○ **Research in**
**the Mathematical Sciences**

Check for
updates

## RESEARCH

# PDE acceleration: a convergence rate analysis and applications to obstacle problems

Jeff Calder[1*] and Anthony Yezzi[2]

*Correspondence:
jcalder@umn.edu
[1]School of Mathematics,
University of Minnesota,
Minneapolis, USA
Full list of author information is
available at the end of the article
**Source code:** https://github.
com/jwcalder/MinimalSurfaces

## Abstract

This paper provides a rigorous convergence rate and complexity analysis for a recently introduced framework, called *PDE acceleration*, for solving problems in the calculus of variations and explores applications to obstacle problems. PDE acceleration grew out of a variational interpretation of momentum methods, such as Nesterov's accelerated gradient method and Polyak's heavy ball method, that views acceleration methods as equations of motion for a generalized Lagrangian action. Its application to convex variational problems yields equations of motion in the form of a damped nonlinear wave equation rather than nonlinear diffusion arising from gradient descent. These *accelerated PDEs* can be efficiently solved with simple explicit finite difference schemes where acceleration is realized by an improvement in the CFL condition from $dt \sim dx^2$ for diffusion equations to $dt \sim dx$ for wave equations. In this paper, we prove a linear convergence rate for PDE acceleration for strongly convex problems, provide a complexity analysis of the discrete scheme, and show how to optimally select the damping parameter for linear problems. We then apply PDE acceleration to solve minimal surface obstacle problems, including double obstacles with forcing, and stochastic homogenization problems with obstacles, obtaining state-of-the-art computational results.

**Keywords:** Accelerated gradient descent, Nesterov acceleration, Primal–dual methods, Minimal surfaces, Stochastic homogenization, Nonlinear wave equations

**Mathematics Subject Classification:** 65M06, 35Q93, 65K10, 49K20

## 1 Introduction

Optimization is one of the most prominent computational problems in science and engineering. For large-scale problems, which are common in machine learning, second-order methods, such as Newton's method, are intractable, and first-order optimization algorithms are the method of choice [9]. One of the oldest first-order algorithms for optimization is gradient descent

$$x_{k+1} = x_k - \alpha \nabla f(x_k). \tag{1}$$

Gradient descent converges reliably for convex problems, and versions of gradient descent (such as stochastic gradient descent) are state of the art in modern large-scale machine learning problems [9].

While gradient descent is a reliable first-order method, for many problems it is very slow to converge. This has led to the development of accelerated versions of gradient descent that incorporate some form of momentum. One example is Polyak's heavy ball method [27]

$$x_{k+1} = x_k - \alpha \nabla f(x_k) + \beta(x_k - x_{k-1}). \tag{2}$$

The term $\beta(x_k - x_{k-1})$ is referred to as momentum and acts to accelerate convergence. Polyak's heavy ball method is simply a discretization of the second-order ODE

$$\ddot{x} + a\dot{x} = -\nabla f(x), \tag{3}$$

which corresponds to the equations of motion for a body in a potential field. This continuum version is also called *heavy ball with friction* and was studied by Attouch, Goudou, and Redont [3] and also by Goudou and Munier [16]. Another example of a momentum descent algorithm is Nesterov's famous accelerated gradient descent [26], one form of which is

$$x_{k+1} = y_k - \alpha \nabla f(y_k), \ \ y_{k+1} = x_{k+1} + \frac{k-1}{k+2}(x_{k+1} - x_k). \tag{4}$$

Nesterov's accelerated gradient descent contains an initial step of gradient descent and then a second momentum step that averages the new update with the previous iterate. Nesterov [26] proved that the method converges (for strongly convex problems) at a rate of $O(1/t^2)$ after $t$ steps, which is optimal for first-order methods.

Many variants of Nesterov acceleration have been proposed over the years, and the methods are very popular in machine learning [35,42], due to both the acceleration in convex problems and the ability to avoid local minima in nonconvex problems. Recent work has begun to shed light on the fundamental nature of acceleration in optimization. Su, Boyd, and Candes [32] showed that Nesterov acceleration is a discretization of the second-order ODE

$$\ddot{x} + \frac{3}{t}\dot{x} = -\nabla f(x). \tag{5}$$

This ODE has been termed *continuous-time Nesterov* [40]. Since the friction coefficient $3/t$ vanishes as $t \to \infty$, many implementations of the algorithm involve *restarting*, whereby time is reset to $t = 0$ whenever the system appears underdamped [40]. Wibisono, Wilson, and Jordan [42] went further, showing that all Nesterov-type accelerated descent methods can be realized as discretizations of equations of motion in a generalized Lagrangian sense. In doing so, they offer a highly insightful and useful variational characterization of accelerated gradient descent.

Following their Lagrangian formulation, Yezzi and Sundaramoorthi [44] developed an accelerated PDE framework for solving active contour models in image segmentation, which are notorious for local minima. In a parallel work, Sundaramoorthi and Yezzi [33] (see also [34]) applied the same ideas to flows of diffeomorphisms, which have applications in computer vision, such as optical flow problems. This *PDE acceleration* framework was further developed by Benyamin, Calder, Sundaramoorthi, and Yezzi [8] in the context of calculus of variations problems defined for functions on $\mathbb{R}^n$, including stability analysis for various explicit and semi-implicit discretization schemes, where they illustrated several examples in image processing such as total variation (TV) and Beltrami regularization. They drew special attention to a general class of regularized optimization problems where the *accelerated PDE* takes the form a damped nonlinear *wave* equation (generalizing (3)

and (5)), and the acceleration is realized as an improvement in the CFL condition from $dt \sim dx^2$ for diffusion equations (or standard gradient descent), to $dt \sim dx$ for wave equations. We also mention that there have been some recent approaches to acceleration in image processing, which involve solving PDEs arising from variational problems [5,17,18,41]. Since these methods are not derived from a variational (Lagrangian) perspective, the methods do not descend on an energy and lack convergence guarantees and convergence rates. Finally, we mention an interesting recent work [29] that considers acceleration-type schemes for nonlinear elliptic equations that *do not* arise from variational formulations. The authors of [29] formulate their schemes to blend together acceleration and gradient descent in such a way that the iterates satisfy a comparison principle, which is then used to prove convergence to steady state. While the application is not variational, the authors observe acceleration similar to the variational setting.

This paper has several contributions. First, we analyze *PDE acceleration* and prove convergence with a linear rate for strongly convex problems. Using the convergence rate, we show that the computational complexity of PDE acceleration is $O(m^{n+1})$ for solving a PDE in dimension $n$ on a grid with $m$ points along each coordinate axis (in other words, $O(N^{(n+1)/n})$ where $N = m^n$ is the number of grid points). This is the same complexity as the conjugate gradient method for linear problems [23]. Second, we provide a linear analysis of PDE acceleration and show how to optimally select the damping coefficient via the solution of an eigenvalue problem. As a toy example, we study the Dirichlet problem and show that PDE acceleration compares favorably to preconditioned conjugate gradient and MINRES methods [23]. In contrast to other indirect methods, the PDE acceleration method is very simple to implement with explicit or semi-implicit Euler discretizations of the wave equation (discussed extensively in [8]) and extends directly to nonlinear problems.

Finally, we apply the PDE acceleration method to efficiently solve minimal surface obstacle problems [12,39,49]. Solving obstacle problems requires resolving a *free boundary*, which makes efficient solutions challenging to obtain. Many algorithms have been proposed for solving classes of obstacle problems; a short list includes penalty methods [11,30,38], splitting and projection algorithms [24,46], free boundary formulations [10,22,25], Lagrange multipliers [19,20], domain decomposition [4], and multigrid methods [21,36]. Of particular interest is a recent primal–dual approach to obstacle problem proposed by Zosso et al. [49], which has some flavor of a momentum-based descent algorithm. The authors of [49] show that their primal–dual approach for obstacle problems is significantly faster than existing approaches. As an independent contribution, we make an improvement to the primal–dual algorithm, allowing it to work for nonlinear minimal surface problems, and we compare the method to PDE acceleration. We find PDE acceleration is approximately $10 \times$ faster in terms of computation time in C code for most experiments, with the difference attributed to the non-explicit dual update in [49]. We also compare against the $L^1$-penalty method of Tran et al. [38], which we find is significantly slower than both primal–dual and PDE acceleration for nonlinear obstacle problems.

We mention that, at the discrete level, PDE acceleration resembles other momentum-based algorithms, such as the heavy ball method or Nesterov acceleration [26,27]. The results in this paper show that there are significant advantages to formulating a general continuum PDE acceleration framework. First, by performing the convergence rate anal-

ysis at the PDE level, we get a mesh-free convergence rate and the number of iterations to converge depends solely on the CFL time step restriction. Second, the parameters in the model—the friction coefficient and time step—can now be chosen optimally from PDE considerations and do not require manual fine-tuning. In particular, the optimal choice for the damping/friction coefficient can be derived from an eigenvalue problem (see Sect. 2.3), while the largest stable time step is determined from the CFL condition [23].

### 1.1 Outline

This paper is organized as follows. In Sect. 2, we review (for the case of functions defined over $\mathbb{R}^n$) and slightly generalize the PDE acceleration framework, prove a linear convergence rate, and analyze the complexity of PDE acceleration. In particular, in Sect. 2.3 we show how to select the damping coefficient optimally for linear problems. In Sect. 3, we study the Dirichlet problem as a toy example and explore connections to primal–dual algorithms. In Sect. 4, we show how to apply PDE acceleration to nonlinear obstacle problems, and describe our improved version of the primal–dual method from [49]. Finally, in Sect. 5 we give results of numerical simulations comparing PDE acceleration to primal–dual and $L^1$-penalty methods for several different obstacle problems, including double obstacle problems with forcing and stochastic homogenization problems with obstacles.

