) at coarse time point t_C and iteration k.*

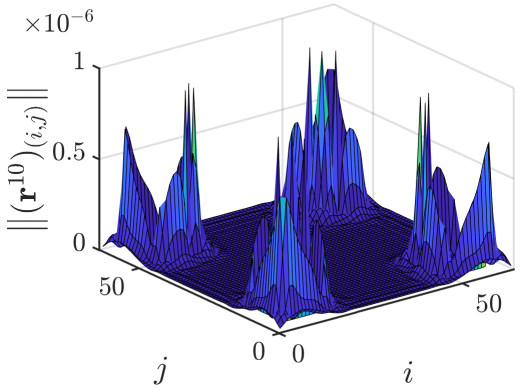


FIG. 6.8. *The 2-norm of $(\mathbf{r}^{10})_{(i,j)}$, where $(\mathbf{r}^{10})_{(i,j)}$ is the (i,j)th spatial component of residual (4.3) for the 10th and final MGRIT iteration.*

ETNA
Kent State University and
Johann Radon Institute (RICAM)

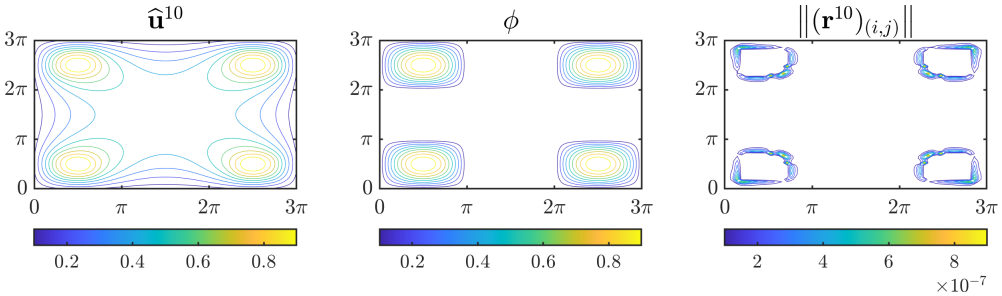


FIG. 6.9. *Contour plots of the approximate solution $\widehat{\mathbf{u}}$ and obstacle ϕ for the 2D EOP, as in Figure 2.1, and the residual of the 10th and final MGRIT iteration in Figure 6.8.*

behaves similarly to that described for MP2-1D in Section 6.1, and MGRIT convergence for both MP2-1D and MP2-2D behaves according to the theory in Section 5.4.

As with MP1 in Table 6.1, we investigate the convergence results for MP2, (2.8), with different coarsening factors m and number of MGRIT levels $\widehat{\ell}$. The results are shown in Tables 6.2 and 6.3. This time, we also consider different sizes for the matrices A_1 in (2.1a) for MP2-1D and A_2 in (2.1b) for MP2-2D. Overall, MGRIT consistently converges in a small number of iterations, as for MP1 in Table 6.1, and we obtain iteration numbers that are almost independent of problem size.

TABLE 6.2
MP2-1D convergence results for different values of n, m, and $\widehat{\ell}$, using proximal gradient operator (3.7) on the fine grid and alternating proximals operator (3.8) on the coarse grid.

$n = 256$, $N_t = 109\,888$

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$	$\widehat{\ell}=8$
$m=4$	10	10	11	12	12	12	13
$m=16$	8	9	10	10			
$m=64$	10	10	10				
$m=256$	10	10					
$m=1024$	11						

$n = 512$, $N_t = 369\,612$

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$	$\widehat{\ell}=8$	$\widehat{\ell}=9$	$\widehat{\ell}=10$
$m=4$	9	12	11	11	12	12	12	12	12
$m=16$	10	10	11	11					
$m=64$	8	10	10						
$m=256$	9	9							
$m=1024$	10								

7. MGRIT parallel speedup estimates. In our numerical results, we verified that MGRIT was successful in solving both MP1, (2.2), and MP2, (2.8), in a small number of iterations. However, each MGRIT iteration does considerably more work than a full sequential solve. Since we are interested in reducing the wall-clock time while solving in parallel, we need to compute the relative cost of the MGRIT solve compared to the sequential solve, and estimate how much time would be saved by solving the problem in parallel. For that, we must first derive a formula for estimating the wall-clock time speedup as a function of the

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 6.3
MP2-2D convergence results for different values of n, m, and $\widehat{\ell}$, using proximal gradient operator (3.7) on the fine grid and alternating proximals operator (3.8) on the coarse grid.