## 2 PDE acceleration framework

We review here the PDE acceleration framework for solving unconstrained problems in the calculus of variations for functions over $\mathbb{R}^n$, as originally presented in [8]. Consider the general unconstrained calculus of variations problem

$$\min_{u \in \mathcal{A}} E[u] := \int_{\Omega} L(x, u, \nabla u)\, \mathrm{d}x, \tag{6}$$

where $\Omega \subset \mathbb{R}^n$ and $\mathcal{A} = g + H_0^1(\Omega)$ or $\mathcal{A} = H^1(\Omega)$. We write $L = L(x, z, p)$ and write $\nabla_x L$, $L_z$, and $\nabla_p L$ for the partial derivatives of $L$ in each variable. There is no loss of generality in considering the unconstrained problem since we will handle constraints (such as obstacles) later with an $L^2$-penalty term (see Sect. 4.1). We define the generalized action integral

$$J[u] = \int_{t_0}^{t_1} k(t)\, (K[u] - b(t)E[u])\ \mathrm{d}t, \tag{7}$$

where $u = u(x, t)$, $k(t)$, $b(t)$ are time-dependent weights and $K[u]$ is the analog of kinetic energy, which we take to be

$$K[u] = \frac{1}{2} \int_{\Omega} \rho(x, u, \nabla u) u_t^2\, \mathrm{d}x, \tag{8}$$

where $\rho : \Omega \times \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}_+$ is a mass density that may depend on $u$ and $\nabla u$. The action $J$ is a Lagrangian action with kinetic energy $K$ and potential energy $E$. We note that the time-dependent weight $k(t)$ is necessary to ensure dissipation of energy (in particular dissipation of the objective $E$; see Lemma 1). Regarding the mass density, often one may take $\rho = \rho(x)$ or $\rho = 1$. The more general setting may be useful, for example, in level set problems like image segmentation, where the object of interest is really the zero level set of $u$, and the kinetic energy of the zero level set can be obtained by selecting $\rho(x, z, p) = \delta(z)/|p|$.

The descent equations for the PDE acceleration method are exactly the Euler–Lagrange equations for $J$, i.e., the equations of motion, which we derive now. Let us write

$$\nabla E[u] := L_z(x, u, \nabla u) - \operatorname{div}\left(\nabla_p L(x, u, \nabla u)\right) \tag{9}$$

and

$$\nabla K[u] := \frac{1}{2}u_t^2 \rho_z(x, u, \nabla u) - \frac{1}{2}\operatorname{div}\left(u_t^2 \nabla_p \rho(x, u, \nabla u)\right) \tag{10}$$

for the Euler–Lagrange equation for $E$ and for $K$. We recall that $\nabla E$ can be interpreted as the gradient of $E$ in the sense that

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\Big|_{\varepsilon=0} E[u + \varepsilon v] = \int_{\Omega} \nabla E[u]\, v \,\mathrm{d}x \tag{11}$$

for all $v$ smooth with compact support in $\Omega$. Using this identity, a variation on $J$ yields

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}\Big|_{\varepsilon=0} J[u + \varepsilon v] = \int_{t_0}^{t_1} \int_{\Omega} k(t)\rho u_t v_t + k(t)\nabla K[u]\, v - k(t)b(t)\nabla E[u]\, v \,\mathrm{d}x,$$

$$= \int_{t_0}^{t_1} \int_{\Omega} \left(-\frac{\mathrm{d}}{\mathrm{d}t}\left(k(t)\rho u_t\right) + k(t)\nabla K[u] - k(t)b(t)\nabla E[u]\right) v \,\mathrm{d}x,$$

for $v \in C_c^{\infty}(\Omega \times (t_0, t_1))$. Therefore, the equations of motion are

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\rho u_t\right) + a(t)\rho u_t = \nabla K[u] - b(t)\nabla E[u], \tag{12}$$

where $a(t) = k'(t)/k(t)$. The nonlinear wave equation (12) is the descent equation for PDE acceleration, and the minimizer of $E$ is obtained by solving the equation for some initial conditions $u(x, 0)$ and $u_t(x, 0)$, and sending $t \to \infty$. The boundary condition depends on the choice of $\mathcal{A}$—we have either $u = g$ or $\nabla_p L \cdot \mathbf{n} = 0$ on $\partial\Omega$, where $\mathbf{n}$ is the unit outward normal to $\partial\Omega$. Of course, we can also consider problems with mixed boundary conditions.

Notice in the equations of motion (12) the gradient $-\nabla E$ is now a forcing term in a damped wave equation, so it contributes to a change in velocity at each time step. The reader should contrast this with gradient descent

$$u_t = -\nabla E[u],$$

where the gradient is *exactly* the velocity term, which can change *instantaneously*.

In special cases, we can recover continuum versions of Polyak's heavy ball method and Nesterov acceleration. For example, if we take $\rho = 1$ and $a(t)$ and $b(t)$ to be constants, we get the PDE continuum version of the heavy ball with friction (3), and for $a(t) = 3/t$, $\rho = 1$, and $b(t) = 1$, we get the continuum version of Nesterov acceleration.

## 2.1 Convergence rate

We prove in this section a convergence rate for the solution $u$ of the equations of motion (12) to the steady-state solution $u^*$ of

$$\nabla E[u^*] = 0 \quad \text{in } \Omega, \tag{13}$$

subject to the Dirichlet condition $u^* = g$ or the Neumann-type condition $\nabla_p L \cdot \mathbf{n} = 0$ on $\partial\Omega$. We assume throughout this section that $\Omega$ is open and bounded with Lipschitz boundary $\partial\Omega$.

We first establish monotonicity of total energy.

**Lemma 1** (Energy monotonicity) *Assume* $a(t), b(t) \geq 0$ *and let u satisfy* (12). *Suppose either* $u(x, t) = g(x)$ *or* $\nabla_p L(x, u, \nabla u) \cdot \mathbf{n} = 0 = \nabla_p \rho(x, u, \nabla u) \cdot \mathbf{n}$ *on* $\partial\Omega$. *Then*

$$\frac{\mathrm{d}}{\mathrm{d}t}(K[u] + b(t)E[u]) = -2a(t)K[u] + b'(t)E[u]. \tag{14}$$

*Proof*  First note that

$$\frac{\mathrm{d}}{\mathrm{d}t}E[u] = \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\Big|_{\varepsilon=0} E[u + \varepsilon u_t] = \int_\Omega \nabla E[u]u_t\,\mathrm{d}x + \int_{\partial\Omega} u_t \nabla_p L(x, u, \nabla u) \cdot \mathbf{n}\,\mathrm{d}S.$$

Due to the boundary condition, either $u_t = 0$ or $\nabla_p L(x, u, \nabla u) \cdot \mathbf{n} = 0$ on $\partial\Omega$. Therefore

$$\frac{\mathrm{d}}{\mathrm{d}t}E[u] = \int_\Omega \nabla E[u]u_t\,\mathrm{d}x. \tag{15}$$

Similarly, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}K[u] = \int_\Omega \rho u_t u_{tt} + \nabla K[u]u_t\,\mathrm{d}x. \tag{16}$$

Using the equations of motion (12), we also have

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}K[u] &= \frac{1}{2}\int_\Omega \rho u_t u_{tt} + u_t \frac{\mathrm{d}}{\mathrm{d}t}(\rho u_t)\,\mathrm{d}x \\
&= \frac{1}{2}\int_\Omega \rho u_t u_{tt} + \nabla K[u]u_t\,\mathrm{d}x - \frac{1}{2}\int_\Omega b(t)\nabla E[u]u_t + a(t)\rho u_t^2\,\mathrm{d}x \\
&= \frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}t}K[u] - \frac{1}{2}b(t)\frac{\mathrm{d}}{\mathrm{d}t}E[u] - a(t)K[u],
\end{aligned}$$

where we used (15) and (16) in the last line. It follows that

$$\frac{\mathrm{d}}{\mathrm{d}t}K[u] + b(t)\frac{\mathrm{d}}{\mathrm{d}t}E[u] = -2a(t)K[u].$$

The proof is completed by adding $b'(t)E[u]$ to both sides.                                        □

*Remark 1*  If $b'(t) \leq 0$ and $E[u] \geq 0$, then the total energy $K[u] + b(t)E[u]$ is monotonically decreasing at a rate controlled by the damping coefficient $a(t)$. In particular, we have

$$\frac{\mathrm{d}}{\mathrm{d}t}(K[u] + b(t)E[u]) \leq -2a(t)K[u]. \tag{17}$$

We now prove a linear convergence rate in the special case that $\rho$, $a$, and $b$ are constants, $E$ has the form

$$E[u] = \int_\Omega \Phi(x, \nabla u) + \Psi(x, u)\,\mathrm{d}x, \tag{18}$$

and $u$ and $u^*$ satisfy the Dirichlet condition $u = g = u^*$ on $\partial\Omega$. We assume that $\Phi, \Psi \in C^2$, $\Phi = \Phi(x, p)$ is convex in $p$, $\Psi = \Psi(x, z)$ is convex in $z$, and for all $p \in \mathbb{R}^n$, $x \in \Omega$, and $z \in \mathbb{R}$

$$\theta I \leq \nabla_p^2 \Phi(x, p) \leq \theta^{-1} I \tag{19}$$

and

$$\Psi_{zz}(x, z) \leq \mu \tag{20}$$

for some $\theta, \mu > 0$.

*Remark 2* If there exists $M > 0$ such that $|\nabla u^*| \leq M$, where $u^*$ solves (13), then we can relax (19) to the condition that $\nabla_p^2 \Phi > 0$ (that is, $\Phi$ is strictly convex in $p$). Indeed, define

$$\overline{\Phi}(x, p) := \Phi(x, p)\varphi(p) + K \max\{|p| - M, 0\}^2, \tag{21}$$

where $K > 0$ and $\varphi \in C^\infty$ is a bump function with $0 \leq \varphi \leq 1$, $\varphi(p) = 1$ for $|p| \leq 2M$, and $\varphi(p) = 0$ for $|p| \geq 4M$. Clearly, $\overline{\Phi}(x, p)$ satisfies (19) for some $\theta > 0$ when $|p| \geq 4M$, and by choosing $K$ large enough, the strict convexity of $\Phi$ and compactness of $\overline{\Omega} \times B(0, 2M)$ allow us to extend condition (19) to all $(x, p) \in \Omega \times \mathbb{R}^n$. Now let $\overline{E}$ be the energy in (18) with $\overline{\Phi}$ in place of $\Phi$. Since $|\nabla u^*| \leq M$ and $\overline{\Phi}(x, p) = \Phi(x, p)$ for all $|p| \leq M$, we see that $\nabla \overline{E}(u^*) = 0$; that is, $u^*$ is the unique solution of $\nabla \overline{E} = 0$ as well. We can use $\overline{E}$ in place of $E$ in PDE acceleration to ensure (19) holds while obtaining the same steady-state solution (though the dynamics can be different). This is mainly a theoretical concern, and not something we do in practical applications.

We note that for minimal surface obstacle problems, the gradients of solutions are Hölder continuous (e.g., $u^* \in C^{1,\alpha}$) [12], and hence, we can always find such an $M$.

From now on, we write $\nabla^2$ in place of $\nabla_p^2$.

We now prove the following linear convergence rate.

**Theorem 1** (Convergence rate) *Let $u$ satisfy (12). Assume (18), (19), and (20) hold, $u = u^*$ on $\partial\Omega$, $a(t) = a > 0$ is constant and $b(t) \equiv 1$ and $\rho \equiv 1$. Then, there exists $C > 0$ depending on $a, \theta, u(x, 0)$, and $u_t(x, 0)$ such that*

$$\|u - u^*\|_{H^1(\Omega)}^2 \leq C \exp\left(-\beta t\right), \tag{22}$$

*where*

$$\beta = \frac{a\sqrt{c^2 + 4\lambda\theta} - ac}{2\sqrt{\lambda\theta} + a} \quad \text{with } c = a + \frac{\mu}{a} + \frac{2\lambda}{a}(\theta^{-1} - \theta), \tag{23}$$

*and $\lambda > 0$ is the Poincaré constant for $\Omega$.*

*Proof* We use energy methods. Since $u$ solves (12), we have

$$u_{tt} + au_t + \nabla E[u] - \nabla E[u^*] = 0.$$

Multiply both sides by $w := u - u^*$ and integrate over $\Omega$ to find

$$\int_\Omega w_{tt}w + aw_tw + (\nabla E[u] - \nabla E[u^*])(u - u^*)\,\mathrm{d}x = 0.$$

Integrating by parts, we have

$$\int_\Omega (\nabla E[u] - \nabla E[u^*])(u - u^*)\,\mathrm{d}x$$
$$= \int_\Omega (\nabla\Phi(\nabla u) - \nabla\Phi(\nabla u^*)) \cdot (\nabla u - \nabla u^*) + (\Psi_z(x, u) - \Psi_z(x, u^*))(u - u^*)\,\mathrm{d}x.$$

Since $\Psi$ is convex in $z$ and $\Phi$ is strongly convex (by (19)) we deduce

$$\int_\Omega (\nabla E[u] - \nabla E[u^*])(u - u^*)\,\mathrm{d}x \geq \theta \int_\Omega |\nabla w|^2\,\mathrm{d}x.$$

It follows that

$$\int_\Omega w_{tt}w + aw_tw + \theta|\nabla w|^2\,\mathrm{d}x \leq 0,$$

and so

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega \frac{1}{2} a^2 w^2 + aww_t \, \mathrm{d}x \leq \int_\Omega aw_t^2 - a\theta |\nabla w|^2 \, \mathrm{d}x = 2aK[w] - a\theta \int_\Omega |\nabla w|^2 \, \mathrm{d}x. \tag{24}$$

This leads us to define the energy

$$e(t) := \int_\Omega \frac{1}{2} a^2 w^2 + aww_t \, \mathrm{d}x + 2(K[u] + E[u] - E[u^*]). \tag{25}$$

Note first that

$$e(t) = \int_\Omega \frac{1}{2}(aw + w_t)^2 + \frac{1}{2} w_t^2 \, \mathrm{d}x + 2(E[u] - E[u^*]) \geq 0,$$

so $e$ is a valid energy. Using Lemma 1 and (24), we have

$$\dot{e}(t) \leq 2aK[w] - 4aK[w] - a\theta \int_\Omega |\nabla w|^2 \, \mathrm{d}x = -a \int_\Omega \theta |\nabla w|^2 + w_t^2 \mathrm{d}x. \tag{26}$$

Now, we compute

$$
\begin{aligned}
E[u] - E[u^*] &= \int_0^1 \frac{\mathrm{d}}{\mathrm{d}t} E[u^* + t(u - u^*)] \, \mathrm{d}t \\
&= \int_0^1 \int_0^s \frac{d^2}{\mathrm{d}t^2} E[u^* + t(u - u^*)] \, \mathrm{d}t \, \mathrm{d}s \\
&= \int_0^1 \int_0^s \int_\Omega \sum_{i,j=1}^n \Phi_{p_i p_j}(\nabla u^* + t(\nabla u - \nabla u^*))(u_{x_i} - u^*_{x_i})(u_{x_j} - u^*_{x_j}) \\
&\quad + \Psi_{zz}(x, u^* + t(u - u^*))(u - u^*)^2 \, \mathrm{d}x \, \mathrm{d}t \, \mathrm{d}s \\
&\leq \int_0^1 \int_0^s \int_\Omega \theta^{-1}|\nabla w|^2 + \mu w^2 \, \mathrm{d}x \, \mathrm{d}t \, \mathrm{d}s \\
&= \frac{1}{2} \int_\Omega \theta^{-1}|\nabla w|^2 + \mu w^2 \, \mathrm{d}x,
\end{aligned}
$$

where the last inequality used (19). Combining this with the Poincaré inequality $\lambda \int_\Omega w^2 \, \mathrm{d}x \leq \int_\Omega |\nabla w|^2 \, \mathrm{d}x$, we have

$$
\begin{aligned}
e(t) &\leq \frac{1}{2} \int_\Omega (a^2 + a\varepsilon + \mu)w^2 + a\varepsilon^{-1} w_t^2 \, \mathrm{d}x + 2(K[u] + E[u] - E[u^*]) \\
&\leq \int_\Omega \left[\tfrac{1}{2}(a^2 + a\varepsilon + \mu)\lambda^{-1} + \theta^{-1}\right] |\nabla w|^2 + \frac{1}{2}\left(2 + a\varepsilon^{-1}\right) w_t^2 \, \mathrm{d}x.
\end{aligned}
$$

Now, there is a unique value of $\varepsilon > 0$ such that

$$\tfrac{1}{2}(a^2 + a\varepsilon + \mu)\lambda^{-1} + \theta^{-1} = \frac{\theta}{2}(2 + a\varepsilon^{-1}). \tag{27}$$

Selecting this value for $\varepsilon$, we have by (26) that

$$e(t) \leq \frac{1}{2}(2 + a\varepsilon^{-1}) \int_\Omega \theta |\nabla w|^2 + w_t^2 \, \mathrm{d}x \leq -\left(\frac{1}{a} + \frac{1}{2\varepsilon}\right) \dot{e}(t).$$

Therefore,

$$\dot{e}(t) \leq \frac{-2a\varepsilon}{2\varepsilon + a} e(t),$$

from which it follows that

$$e(t) \leq e(0) \exp\left(\frac{-2a\varepsilon}{2\varepsilon + a} t\right). \tag{28}$$

All that remains is to compute $\varepsilon$. Note that (27) is equivalent to

$$\varepsilon^2 + \left(a + \frac{\mu}{a} + \frac{2\lambda}{a}(\theta^{-1} - \theta)\right)\varepsilon - \theta\lambda = 0,$$

and hence

$$2\varepsilon = \sqrt{c^2 + 4\lambda\theta} - c \leq 2\sqrt{\lambda\theta},$$

where

$$c = a + \frac{\mu}{a} + \frac{2\lambda}{a}(\theta^{-1} - \theta).$$

$\square$

*Remark 3* The energy estimates obtained in the proof of Theorem 1 establish uniqueness of solutions to (12) with values in $H^1(\Omega)$ and Dirichlet boundary conditions. We expect that existence of solutions follows from combining these energy estimates with a standard Galerkin approximation [15]. Such a result is outside the scope of this paper, and we leave it to future work.