$n^2 = 64^2$, $N_t = 13\,256$

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$
$m=4$	10	13	13	13	13	13
$m=16$	11	12	12			
$m=64$	10	10				
$m=256$	9					

$n^2 = 128^2$, $N_t = 46\,431$

	$\widehat{\ell}=2$	$\widehat{\ell}=3$	$\widehat{\ell}=4$	$\widehat{\ell}=5$	$\widehat{\ell}=6$	$\widehat{\ell}=7$	$\widehat{\ell}=8$
$m=4$	10	10	10	10	11	11	11
$m=16$	9	9	9				
$m=64$	9	9					
$m=256$	9						

relevant parameters. Based on [41] and [32], which provide speedup estimates for the parareal method, we derive a formula for the MGRIT speedup estimate. In the following estimates, we assume that parallel communication costs are negligible, as assumed for the parareal speedup estimates in [32, 41].

7.1. Estimating two-level MGRIT wall-clock time speedup. First, we consider a two-level MGRIT scheme as in Section 4.1. Let N_f denote the number of time intervals on the fine-grid level $\ell = 1$, and let t_f denote the wall-clock time required to apply the fine-grid time-stepping operator Φ once. Similarly, let N_c denote the total number of time intervals on the coarse-grid level $\ell = 2$, and let t_c denote the wall-clock time required to apply the coarse-grid time-stepping operator Φ_Δ once. The coarsening factor, m, is given by $m = N_f/N_c$. Let N_p denote the number of processors; full parallelization is obtained when we take $N_p = N_c = N_f/m$, such that we allot each coarse-grid step to its own processor, along with m fine-grid steps. Finally, let N_{it} be the number of MGRIT iterations required for convergence to a set tolerance.

The speedup, given by the ratio of the computation time for the sequential solve and the MGRIT solve, is given by

$$S_2(N_p) = \frac{N_f t_f}{N_{it}(N_c t_c + 2N_f t_f/N_p)}. \tag{7.1}$$

The numerator $N_f t_f$ represents the total cost of the sequential solve. In the denominator, we have the cost of the MGRIT solve: For each of the N_{it} MGRIT iterations, the $2N_f t_f/N_p$ term accounts for the fine-grid cost of the parallel FCF-relaxation and correcting the coarse-grid solution, which in total amount to the cost of two full sequential solves that are solved in parallel by N_p processors. The $N_c t_c$ term accounts for the cost of computing the C-point residual. Recalling that $N_f/N_c = m$ and assuming full parallelization (i.e., $N_p = N_f/m$), we rearrange (7.1) to find

$$\begin{aligned} S_2(m) &= \frac{N_f t_f}{N_{it}(N_c t_c + 2N_f t_f/N_p)} \\ &= \frac{1}{N_{it}\left(\frac{1}{m}\frac{t_c}{t_f} + \frac{2}{N_p}\right)} = \frac{1}{N_{it}\left(\frac{1}{m}\frac{t_c}{t_f} + \frac{2m}{N_f}\right)}. \end{aligned} \tag{7.2}$$

ETNA
Kent State University and
Johann Radon Institute (RICAM)

In Section 6, we observed that N_{it} does not vary significantly with m, so in what follows we assume it is a constant independent of m. We further assume t_c is independent of m, such that setting the derivative of S_2 with respect to m equal to zero then yields

$$S_2'(m) = 0 \iff -\frac{1}{m^2}\frac{t_c}{t_f} + \frac{2}{N_f} = 0 \iff m = \sqrt{\frac{N_f}{2}\frac{t_c}{t_f}}, \tag{7.3}$$

where for simplicity we assume the minimizer $\sqrt{(N_f/2)t_c/t_f}$ is an integer, such that we can choose $m = m^* := \sqrt{(N_f/2)t_c/t_f}$ as the optimal coarsening factor; in our results in the tables below, we round $\sqrt{(N_f/2)t_c/t_f}$ to the nearest integer. Denote $\alpha := t_c/t_f$, such that $m^* = \sqrt{\alpha N_f/2}$. Taking $m = m^*$ in (7.2), and simplifying gives

$$\begin{aligned} S_2(m^*) &= \frac{1}{N_{it}\left(\frac{1}{\sqrt{\alpha N_f/2}}\alpha + 2\frac{\sqrt{\alpha N_f/2}}{N_f}\right)} \\ &= \frac{1}{N_{it}\left(\sqrt{\frac{2\alpha}{N_f}} + \sqrt{\frac{2\alpha}{N_f}}\right)} = \frac{\sqrt{N_f}}{N_{it}2\sqrt{2\alpha}}. \end{aligned} \tag{7.4}$$

We observe that the optimal speedup is a function of N_f, N_{it}, and $\alpha := t_c/t_f$. Since the optimal speedup is proportional to $\sqrt{N_f}$, it is clear that more speedup can be obtained and MGRIT will be more effective when the fine-level optimization method converges slowly and requires a large N_f. Similarly, to keep N_{it} small it is important to choose a coarse-grid operator that closely matches the fine operator, see [9, 14]. Finally, α can be brought close to 1 by making t_c as close as possible to t_f, for example, by optimizing the implicit computations done with the coarse-grid operator or by substituting it by a different one with smaller t_c, without increasing N_t.