*Remark 4* Instead of formulating PDE acceleration at the continuum level and then discretizing the descent equations to compute the solution, it is possible to formulate PDE acceleration entirely in the discrete-space setting, by starting with a discretization of $E[u]$. Then, the resulting descent equations become a discretization of (12) that is discrete in space and continuous in time, provided the discrete divergence is defined as the exact numerical adjoint of the discrete gradient (e.g., forward differences for the gradient and backward differences for the divergence, as we use in Sect. 5). We expect the proof of Theorem 1 to extend, with minor modifications, in this situation.

## 2.2 Computational complexity

Theorem 1 allows us to analyze the computational complexity of PDE acceleration. Suppose our domain is the unit box $[0, 1]^2$ and we discretize the problem on an $m \times m$ grid with uniform spacing $dx = 1/m$. Since we can discretize the wave equation (12) with a time step $dt = O(dx) = O(1/m)$ while satisfying the CFL condition, the number of iterations required to converge to within a tolerance of $\varepsilon > 0$ in the $H^1$ norm satisfies

$$k \leq c\beta^{-1} m \log(C\varepsilon^{-1}). \tag{29}$$

Using an explicit time stepping scheme, each iteration has complexity $O(m^2)$; hence, the computational complexity for 2D problems is

$$\text{2D complexity} = O(\beta^{-1} m^3 \log(\varepsilon^{-1})). \tag{30}$$

A similar computation for 3D problems on a $m \times m \times m$ grid yields a complexity of

$$\text{3D complexity} = O(\beta^{-1} m^4 \log(\varepsilon^{-1})). \tag{31}$$

If we write the complexity in terms of the number of unknowns, which is $N = m^2$ in 2D and $N = m^3$ in 3D, then the complexity is $O(N^{3/2})$ for 2D problems and $O(N^{4/3})$

for 3D problems. For linear problems, this complexity is similar to conjugate gradient methods [23].

We remark that the discussion above does not consider numerical discretization errors that can accumulate over iterations, and should be viewed as a heuristic to motivate the convergence rate results we present in Sect. 5. However, we can easily do a rough analysis of the discretization errors. If we discretize the wave equation with a first order in time and second order in space discretization, then the numerical errors will accumulate to $O(k\mathrm{d}x^3)$ after $k$ iterations, provided the solution is smooth. Combining this with (22) from Theorem 1, we have

$$\|u_{\mathrm{d}x,\mathrm{d}t}^k - u^*\|_{L^2(\Omega)}^2 \leq C\left(\exp\left(-\beta k\mathrm{d}t\right) + k^2\mathrm{d}x^6\right),$$

where $u_{\mathrm{d}x,\mathrm{d}t}^k$ denotes the numerical solution on a grid with spacing $(\mathrm{d}x, \mathrm{d}t)$ after $k$ steps, extended to $\Omega$ in any natural way (e.g., say, piecewise constant). Setting $\exp\left(-\beta k\mathrm{d}t\right) = \mathrm{d}x^6$ and using the stability condition $\mathrm{d}t = c\,\mathrm{d}x$, we find that

$$\|u_{\mathrm{d}x,\mathrm{d}t}^k - u^*\|_{L^2(\Omega)} \leq C\beta^{-1}\mathrm{d}x^2\log(m), \tag{32}$$

after $k = O(\beta^{-1}m\log(m))$ iterations, where we recall $m = 1/\mathrm{d}x$. We can compare this to (29), which arrives at a similar estimate for the number of iterations $k$ without considering discretization errors. We also point out that (32) is a convergence rate to the true solution $u^*$ of the continuum problem, and not a convergence rate to the solution of a discretized energy as one would normally see in convergence rates for discrete optimization problems.

## 2.3 Optimal damping for linear problems

While Theorem 1 provides a convergence rate for the PDE acceleration method, it does not indicate how to optimally select the damping parameter $a$ to achieve the optimal rate. Here, we consider the selection of $a$ for linear problems. For nonlinear problems, we propose to linearize and apply the analysis described below.

Let $L$ be a uniformly elliptic second-order partial differential operator in divergence form, that is,

$$Lu = -\sum_{i,j=1}^{n}(a^{ij}u_{x_i})_{x_j}, \tag{33}$$

where $A(x) := (a^{ij}(x))_{ij} \in C^\infty(\Omega)$, and there exists $\theta > 0$ such that $A(x) \geq \theta I$ for all $x$. We consider the PDE acceleration method for solving the Dirichlet problem

$$\begin{cases} Lu^* + cu^* = f & \text{in } \Omega, \\ \qquad\quad u^* = g & \text{on } \partial\Omega, \end{cases} \tag{34}$$

where $c \geq 0$. The equations of motion with constant $a(t) = a$ and $b(t) = b$ are

$$\begin{cases} u_{tt} + au_t + bLu + bcu = f & \text{in } \Omega \times (0,\infty), \\ \qquad\qquad\qquad\qquad u = g & \text{on } \partial\Omega \times (0,\infty), \\ \qquad\qquad\qquad\qquad u = u_0 & \text{on } \Omega \times \{t = 0\}. \end{cases} \tag{35}$$

Let $w(x,t) = u(x,t) - u^*(x)$. Then, $w$ satisfies

$$\begin{cases} w_{tt} + aw_t + bLw + bcw = 0 & \text{in } \Omega \times (0,\infty), \\ \qquad\qquad\qquad\qquad w = 0 & \text{on } \partial\Omega \times (0,\infty), \\ \qquad\qquad\qquad\qquad w = u_0 & \text{on } \Omega \times \{t = 0\}. \end{cases} \tag{36}$$

We can expand the solution $w$ in a Fourier series

$$w(x, t) = \sum_{k=1}^{\infty} d_k(t) v_k, \tag{37}$$

where $v_1, v_2, \ldots$ is an orthonormal basis for $L^2(\Omega)$ consisting of Dirichlet eigenfunctions of $L$ with corresponding eigenvalues

$$0 < \lambda_1 \le \lambda_2 \le \lambda_3 \le \cdots.$$

That is, the function $v_k(x)$ satisfies

$$\begin{cases} Lv_k = \lambda_k v_k & \text{in } \Omega \\ v_k = 0 & \text{on } \partial\Omega. \end{cases} \tag{38}$$

Substituting (37) into (36), we find that

$$d_k''(t) + a d_k'(t) + b(\lambda_k + c) d_k(t) = 0.$$

The general solution is

$$d_k(t) = A e^{r_{k,1} t} + B e^{r_{k,2} t},$$

where

$$r_{k,1} = -\frac{a}{2} + \frac{1}{2}\sqrt{a^2 - 4b(\lambda_k + c)} \quad \text{and} \quad r_{k,2} = -\frac{a}{2} - \frac{1}{2}\sqrt{a^2 - 4b(\lambda_k + c)}.$$

Hence, the optimal decay rate is of the form $e^{-at/2}$ provided that

$$a^2 - 4b(\lambda_k + c) \le 0 \quad \text{for all } k \ge 1.$$

This leads to the optimal choice for the damping coefficient

$$a = 2\sqrt{b(\lambda_1 + c)}. \tag{39}$$

With this choice of damping, we have the convergence rate

$$|u(x, t) - u^*(x)| \le C \exp\left(-\sqrt{b(\lambda_1 + c)}\, t\right), \tag{40}$$

for some constant $C > 0$ depending on the initial condition $u_0$.

We note that when $L$ is degenerate, so $\lambda_1 = 0$, the convergence rate is

$$|u(x, t) - u^*(x)| \le C \exp\left(-\sqrt{bc}\, t\right), \tag{41}$$

with the optimal choice $a = 2\sqrt{bc}$. In particular, if $c = 0$ then the method does not converge, since there are undamped Fourier modes.

## 3 Dirichlet problem

As an illustrative example, we consider the Dirichlet problem

$$\min\left\{ \frac{1}{2} \int_{\Omega} |\nabla u|^2 \, dx \ : \ u \in H^1(\Omega) \text{ and } u = g \text{ on } \partial\Omega \right\}. \tag{42}$$

Gradient descent corresponds to solving the heat equation

$$\begin{cases} u_t - \Delta u = 0 & \text{in } \Omega \times (0, \infty), \\ \quad\quad u = g & \text{on } \partial\Omega \times (0, \infty), \\ \quad\quad u = u_0 & \text{on } \Omega \times \{t = 0\}, \end{cases} \tag{43}$$

while PDE acceleration corresponds to solving the damped wave equation

$$\begin{cases} u_{tt} + au_t - b\Delta u = 0 & \text{in } \Omega \times (0, \infty), \\ \qquad\qquad u = g & \text{on } \partial\Omega \times (0, \infty), \\ \qquad\qquad u = u_0 & \text{on } \Omega \times \{t = 0\}. \end{cases} \qquad (44)$$

For concreteness, we consider the domain $\Omega := [0, 1]^2$. Here, the first Dirichlet eigenvalue is $\lambda_1 = \pi^2$, and hence the optimal choice of the damping coefficient from (39) is

$$a = 2\pi\sqrt{b}.$$

With this choice of $a$, the accelerated PDE method converges to the solution of the Dirichlet problem (42) at a rate of $\exp(-b\pi t)$. There are no parameters to select in the heat equation (43). It is possible to show with a Fourier expansion that the solution of the heat equation (43) converges to the solution of the Dirichlet problem (42) at a rate of $\exp(-\pi^2 t)$. So far there is not much difference between the two equations—part of the difference comes from numerical stiffness, as explained below.

### 3.1 Runtime comparisons

To solve both equations, we use the standard discretizations

$$u_t \approx \frac{u_{ij}^{n+1} - u_{ij}^n}{\mathrm{d}t}, \; u_{tt} \approx \frac{u_{ij}^{n+1} - 2u_{ij}^n + u_{ij}^{n-1}}{\mathrm{d}t^2}, \; \text{and}$$

$$\Delta u \approx \frac{u_{i+1,j}^n + u_{i-1,j}^n + u_{i,j+1}^n + u_{i,j-1}^n - 4u_{ij}^n}{\mathrm{d}x^2},$$

and explicit forward time stepping. The CFL condition for the damped wave equation is

$$\mathrm{d}t \leq \frac{\mathrm{d}x}{\sqrt{2b}}.$$

By (40), the error decays like $\exp(-2\pi\sqrt{b}t)$. Therefore, to solve the problem to within a tolerance of $\varepsilon$ we need $k$ iterations, where $k$ satisfies

$$\varepsilon = C \exp\left(-2\pi\sqrt{b}k\mathrm{d}t\right) = C \exp\left(-\sqrt{2}\pi k\mathrm{d}x\right).$$

Hence, we need

$$k = \frac{1}{\sqrt{2}\pi\,\mathrm{d}x}\log(C\varepsilon^{-1})$$

iterations. Note this is independent of $b$. Additionally, if we saturate the CFL condition and set $\mathrm{d}t = \mathrm{d}x/\sqrt{2b}$, then $b$ does not even appear in the numerical discretization of (44). We contrast this with the heat equation (43), where the CFL condition is $\mathrm{d}t \leq \mathrm{d}x^2/4$. Here, we need

$$k = \frac{4}{\pi^2\mathrm{d}x^2}\log(C\varepsilon^{-1})$$

iterations for convergence. Table 1 shows a comparison of the performance of PDE acceleration, gradient descent, and the primal–dual algorithm from [49] for solving the Dirichlet problem on various grid sizes. We used the boundary condition $g(x_1, x_2) = \sin(2\pi x_1^2) + \sin(2\pi x_2^2)$ and ran each algorithm until the finite difference scheme was satisfied with an error of less than $\mathrm{d}x^2$. The initial conditions for both algorithms were $u(x, 0) = g(x)$. For the primal–dual algorithm [49], we set $r_1 = 4\pi^2 r_2$, which is provably optimal using similar methods as in Sect. 2.3. We see that PDE acceleration is more

**Table 1** Comparison of PDE acceleration, primal–dual, and gradient descent for solving the Dirichlet problem

| Mesh | Our method | | Primal–dual [49] | | Gradient descent | |
|---|---|---|---|---|---|---|
| | Time (s) | Iterations | Time (s) | Iterations | Time (s) | Iterations |
| $64^2$ | 0.012 | 399 | 0.02 | 592 | 0.148 | 8404 |
| $128^2$ | 0.05 | 869 | 0.11 | 1384 | 2.4 | 3872 |
| $256^2$ | 0.38 | 1898 | 1.0 | 3027 | 40 | 174,569 |
| $512^2$ | 4.8 | 4114 | 13.1 | 6831 | 1032 | 774,606 |
| $1024^2$ | 41 | 8813 | 115 | 14,674 | 23,391 | 3,399,275 |

Runtimes are for C code

than twice as fast as primal–dual, while both significantly outperform standard gradient descent. We mention that our method converges to engineering precision very quickly (in about one fifth of the iteration count displayed in Table 1), while the majority of iterations are taken to resolve the solution up to the $O(\mathrm{d}x^2)$ error (for the $1024 \times 1024$ grid, the error tolerance is $\mathrm{d}x^2 \approx 10^{-6}$).

Of course, we do not recommend using PDE acceleration or primal–dual methods for solving linear Poisson problems. In the linear setting, there are faster algorithms available. For comparison, we show in Table 2 the runtimes for incomplete Cholesky preconditioned conjugate gradient and MINRES, Gauss–Seidel with successive overrelaxation, Matlab backslash, and the multigrid method with V-cycles. We see that PDE acceleration is comparable to preconditioned MINRES and conjugate gradient, while Matlab backslash (Cholesky factorization and triangular solve) and multigrid are significantly faster. We did not compare against FFT methods since they are specific to constant coefficient linear problems, which is rather restrictive, and would be comparable to multigrid and backslash.

We mention that for general linear PDE, multigrid is normally much faster than Matlab backslash. The 2D Poisson equation is a special case where the linear system has a simple banded structure and direct solvers are highly efficient and comparable to multigrid. Moving to 3D problems, one would expect multigrid to outperform backslash. Furthermore, our implementation of multigrid with V-cycles may not be optimal and further improvements could be possible. We emphasize that the Dirichlet problem is simply a toy illustrative example of PDE acceleration compared to gradient descent, and it is outside the scope of this paper to provide a thorough comparison to all linear solvers (e.g., other preconditioners, different multigrid cycling, etc.). Our real interest is in nonlinear problems with constraints. The linear methods that we compared against here do not extend directly to nonlinear problems, much less to obstacle constrained problems. The PDE acceleration method is formulated in the general nonlinear case and is provably convergent with the same rate for nonlinear problems. In practice, we usually see similar computation times for nonlinear problems (see Sect. 5).
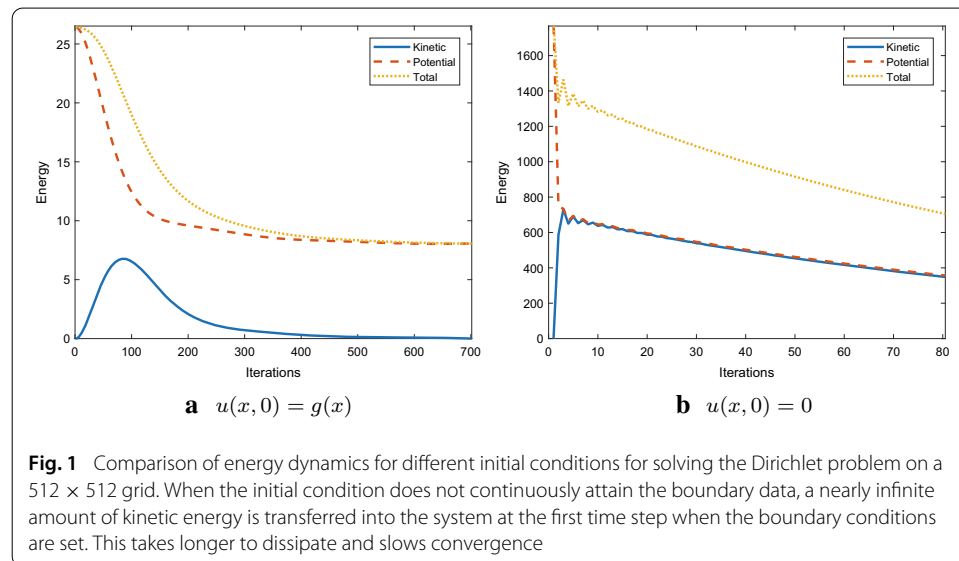
### 3.2 Initial condition

We mention that the choice of initial condition can affect the computation time. If the initial condition does not continuously attain the boundary data, then fixing the boundary data on the first time step transfers a large amount of kinetic energy into the system that takes longer to dissipate. See Fig. 1 for a depiction of the kinetic, potential, and total energy for initial conditions $u(x, 0) = g(x)$ and $u(x, 0) = 0$. The rate of convergence is not affected; it is just the constant in front, which corresponds to the initial energy, that is

**Table 2   Runtimes in seconds for incomplete Cholesky preconditioned conjugate gradient, MINRES, Gauss–Seidel with successive overrelaxation, Matlab backslash, and V-cycle multigrid for solving the Dirichlet problem**

| Mesh | PCG | | MINRES | | Gauss–Seidel | | Backlash | Multigrid |
|---|---|---|---|---|---|---|---|---|
| | Time | Iter. | Time | Iter. | Time | Iter. | Time | Time |
| $64^2$ | 0.027 | 56 | 0.028 | 54 | 0.017 | 197 | 0.013 | 0.036 |
| $128^2$ | 0.08 | 120 | 0.093 | 114 | 0.08 | 432 | 0.037 | 0.048 |
| $256^2$ | 0.67 | 251 | 0.78 | 240 | 0.72 | 1020 | 0.12 | 0.13 |
| $512^2$ | 7.9 | 523 | 9.4 | 500 | 6.1 | 2046 | 0.61 | 0.55 |
| $1024^2$ | 69 | 1089 | 81 | 1044 | 50 | 4100 | 3 | 2.6 |

The Gauss–Seidel method was implemented in C, while the other algorithms were implemented in Matlab



**Fig. 1**   Comparison of energy dynamics for different initial conditions for solving the Dirichlet problem on a $512 \times 512$ grid. When the initial condition does not continuously attain the boundary data, a nearly infinite amount of kinetic energy is transferred into the system at the first time step when the boundary conditions are set. This takes longer to dissipate and slows convergence

larger in this case. For example, in the simulation above, if we start from $u(x, 0) = 0$ on the $512 \times 512$ grid, the computation takes 5529 iterations, or about roughly $1.4 \times$ more iterations compared to the initial condition $u(x, 0) = g(x)$.