7.2. Estimating three-level MGRIT wall-clock time speedup. Next, we introduce a third level, $\ell = 3$, which is obtained from coarsening the $\ell = 2$ coarse-grid level by the same coarsening factor m. Associated with $\ell = 3$ is the time-stepping operator Φ_Δ^3, defined according to (5.6) (i.e., an operator similar to Φ_Δ with an m times larger step size), which takes $m^2\Delta t$-sized steps. Let N_{cc} be the number of time intervals on the $\ell = 3$ level, and let t_{cc} denote the wall-clock time required to compute one Φ_Δ^3 step. We assume that $t_{cc} = t_c$, since we use the same implicit scheme on both coarse levels $\ell = 2$ and $\ell = 3$, and $N_{cc} = N_c/m = N_f/m^2$, assuming N_{cc} is an integer.

For $\widehat{\ell} = 3$, we get a speedup estimate formula of

$$S_3(N_p) = \frac{N_f t_f}{N_{it}\left(N_{cc}t_{cc} + \frac{3N_c t_c}{\min\{N_p, N_c/m\}} + \frac{2N_f t_f}{N_p}\right)}.$$

The reasoning is similar to that for $\widehat{\ell} = 2$, with the difference that here we replace the original coarse-grid work in (7.1), with cost $N_c t_c$, with a new two-grid MGRIT solve. Note that on the $\ell = 2$ level, a post F-relaxation step must be done to distribute the correction from the coarse-grid solve in $\ell = 3$, such that the total relaxation cost on the $\ell = 2$ grid is roughly that of three sequential coarse solves. Since there are only N_c/m intervals on the first coarse level, we can use at most N_c/m processors on that level. Assuming full parallelization on the fine

ETNA
Kent State University and
Johann Radon Institute (RICAM)

level $N_p = N_c$, we have that $\min\{N_p, N_c/m\} = \min\{N_c, N_c/m\} = N_c/m$. In that case,

$$S_3(N_p) = \frac{N_f t_f}{N_{it}\left(N_{cc}t_{cc} + \frac{3N_c t_c}{N_c/m} + \frac{2N_f t_f}{N_c}\right)} = \frac{1}{N_{it}\left(\frac{1}{m^2}\frac{t_c}{t_f} + \frac{2m}{N_f}\left(1 + \frac{3}{2}\frac{t_c}{t_f}\right)\right)}. \tag{7.5}$$

Following a similar reasoning to that for finding the optimal two-level coarsening factor (7.3), we set the derivative of S_3 with respect to m equal to zero and get

$$S_3'(N_p) = 0 \iff -\frac{1}{m^3}\frac{t_c}{t_f} + \frac{2}{N_f} + \frac{2}{N_f}\frac{3}{2}\frac{t_c}{t_f} = 0 \iff m = \sqrt[3]{\frac{N_f}{2}\left(\frac{1}{t_f/t_c + 3/2}\right)}. \tag{7.6}$$

If the cube root above is an integer, we take

$$m = m^* := \sqrt[3]{\frac{N_f}{2}\left(\frac{1}{t_f/t_c + 3/2}\right)}$$

to be the optimal coarsening factor; otherwise, we take the nearest integer. We note that this analysis can be further extended for more levels, but here we restrict ourselves to the two- and three-level cases.

7.3. MGRIT parallel speedup estimates for MP1 and MP2. Using the formulas derived for two and three levels, we now present the computed MGRIT parallel speedup estimates for MP1 in (2.2) and MP2 in (2.8) in Tables 7.1 and 7.2. First, we compute m^* for $\ell = 2$ and $\ell = 3$ using (7.3) and (7.6), respectively; if the resulting value is not an integer, we round it to the nearest integer and call that m^* instead. We choose $m = m^*$ as the coarsening factor for the MGRIT solve, which gives us N_{it}^* iterations in our numerical experiments; note that this is not the optimal number of MGRIT iterations, but simply the value of N_{it} obtained experimentally when using $m = m^*$. We then use $m = m^*$ and $N_{it} = N_{it}^*$ in the speedup formulas (7.2) and (7.5), where we also use an empirically estimated $\alpha = t_c/t_f$.

Next, in Tables 7.3 and 7.4 we consider a hypothetical *ideal* case in which $t_c/t_f = 1 = \widehat{\alpha}$. Even if $t_c/t_f = 1$ is unlikely, this provides a lower bound for the value of the optimal coarsening factor m^*, since m^* in (7.3) and (7.6) is increasing with respect to t_c/t_f. Following the previously outlined process, we compute the optimal $\widehat{m}^*$, and then obtain $\widehat{N_{it}}^*$ numerically, which are used alongside $\widehat{\alpha} = 1$ for the two- and three-level speedup formulas to provide an estimated upper bound for speedup. We note that, since the fine grid, $t_i = i\Delta t$, $i = 0, 1, \ldots, N_t$, has $N_t + 1$ points, there are $N_f = N_t$ fine-grid intervals. Similarly, $N_c = N_T$.