This minor issue can be easily fixed in one of two ways. First, we can, if possible, choose an initial condition that continuously attains the boundary data. A second solution is to start from any arbitrary initial condition and then change the boundary conditions gradually, instead of instantaneously. This can be done by gradient descent on the energy

$$I(u) = \frac{1}{2} \int_{\partial \Omega} (u - g)^2 \, \mathrm{d}S.$$

That is, on the boundary we solve the ordinary differential equation

$$u_t = g - u.$$

Both solutions give similar improvements in the speed of convergence in our simulations. To keep the algorithm simple, we chose not to implement either of these fixes in the rest of the paper. There are many other tricks that one can play with to speed up convergence, such as increasing the damping factor $a$ as a function of time, or incorporating multigrid methods. We leave investigations along these lines to future work.

### 3.3 Connection to primal–dual methods

In the context of the Dirichlet problem, there is a close connection between primal–dual methods [49], and PDE acceleration. This was explored briefly in [49], where it was observed that their primal–dual algorithm for solving the Dirichlet problem can be interpreted as a numerical scheme for a damped wave equation. We go further here, and give a PDE interpretation of primal–dual methods and show exactly how they are related to PDE acceleration for the Dirichlet problem.

We recall that the convex dual, or Legendre–Fenchel transform, of a function $\Phi : \mathbb{R}^n \to \mathbb{R}$ is

$$\Phi^*(p) = \max_{x \in \mathbb{R}^n} \{ x \cdot p - \Phi(x) \}. \tag{45}$$

If $\Phi$ is convex, then by convex duality we have $\Phi^{**} := (\Phi^*)^* = \Phi$. We assume $\Phi : \mathbb{R}^n \to \mathbb{R}$ is convex and consider for simplicity the problem

$$\min_u \int_\Omega \Phi(\nabla u) \, \mathrm{d}x, \tag{46}$$

subject to a Dirichlet boundary condition $u = g$. A primal–dual algorithm for solving (46) expresses $\Phi$ through its convex dual $\Phi^*$ giving the initially more looking complicated formation

$$\min_u \max_p \int_\Omega p \cdot \nabla u - \Phi^*(p) \, \mathrm{d}x. \tag{47}$$

Here, $u : \Omega \to \mathbb{R}$ is the primal variable and $p : \Omega \to \mathbb{R}^n$ is the dual variable. Given $p \cdot \mathbf{n} = 0$ on $\partial\Omega$, we can integrate by parts to express the problem as

$$\min_u \max_p \int_\Omega -u \operatorname{div}(p) - \Phi^*(p) \, \mathrm{d}x. \tag{48}$$

The primal–dual algorithm in [49] solves (47) by alternating proximal updates on $p$ and $u$ until convergence (see Sect. 4.2). In the continuum, this is equivalent to jointly performing gradient descent on $u$ and gradient ascent on $p$, which corresponds to the coupled PDEs

$$\begin{cases} p_t = a(t)(\nabla u - \nabla\Phi^*(p)), \\ u_t = \operatorname{div}(p). \end{cases} \tag{49}$$

The factor $a(t)$ is the ratio of the time steps between the proximal updates on $u$ and $p$ in the primal–dual algorithm; in the notation of [49], $a = r_1/r_2$. To the best of our knowledge, this PDE interpretation of primal–dual algorithms is a new observation. In particular, we use this observation to optimally set ratio $r_1/r_2$ for the primal–dual method in Sect. 4.2.

For the Dirichlet problem, $\Phi(p) = \Phi^*(p) = \frac{1}{2}|p|^2$ and (49) becomes

$$\begin{cases} p_t = a(t)(\nabla u - p), \\ u_t = \operatorname{div}(p). \end{cases} \tag{50}$$

In this case, we can eliminate the dual variable and we obtain the damped wave equation

$$u_{tt} + a(t)u_t - a(t)\Delta u = 0. \tag{51}$$

Contrasting this with (44), we see the key difference between primal–dual and PDE acceleration methods is that primal–dual methods are unable to adjust the damping coefficient $a(t)$ independently of other terms in the PDE.

This explicit connection between primal–dual and PDE acceleration seems to be a coincidence for the Dirichlet problem and does not hold in any other case that we are

aware of. In particular, it seems necessary that $\nabla \Phi^*(p)$ is linear in $p$ in order to convert the system (49) into a scalar wave equation in $u$. We can eliminate the primal variable by differentiating the equation for $p_t$ in (50) to obtain

$$p_{tt} + \left( aDF(p) + \frac{a'(t)}{a(t)} \right) p_t = a(t)\nabla \mathrm{div}(p), \tag{52}$$

where $F(p) := \nabla \Phi^*(p)$. However, this is no longer a wave equation.

## 4 Obstacle problems

Consider the standard obstacle problem

$$\min_{u \in \mathcal{A}} E[u] = \int_{\Omega} \Phi(x, \nabla u)\, \mathrm{d}x, \tag{53}$$

where

$$\mathcal{A} = \left\{ u \in H^1(\Omega) \,:\, u \geq \varphi \text{ in } \Omega \text{ and } u = g \text{ on } \partial\Omega \right\}, \tag{54}$$

and the obstacle $\varphi$ satisfies $\varphi \leq g$ on $\partial\Omega$. We recall that the solution $u$ of the obstacle problem (53) satisfies the boundary value problem

$$\begin{cases} \max\{-\nabla E[u], \varphi - u\} = 0 & \text{in } \Omega, \\ \qquad\qquad\qquad u = g & \text{on } \partial\Omega. \end{cases} \tag{55}$$

This is a classical fact; we sketch the formal argument for completeness. If $v \in C^\infty(\overline{\Omega})$ is *nonnegative*, then for any $\varepsilon \geq 0$ we have $u + \varepsilon v \in \mathcal{A}$ and hence

$$E[u + \varepsilon v] - E[u] \geq 0.$$

Dividing by $\varepsilon$ and sending $\varepsilon \to 0^+$ yields

$$\int_{\Omega} \nabla E[u] v\, \mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}\varepsilon}\Big|_{\varepsilon=0} E[u + \varepsilon v] \geq 0 \tag{56}$$

for all nonnegative $v$. Hence, $\nabla E[u] \geq 0$ in $\Omega$. Furthermore, on any ball $B(x, r) \subset \Omega$ where $u > \varphi$, we can relax the nonnegativity constraint on $v$ and still ensure $u + \varepsilon v \in \mathcal{A}$ for small $\varepsilon > 0$. It follows that $\nabla E[u] = 0$ on the set $\{u > \varphi\}$, which establishes (55). We note that solutions of (55) are properly interpreted in the viscosity sense [7,14].

### 4.1 PDE acceleration

We now show how to apply PDE acceleration to the obstacle problem (53). For the moment, we consider the $L^2$-penalized formulation

$$\min_{u \in H^1(\Omega)} \left\{ \int_{\Omega} \Phi(x, \nabla u) + \frac{\mu}{2}(\varphi - u)_+^2\, \mathrm{d}x \,:\, u = g \text{ on } \partial\Omega \right\}. \tag{57}$$

Theorem 1 guarantees that PDE acceleration will converge with a linear rate for any finite $\mu > 0$. However, we need to send $\mu \to \infty$ to recover the solution of the constrained problem (53). We will see, however, that the accelerated PDE method for (57) is insensitive to the choice of $\mu$, and can be easily solved for $\mu > 0$ large, and in numerics, we send $\mu \to \infty$ and obtain a very simple scheme for solving (53). We explain in more detail below.

The PDE-accelerated equations of motion (12) for the penalized problem (57) are

$$u_{tt} + au_t = -\nabla E[u] + \mu(\varphi - u)_+, \tag{58}$$

subject to the Dirichlet condition $u = g$ on $\partial\Omega$, where

$$\nabla E[u] = \text{div}\left(\nabla_p \Phi(x, \nabla u)\right).$$

We now discretize in time using the standard finite differences

$$u_t \approx \frac{u^{n+1} - u^n}{\mathrm{d}t} \quad \text{and} \quad u_{tt} \approx \frac{u^{n+1} - 2u^n + u^{n-1}}{\mathrm{d}t^2}.$$

The important point now is that we handle the penalty term *implicitly*. The discrete in time scheme becomes

$$(1 + a\mathrm{d}t)u^{n+1} - \mu\mathrm{d}t^2(\varphi - u^{n+1})_+ = (2 + a\mathrm{d}t)u^n - u^{n-1} - \mathrm{d}t^2\nabla E[u]. \tag{59}$$

Since the left-hand side is strictly increasing in $u^{n+1}$, there is a unique solution of (59). We can compute the solution explicitly as follows:

$$\begin{cases} v = \dfrac{(2 + a\mathrm{d}t)u^n - u^{n-1} - \mathrm{d}t^2\nabla E[u]}{1 + a\mathrm{d}t}, \\[2mm] w = \dfrac{(2 + a\mathrm{d}t)u^n - u^{n-1} - \mathrm{d}t^2\nabla E[u] + \mu\mathrm{d}t^2\varphi}{1 + a\mathrm{d}t + \mu\mathrm{d}t^2}, \\[2mm] u^{n+1}(x) = \begin{cases} v(x), & \text{if } v(x) \geq \varphi(x), \\ w(x), & \text{otherwise.} \end{cases} \end{cases} \tag{60}$$

The scheme is simple to implement, and the CFL condition is dictated solely by the discretization of $\nabla E[u]$ and is independent of the penalty $\mu$. In practice, we find the algorithm is completely insensitive to the choice of $\mu$ and runs efficiently for, say, $\mu > 10^{10}$.

Instead of choosing a very large value for $\mu$, we can in fact send $\mu \to \infty$ in scheme (60). Indeed, the only place $\mu$ appears is in the update for $w$, and taking the limit as $\mu \to \infty$, we find that $w = \varphi$. Hence, we obtain the simpler scheme

$$\begin{cases} v = \dfrac{(2 + a\mathrm{d}t)u^n - u^{n-1} - \mathrm{d}t^2\nabla E[u]}{1 + a\mathrm{d}t} \\[2mm] u^{n+1}(x) = \max\{v(x), \varphi(x)\} \end{cases} \tag{61}$$

as the limit of (60) as $\mu \to \infty$. In our simulations, we use scheme (61), since it is simpler and more intuitive, but the results are identical, up to machine precision, using scheme (60) with $\mu = 10^{10}$. We use finite differences to discretize $\nabla E[u]$ in this paper—in particular, we discretize the gradient and divergence separately, using forward differences for $\nabla u$ and backward differences for the divergence. We set the damping parameter to be the optimal value $a = 2\pi$ from the linear analysis in Sect. 2.3. We run the iterations (61) until

$$|\max\{-\nabla E[u^n], \varphi - u^n\}| \leq \mathrm{d}x\|\varphi\|_{L^\infty} \tag{62}$$

at all grid points.

We should note there is nothing specific about finite difference schemes in this accelerated framework; one could just as easily use finite elements, spectral methods, or any other numerical PDE method. Once a discretization is settled on, the time step restriction on $\mathrm{d}t$ follows from the CFL condition, which is straightforward to derive (see Sect. 5).

### 4.2 Primal–dual algorithms

Recently in [49], a primal–dual algorithm was proposed for obstacle problems, and it was shown to be several orders of magnitude faster than existing state-of-the-art methods.

We compare PDE acceleration against an improved version of the primal–dual algorithm from [49], which is described below.

The primal–dual algorithm solves the minimal surface obstacle problem

$$\min_{u \in \mathcal{A}} \int_{\Omega} \sqrt{1 + |\nabla u|^2} \, dx, \tag{63}$$

following roughly the outline in Sect. 3.3. We compute the convex dual of $\Phi(x) = \sqrt{1 + |x|^2}$ to be

$$\Phi^*(p) = \begin{cases} -\sqrt{1 - |p|^2}, & \text{if } |p| \leq 1, \\ \infty, & \text{if } |p| > 1. \end{cases} \tag{64}$$

The primal–dual algorithm from [49] for solving (63) solves the equivalent primal–dual formulation

$$\min_{u \geq \varphi} \max_{|p| \leq 1} \int_{\Omega} p \cdot \nabla u + \sqrt{1 - |p|^2} \, dx$$

by alternatively updating the primal variable $u$ and the dual variable $p$ with proximal updates. The full algorithm is given below.

$$\begin{cases} p^{n+1}(x) = \arg \min_{|p| \leq 1} \left\{ -\nabla \overline{u}^n(x) \cdot p - \sqrt{1 - |p|^2} + \frac{1}{2r_1} |p - p^n(x)|^2 \right\}, \\ u^{n+1} = \max\{\varphi, u^n + r_2 \text{div}(p^{n+1})\}, \\ \overline{u}^{n+1} = 2u^{n+1} - u^n. \end{cases} \tag{65}$$

The final step is an overrelaxation, and we set the Dirichlet condition $u = g$ on $\partial \Omega$ at each step. If the problem is discretized on a grid with spacing $dx$, then the method converges for any choices of $r_1, r_2$ with $r_1 r_2 \leq dx^2/6$ [49]. In fact, as noticed in Sect. 3.3, the ratio $r_1/r_2$ plays the role of the damping parameter $a$ in PDE acceleration (61), allowing us to set $r_1/r_2 = 4\pi^2$, which is optimal for $\Omega = [0, 1]^n$ via the linear analysis in Sect. 2.3.

While the update in the dual variable $p(x)$ is pointwise, it is not an explicit update and involves solving a constrained convex optimization problem. We contrast this with the PDE acceleration update (61) which is simple and explicit. In [49], the authors propose to solve the dual problem with iteratively re-weighted least squares (IRLS), that is, they define $\psi^0 = p^n(x)$ and iteratively solve

$$\psi^{k+1} = \arg \min_{|\psi| \leq 1} \left\{ -\nabla \overline{u}^n(x) \cdot \psi + \frac{1}{2} \frac{|\psi|^2 - 1}{\sqrt{1 - |\psi^k|^2}} + \frac{1}{2r_1} |\psi - p^n(x)|^2 \right\}, \tag{66}$$

setting $p^{n+1}(x) = \lim_{k \to \infty} \psi^k$. Actually, in [49] the factor of $\frac{1}{2}$ in front of the term $\frac{|\psi|^2 - 1}{\sqrt{1 - |\psi^k|^2}}$ is missing; this is required to ensure that if the iterations converge, then the fixed point satisfies the correct optimality conditions for the original dual problem. It is claimed in [48,49] that the IRLS iterations converge for $r_1$ sufficiently small. However, inspecting the proof in [48, Lemma 4.4] it appears the restriction on $r_1$ is impractical for $dx \ll 1$. In practice, we find that for $dx \ll 1$ the IRLS iterations drift outside of the unit ball $|\psi^k| \leq 1$ after only a few iterations, in which case (66) is not well defined and the iterations cannot continue. Simple fixes that we tried, such as projecting back onto the unit ball, were found to not be useful. We note we observed failure of the IRLS iterations even for small values of $r_1$.[1]

---

[1]Even if the IRLS iterations were to converge for extremely small $r_1 > 0$, the performance of the primal–dual method is highly sensitive to the ratio $r_1/r_2$, and convergence of the primal–dual iterations is extremely slow for very small or very large $r_1$.

We propose another method for solving the dual problem that is robust and works for any value of $r_1$ and $dx$. We describe our method below. For convenience, let us define

$$F(p) = -\nabla \overline{u}(x) \cdot p - \sqrt{1 - |p|^2} + \frac{1}{2r_1}|p - p^n(x)|^2.$$

We also compute

$$\nabla F(p) = -\frac{1}{r_1}(p^n(x) + r_1 \nabla \overline{u}(x)) + \frac{p}{\sqrt{1 - |p|^2}} + \frac{1}{r_1}p. \tag{67}$$

Then, the dual problem is $p^{n+1}(x) = \arg\min_{|p| \leq 1} F(p)$. We first note that since $F((1 - \varepsilon)p) < F(p)$ for any $p$ with $|p| = 1$ and $\varepsilon > 0$ sufficiently small, we must have $|p^{n+1}(x)| < 1$, and so $\nabla F(p^{n+1}(x)) = 0$. For any $\eta$ with $\eta \cdot p^{n+1}(x) = 0$, we have

$$0 = \eta \cdot r_1 \nabla F(p^{n+1}(x)) = -(p^n(x) + r_1 \nabla \overline{u}(x)) \cdot \eta.$$

Therefore, $p^{n+1}(x) = \alpha q(x)$ for some $\alpha \in (-1, 1)$, where

$$q(x) = \begin{cases} \frac{p^n(x) + r_1 \nabla \overline{u}(x)}{|p^n(x) + r_1 \nabla \overline{u}(x)|}, & \text{if } p^n(x) + r_1 \nabla \overline{u}(x) \neq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{68}$$

The value of $\alpha \in (-1, 1)$ is the unique root of the function

$$f(\alpha) := q(x) \cdot r_1 \nabla F(\alpha q(x)) = \alpha + \frac{r_1 \alpha}{\sqrt{1 - \alpha^2}} - |p^n + r_1 \nabla \overline{u}(x)|. \tag{69}$$

Since $F$ is strictly convex, $f$ is strictly increasing in $\alpha$, and so we can compute the root of $f$ with a simple bisection search. Inspecting (69), we see that $\alpha \in [0, \min\{1, N\}]$, where $N = |p^n + r_1 \nabla \overline{u}(x)|$. For $\alpha$ is this range, we can perform some algebraic manipulations on $f$ to see that we can instead bisect on the function

$$g(\alpha) = r_1^2 \alpha^2 - (1 - \alpha^2)(\alpha - N)^2,$$

which does not involve the costly square root operation. The method is guaranteed to converge, and the accuracy is directly related to the number of bisection iterations, that is,