First, we note that two-level MGRIT is predicted to provide speedup over sequential solves in all of our estimates. For most cases, the estimated speedups for $\ell = 3$ were considerable improvements over the $\ell = 2$ speedup. For all cases, increasing N led to improved speedup. As the sizes N of the Laplacian matrices (2.1a) and (2.1b) increase, more sequential iterations are needed until convergence is achieved, which in turn increases the size of the MGRIT time grids. Optimal speedup formula (7.4) shows that larger grids often benefit from more parallelization. On the other hand, increasing N is also correlated with an increased t_c/t_f ratio, due to the cost of the implicit coarse-grid time-stepping operations increasing relatively more than that of the explicit fine-grid ones. However, in our experiments the larger t_c/t_f ratios were offset by the increase in N_t, such that speedup was improved with larger N.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 7.1
Speedup estimates for MP1, (2.2), for different values of n and $\widehat{\ell}$, using the gradient descent operator (3.4) on the fine grid and the proximal point operator (3.5) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated t_c/t_f values given by $\alpha = 2.9$ for $n = 40$, $\alpha = 3.1$ for $n = 60$, and $\alpha = 3.5$ for $n = 80$; and N_p denotes the number of processors required to achieve optimal speedup.

$n = 40$, $N_t = 8503$

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	112	7	2.73	76
3	27	8	3.29	315

$n = 60$, $N_t = 18\,050$

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	167	8	3.37	109
3	35	8	5.11	516

$n = 80$, $N_t = 30\,018$

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	229	8	4.09	132
3	42	8	6.41	715

TABLE 7.2
Speedup estimates for MP1, (2.2), for different values of n and $\widehat{\ell}$, using the gradient descent operator (3.4) on the fine grid and the proximal point operator (3.5) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on a hypothetical lower bound for t_c/t_f given by $\widehat{\alpha} = 1$; and N_p denotes the number of processors required to achieve optimal speedup.

$n = 40$, $N_t = 8503$

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	65	8	4.07	131
3	22	8	8.33	387

$n = 60$, $N_t = 18\,050$

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	95	8	5.93	190
3	28	8	13.84	645

$n = 80$, $N_t = 30\,018$

ℓ	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	123	8	7.65	245
3	33	8	19.48	910

TABLE 7.3
Speedup estimates for MP2-1D, (2.8), for different values of n and $\widehat{\ell}$, using the proximal gradient operator (3.7) on the fine grid and the alternating proximals operator (3.8) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated t_c/t_f values given by $\alpha = 2.4$ for $n = 256$, and $\alpha = 3.9$ for $n = 512$; and N_p denotes the number of processors required to achieve optimal speedup.

$n = 256$, $N_t = 109\,888$

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	363	10	7.56	303
3	60	9	19.52	1832

$n = 512$, $N_t = 369\,612$

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	849	10	10.88	436
3	100	9	27.12	3697

TABLE 7.4
Speedup estimates for MP2-1D, (2.8), for different values of n and $\widehat{\ell}$, using the proximal gradient operator (3.7) on the fine grid and the alternating proximals operator (3.8) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on a hypothetical lower bound for t_c/t_f given by $\widehat{\alpha} = 1$; and N_p denotes the number of processors required to achieve optimal speedup.

$n = 256$, $N_t = 109\,888$

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	234	10	11.72	470
3	52	9	40.61	2114

$n = 512$, $N_t = 369\,612$

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	430	10	21.49	860
3	77	10	82.62	4801

The results in Table 7.5 are exceptions to the above. For the $n^2 = 64^2$ experiment, the estimated t_f/t_c ratio of $\alpha = 13$ was very large. This is expected, since we are using 2D Laplacians (2.1b) instead of 1D ones (2.1a), thus increasing the relative cost of implicit steps. In fact, α was so large that the two-level solve is predicted to perform better than the three-level solve, likely since using three levels requires an additional post-relaxation step on the $\ell = 2$ coarse grid. However, when increasing the problem size to $n^2 = 128^2$, the t_c/t_f ratio was reduced to $\alpha = 7$. To understand why this happens, recall that the fine-grid proximal gradient operator (3.7) is $\Phi = P_{sg} \circ G_{sf}$, while the coarse-grid alternating proximals operator (3.8) is $\Phi_\Delta = P_{sg} \circ P_{sf}$, with $s = 1/L$ for Φ and $s = m/L$ for Φ_Δ. As N increases, the cost of

ETNA
Kent State University and
Johann Radon Institute (RICAM)

TABLE 7.5
Speedup estimates for MP2-2D, (2.8), for different values of n and $\widehat{\ell}$, using the proximal gradient operator (3.7) on the fine grid and the alternating proximals operator (3.8) on the coarse grid. Here, m^* and S^* are computed based on the numerically estimated t_c/t_f values given by $\alpha = 13$ for $n^2 = 64^2$, and $\alpha = 7$ for $n^2 = 128^2$; and N_p denotes the number of processors required to achieve optimal speedup.

$n^2 = 64^2$, $N_t = 13\,256$

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	294	9	1.25	46
3	46	10	0.67	289

$n^2 = 128^2$, $N_t = 46\,431$

$\widehat{\ell}$	m^*	N_{it}^*	S_ℓ^*	N_p
2	403	9	3.19	116
3	58	10	3.24	801

TABLE 7.6
Speedup estimates for MP2-2D, (2.8), for different values of n and $\widehat{\ell}$, using the proximal gradient operator (3.7) on the fine grid and the alternating proximals operator (3.8) on the coarse grid. Here, $\widehat{m}^*$ and $\widehat{S}^*$ are computed based on a hypothetical lower bound for t_c/t_f given by $\widehat{\alpha} = 1$; and N_p denotes the number of processors required to achieve optimal speedup.

$n^2 = 64^2$, $N_t = 13\,256$

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	81	10	4.07	164
3	25	9	10.07	531

$n^2 = 128^2$, $N_t = 46\,431$

$\widehat{\ell}$	$\widehat{m}^*$	$\widehat{N_{it}}^*$	$\widehat{S}_\ell^*$	N_p
2	152	10	7.61	306
3	39	9	22.87	1191

the P_{sg} step in both Φ and Φ_Δ accounts for a larger share of both t_c and t_f, such that the t_c/t_f ratio is reduced. While still large, t_c/t_f was reduced enough that the three-level solve is estimated to provide a slight speedup improvement over the two-level solve.

In particular, Tables 7.3 and 7.5 exemplify the conditions for which our method performs best and worse, respectively; in the former, we have large N_t and small α, while the opposite is true for the latter. The implicit steps in our MGRIT implementation were calculated using LU-factorization with saved L and U factors to avoid having to repeatedly invert large matrices directly; improving this calculation can bring the t_c/t_f ratio to lower, more beneficial levels, hence bringing speedup estimates closer to the hypothetical $t_c/t_f = 1$ results in Tables 7.2, 7.4, and 7.6. While not explored here, we expect speedups to potentially improve with more levels; this is supported by Tables 6.1, 6.2, and 6.3, in which we observe that N_{it} remains fairly consistent across all possible values ℓ. Finally, increasing the resolution of MP2-2D to $n^2 = 256^2$ or $n^2 = 512^2$ to match the tests for MP2-1D, would dramatically increase N_t, which would also be expected to substantially increase the speedup.

8. Issues with momentum-accelerated algorithms. So far, we have only worked with standalone versions of gradient-based methods such as gradient descent, (3.4), and proximal gradient descent, (3.7). However, in practice they are often replaced by accelerated variants; for example, gradient descent is often replaced by Nesterov's accelerated gradient descent [33]. Starting from $\mathbf{u}_0 \in \mathbb{R}^N$ and letting $0 < s < 2/L$, Nesterov's accelerated gradient sequence $\{\mathbf{u}_k, \mathbf{v}_k\}$, given by

$$\begin{aligned} \mathbf{u}_k &= \mathbf{v}_{k-1} - s\nabla f(\mathbf{v}_{k-1}), \\ \mathbf{v}_k &= \mathbf{u}_k + \beta_k(\mathbf{u}_k - \mathbf{u}_{k-1}), \end{aligned}$$

with momentum parameter $\beta_k = (k-1)/(k+2)$, converges to a minimizer of f with a convergence rate of $\mathcal{O}(1/k^2)$ for convex f with Lipschitz gradient [34]. The connection between optimization method and ODE still holds, although not as trivially as in Section 3.3. In [43], a second-order ODE was derived which is the exact limit of Nesterov's scheme as the

ETNA
Kent State University and
Johann Radon Institute (RICAM)

time step s approaches 0:

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2}\mathbf{u}(t) + \frac{3}{t}\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{u}(t) + \nabla f(\mathbf{u}(t)) = 0, \tag{8.1}$$

for $t > 0$ with initial conditions $\mathbf{u}(0) = \mathbf{u}_0$ and $\mathrm{d}\mathbf{u}(0)/\mathrm{d}t = 0$, where $\mathbf{u}_0$ is the initial guess. The time parameter in this ODE is related to the step size s in Nesterov's scheme via $t \approx k\sqrt{s}$.

Initially, we expected that MGRIT in standard form with implicit discretization on the coarse grid, as explained above, would work well for MP1, (2.2), with Nesterov's accelerated gradient operator (8.1) as the fine-grid operator and a similar momentum-accelerated proximal point operator (3.5) as the coarse-grid operator, given the positive results obtained with gradient descent (3.4). However, in our preliminary work, MGRIT has failed to converge except for when using very small constant values of $\beta = \epsilon > 0$, in which case Nesterov's scheme behaves almost exactly like gradient descent. Unlike gradient descent, which we have observed to behave similarly to a parabolic PDE in Section 5.1, the same does not apply for Nesterov's accelerated gradient descent as explained in the following remark.

REMARK 8.1 (Nesterov's accelerated gradient ODE and hyperbolic PDE connection). Consider Nesterov's accelerated gradient ODE (8.1) applied to MP1, (2.2), such that $\nabla f(\mathbf{u}(t)) = A_1\mathbf{u}$, where A_1 is the 1D Laplacian matrix in (2.1a) with $(A_1\mathbf{u})_i \approx -\partial^2 u(x_i,t)/\partial x^2$. Then, we have

$$\frac{\partial^2}{\partial t^2}u(x,t) + \frac{3}{t}\frac{\partial}{\partial t}u(x,t) - \frac{\partial^2}{\partial x^2}u(x,t) = 0,$$

which is a hyperbolic PDE.

This behavior similar to a hyperbolic PDE, which MGRIT in standard form is known to struggle with, might be a reason for why Nesterov's accelerated gradient did not work with MGRIT using the standard algorithmic approach operators described in Section 4. On the other hand, it has recently been shown that MGRIT can be made to work for hyperbolic PDEs by carefully considering nonstandard specialized coarse operators: research on modified semi-Lagrangian coarse-grid operators for advection-dominated and hyperbolic PDEs [9, 10, 11, 12, 26] has shown progress in adapting MGRIT to solve hyperbolic PDEs, which indicates a promising direction for further research on applying MGRIT to momentum-accelerated optimization algorithms.

9. Conclusions. We have presented a framework for solving optimization problems all at once, in the sense that sequential solves are replaced by a parallel solver in which the sequential iterations are computed simultaneously. Using the MGRIT algorithm to parallelize over iterations of the optimization method instead of over time, we demonstrated a multilevel parallel-in-iteration method that can converge in a small number of iterations, and also provides considerable estimated speedup for both smooth and nonsmooth problems. Further improvements to this framework and the consideration of other model problems can be investigated by the continued exploration of the connection between optimization algorithms and ODEs, especially those related to diffusion-dominated problems for which MGRIT performs well. Among the problems of our interest, we list constrained optimization problems using ADMM (alternating direction method of multipliers), minimizations of nonquadratic functions (such as the cross-entropy loss function from logistic regression), and applications of stochastic approximations such as stochastic gradient descent, to name a few. We established that, in order to extend this framework to momentum-accelerated algorithms such as Nesterov's accelerated gradient descent and FISTA, we will likely need to address MGRIT's difficulties with handling hyperbolic PDEs. Adapting MGRIT with this intent is an intricate problem-specific endeavor, which we leave as a suggestion for future work.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

REFERENCES

[1] A. ANG, H. DE STERCK, AND S. VAVASIS, *MGProx: a nonsmooth multigrid proximal gradient method with adaptive restriction for strongly convex optimization*, SIAM J. Optim., 34 (2024), pp. 2788–2820.
[2] J. D. ANGRIST AND J.-S. PISCHKE, *Mostly Harmless Econometrics: An Empiricist's Companion*, Princeton University Press, Princeton, 2009.
[3] H. ATTOUCH, Z. CHBANI, J. PEYPOUQUET, AND P. REDONT, *Fast convergence of inertial dynamics and algorithms with asymptotic vanishing viscosity*, Math. Program., 168 (2018), pp. 123–175.
[4] H. H. BAUSCHKE AND W. M. MOURSI, *An Introduction to Convexity, Optimization, and Algorithms*, SIAM, Philadelphia, 2024.
[5] A. BECK AND M. TEBOULLE, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM J. Imaging Sci., 2 (2009), pp. 183–202.
[6] A. BRANDT AND C. W. CRYER, *Multigrid algorithms for the solution of linear complementarity problems arising from free boundary problems*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 655–684.
[7] L. A. CAFFARELLI, *The obstacle problem revisited*, J. Fourier Anal. Appl., 4 (1998), pp. 383–402.
[8] A. CHAMBOLLE AND T. POCK, *An introduction to continuous optimization for imaging*, Acta Numer., 25 (2016), pp. 161–319.
[9] H. DE STERCK, R. D. FALGOUT, S. FRIEDHOFF, O. A. KRZYSIK, AND S. P. MACLACHLAN, *Optimizing multigrid reduction-in-time and parareal coarse-grid operators for linear advection*, Numer. Linear Algebra Appl., 28 (2021), Paper No. e2367, 22 pages.
[10] H. DE STERCK, R. D. FALGOUT, AND O. A. KRZYSIK, *Fast multigrid reduction-in-time for advection via modified semi-Lagrangian coarse-grid operators*, SIAM J. Sci. Comput., 45 (2023), pp. A1890–A1916.
[11] H. DE STERCK, R. D. FALGOUT, O. A. KRZYSIK, AND J. B. SCHRODER, *Efficient multigrid reduction-in-time for method-of-lines discretizations of linear advection*, J. Sci. Comput., 96 (2023), Paper No. 1, 31 pages.
[12] H. DE STERCK, S. FRIEDHOFF, A. J. M. HOWSE, AND S. P. MACLACHLAN, *Convergence analysis for parallel-in-time solution of hyperbolic systems*, Numer. Linear Algebra Appl., 27 (2020), Paper No. e2271, 31 pages.
[13] H. DE STERCK, S. FRIEDHOFF, O. A. KRZYSIK, AND S. P. MACLACHLAN, *Multigrid reduction-in-time convergence for advection problems: a Fourier analysis perspective*, Numer. Linear Algebra Appl., 32 (2025), Paper No. e2593, 25 pages.
[14] V. A. DOBREV, T. KOLEV, N. A. PETERSSON, AND J. B. SCHRODER, *Two-level convergence theory for multigrid reduction in time (MGRIT)*, SIAM J. Sci. Comput., 39 (2017), pp. S501–S527.
[15] M. EMMETT AND M. L. MINION, *Toward an efficient parallel in time method for partial differential equations*, Commun. Appl. Math. Comput. Sci., 7 (2012), pp. 105–132.
[16] R. D. FALGOUT, S. FRIEDHOFF, T. V. KOLEV, S. P. MACLACHLAN, AND J. B. SCHRODER, *Parallel time integration with multigrid*, SIAM J. Sci. Comput., 36 (2014), pp. C635–C661.
[17] R. D. FALGOUT, T. A. MANTEUFFEL, B. O'NEILL, AND J. B. SCHRODER, *Multigrid reduction in time for nonlinear parabolic problems: a case study*, SIAM J. Sci. Comput., 39 (2017), pp. S298–S322.
[18] M. P. FRIEDLANDER AND P. TSENG, *Exact regularization of convex programs*, SIAM J. Optim., 18 (2008), pp. 1326–1350.
[19] M. J. GANDER, *50 years of time parallel time integration*, in Multiple Shooting and Time Domain Decomposition Methods, T. Carraro, M. Geiger, S. Körkel, and R. Rannachev, eds. vol. 9 of Contrib. Math. Comput. Sci., Springer, Cham, 2015, pp. 69–113.
[20] M. J. GANDER, F. KWOK, AND J. SALOMON, *ParaOpt: a parareal algorithm for optimality systems*, SIAM J. Sci. Comput., 42 (2020), pp. A2773–A2802.
[21] M. J. GANDER AND S. VANDEWALLE, *Analysis of the parareal time-parallel time-integration method*, SIAM J. Sci. Comput., 29 (2007), pp. 556–578.
[22] B. GAO AND L. PAVEL, *On the properties of the softmax function with application in game theory and reinforcement learning*, Preprint on arXiv, 2017. https://arxiv.org/abs/1704.00805
[23] G. H. GOLUB, P. C. HANSEN, AND D. P. O'LEARY, *Tikhonov regularization and total least squares*, SIAM J. Matrix Anal. Appl., 21 (1999), pp. 185–194.
[24] J. HAHNE, B. POLENZ, I. KULCHYTSKA-RUCHKA, S. FRIEDHOFF, S. ULBRICH, AND S. SCHÖPS, *Parallel-in-time optimization of induction motors*, J. Math. Ind., 13 (2023), Paper No. 6, 16 pages.
[25] A. J. HOWSE, H. DE STERCK, R. D. FALGOUT, S. MACLACHLAN, AND J. SCHRODER, *Parallel-in-time multigrid with adaptive spatial coarsening for the linear advection and inviscid Burgers equations*, SIAM J. Sci. Comput., 41 (2019), pp. A538–A565.
[26] O. A. KRZYSIK, H. DE STERCK, R. D. FALGOUT, AND J. B. SCHRODER, *Parallel-in-time solution of hyperbolic PDE systems via characteristic-variable block preconditioning*, SIAM J. Sci. Comput., in press, pp. S337–S363. https://doi.org/10.1137/24M1673310
[27] J.-L. LIONS, Y. MADAY, AND G. TURINICI, *Résolution d'EDP par un schéma en temps "pararéel"*, C. R. Acad. Sci. Paris Sér. I Math., 332 (2001), pp. 661–668.

ETNA
Kent State University and
Johann Radon Institute (RICAM)

[28] P.-L. LIONS AND B. MERCIER, *Splitting algorithms for the sum of two nonlinear operators*, SIAM J. Numer. Anal., 16 (1979), pp. 964–979.
[29] Y. MADAY, M.-K. RIAHI, AND J. SALOMON, *Parareal in time intermediate targets methods for optimal control problems*, in Control and Optimization with PDE constraints, K. Bredies, C. Clason, K. Kunisch, and G. von Winckel, eds., vol. 164 of Internat. Ser. Numer. Math., Birkhäuser/Springer, Basel, 2013, pp. 79–92.
[30] J. MANDEL, *A multilevel iterative method for symmetric, positive definite linear complementarity problems*, Appl. Math. Optim., 11 (1984), pp. 77–95.
[31] O. L. MANGASARIAN, *Sufficiency of exact penalty minimization*, SIAM J. Control Optim., 23 (1985), pp. 30–37.
[32] M. L. MINION, *A hybrid parareal spectral deferred corrections method*, Commun. Appl. Math. Comput. Sci., 5 (2010), pp. 265–301.
[33] Y. NESTEROV, *A method for solving the convex programming problem with convergence rate* $o(1/k^2)$, Dokl. Akad. Nauk SSSR, 269 (1983), pp. 543–547.
[34] ———, *Introductory Lectures on Convex Optimization*, Springer, New York, 2004.
[35] B. O'DONOGHUE AND E. CANDÈS, *Adaptive restart for accelerated gradient schemes*, Found. Comput. Math., 15 (2015), pp. 715–732.
[36] B. W. ONG AND J. B. SCHRODER, *Applications of time parallelization*, Comput. Vis. Sci., 23 (2020), Paper No. 11, 15 pages.
[37] N. PARIKH AND S. BOYD, *Proximal algorithms*, Found. Trends Optim., 1 (2014), pp. 127–239.
[38] M. RIES, U. TROTTENBERG, AND G. WINTER, *A note on MGR methods*, Linear Algebra Appl., 49 (1983), pp. 1–26.
[39] J.-F. RODRIGUES, *Obstacle Problems in Mathematical Physics*, North-Holland, Amsterdam, 1987.
[40] O. ROMERO, M. BENOSMAN, AND G. J. PAPPAS, *ODE discretization schemes as optimization algorithms*, in 2022 IEEE 61st Conference on Decision and Control (CDC), IEEE Conference Proceedings, Los Alamitos, 2022, pp. 6318–6325.
[41] D. RUPRECHT AND R. KRAUSE, *Explicit parallel-in-time integration of a linear acoustic-advection system*, Comput. & Fluids, 59 (2012), pp. 72–83.
[42] B. S. SOUTHWORTH, *Necessary conditions and tight two-level convergence bounds for parareal and multigrid reduction in time*, SIAM J. Matrix Anal. Appl., 40 (2019), pp. 564–608.
[43] W. SU, S. BOYD, AND E. J. CANDÈS, *A differential equation for modeling Nesterov's accelerated gradient method: theory and insights*, J. Mach. Learn. Res., 17 (2016), Paper No. 153, 43 pages.
[44] J. SUYKENS AND J. VANDEWALLE, *Least squares support vector machine classifiers*, Neural Proc. Lett., 9 (1999), pp. 293–300.
[45] Y. TAI, *A survey of regression algorithms and connections with deep learning*, Preprint on arXiv, 2021. https://arxiv.org/abs/2104.12647
[46] G. TRAN, H. SCHAEFFER, W. M. FELDMAN, AND S. J. OSHER, *An* L^1 *penalty method for general obstacle problems*, SIAM J. Appl. Math., 75 (2015), pp. 1424–1444.
[47] U. TROTTENBERG, C. W. OOSTERLEE, AND A. SCHULLER, *Multigrid*, Academic Press, San Diego, 2001.
[48] S. ULBRICH, *Preconditioners based on "parareal" time-domain decomposition for time-dependent PDE-constrained optimization*, in Multiple Shooting and Time Domain Decomposition Methods, vol. 9 of Contrib. Math. Comput. Sci., Springer, Cham, 2015, pp. 203–232.
[49] R. VUCHKOV, E. C. CYR, A. JAVEED, AND D. RIDZAL, *A parallel-in-time multigrid preconditioner for optimal control*, Preprint on arXiv, 2024. https://arxiv.org/abs/2405.04808
[50] C. WU AND J. W. L. WAN, *Multigrid methods with Newton-Gauss-Seidel smoothing and constraint preserving interpolation for obstacle problems*, Numer. Math. Theory Methods Appl., 8 (2015), pp. 199–219.