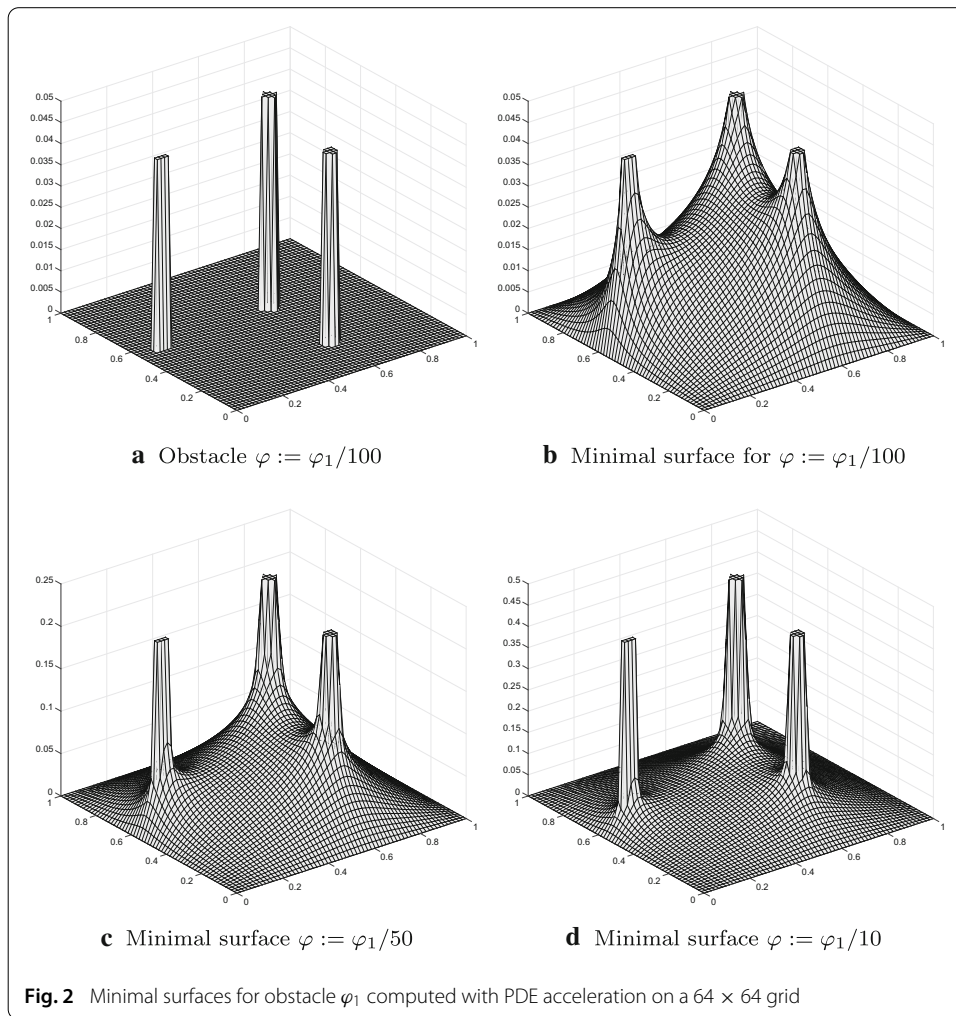$$\text{Bisection search error} \leq \frac{1}{2^{k+1}},$$

where $k$ is the number of bisections.

We emphasize that the IRLS method proposed in [49] does not converge for any of the simulations presented in Sect. 5. Thus, the new bisection method is required to allow comparisons against the primal–dual algorithm for the nonlinear minimal surface problem.

In our implementation of the primal–dual method, we use forward differences for $\nabla u$ and backward differences for the divergence, as in [49]. We set the ratio $r_1/r_2 = 4\pi^2$ based on an optimal linear analysis as in Sect. 2.3, along with the CFL condition $r_1 r_2 \leq dx^2/6$ [49]. We choose the number of bisection iterations so that the dual problem is solved to an accuracy of $\varepsilon dx^2$, where $\varepsilon$ is the accuracy to which we wish to solve the obstacle problem. This requires around 30 iterations for most of our simulations. We run the algorithm until the residual condition (62) is satisfied.

## 5 Experiments

We give here some applications of the PDE acceleration method for solving various obstacle problems. All algorithms, including our improved primal–dual method and the $L^1$-penalty method [38], were implemented in C and run on a laptop with a 64-bit 2.20GHz CPU. The code for all simulations is available on GitHub: https://github.com/jwcalder/MinimalSurfaces.

**a** Obstacle $\varphi := \varphi_1/100$      **b** Minimal surface for $\varphi := \varphi_1/100$

**c** Minimal surface $\varphi := \varphi_1/50$      **d** Minimal surface $\varphi := \varphi_1/10$

**Fig. 2** Minimal surfaces for obstacle $\varphi_1$ computed with PDE acceleration on a $64 \times 64$ grid

## 5.1 Minimal surface obstacle problems

We first consider the constrained minimal surface problem

$$\min \left\{ \int_\Omega \sqrt{1 + |\nabla u|^2} \, \mathrm{d}x \; : \; u \in H_0^1(\Omega) \text{ and } u \geq \varphi \right\}. \tag{70}$$

Here, $\varphi : \Omega \to \mathbb{R}$ is the obstacle and $\Omega = [0, 1]^2$. We solve the problem with the PDE acceleration scheme (61) using the implementation described in Sect. 4.1. Here,
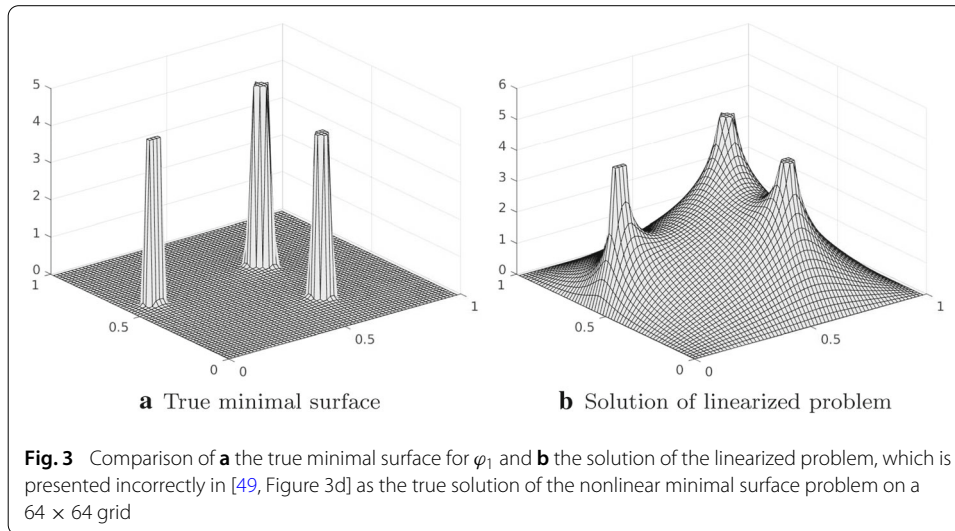
$$\nabla E[u] = -\mathrm{div} \left( \frac{\nabla u}{\sqrt{1 + |\nabla u|^2}} \right),$$

and the CFL condition dictates that $\mathrm{d}t < \mathrm{d}x/\sqrt{2}$. We set $\mathrm{d}t = 0.8\mathrm{d}x/\sqrt{2}$.

The first obstacle we consider is from [49] and is given by

$$\varphi_1(x_1, x_2) = \begin{cases} 5 & \text{for } |x_1 - 0.6| + |x_2 - 0.6| < 0.04, \\ 4.5 & \text{for } (x_1 - 0.6)^2 + (x_2 - 0.25)^2 < 0.001, \\ 4.5 & \text{for } x_2 = 0.57 \text{ and } 0.075 < x_1 < 0.13, \\ 0 & \text{otherwise.} \end{cases} \tag{71}$$

Figure 2 shows the obstacle $\varphi := \varphi_1/100$, and the minimal surfaces computed with the PDE acceleration algorithm for $\varphi := \varphi_1/100$, $\varphi := \varphi_1/20$, and $\varphi := \varphi_1/10$. Figure 2b shows

**a** True minimal surface　　　　　　**b** Solution of linearized problem

**Fig. 3** Comparison of **a** the true minimal surface for $\varphi_1$ and **b** the solution of the linearized problem, which is presented incorrectly in [49, Figure 3d] as the true solution of the nonlinear minimal surface problem on a $64 \times 64$ grid

a short obstacle with small deflections, and the solution in this case is well approximated by the linearized minimal surface problem

$$\min\left\{\int_\Omega \frac{1}{2}|\nabla u|^2 \, \mathrm{d}x \ : \ u \in H_0^1(\Omega) \text{ and } u \geq \varphi\right\}. \tag{72}$$

The obstacles in Fig. 2c, d are significantly taller and the true minimal surfaces are not well captured by linearization. In particular, the minimal surface for $\varphi := \varphi_1$ is nearly identically zero, which we show in Fig. 3a.

We remark that this is in contrast to previous work (see [49, Fig. 3(d)]), which reported that the minimal surface for $\varphi_1$ resembles Fig. 2b (the minimal surface for $\varphi_1/100$). We show the true minimal surface for $\varphi_1$, computed by our algorithm, in Fig. 3a, and the solution of the linearized minimal surface equation in Fig. 3b, which agrees very closely by eye with [49, Fig. 3(d)], suggesting that [49] are in fact solving the linearized minimal surface problem, and not the true nonlinear minimal surface problem. We suspect this is due to the authors of [49] mistakenly taking $\mathrm{d}x = 1$ in their code, which has the effect of drastically reducing the height of the obstacles and putting one in the linear setting. Since the nonlinear minimal surface equation is not homogeneous in the gradient, one cannot scale away the spatial resolution as can be done for the linearized equation.

It is easy to see that Fig. 3b cannot be the true minimal surface for $\varphi_1$ by computing the surface area of the solutions: Our solution in Fig. 3a has surface area 3.9855, while the surface area of Fig. 3b is 8.5105. The obstacle itself has surface area 4.1691. All computations are on the same $64 \times 64$ grid. We can also compute true surface area of the obstacle $\varphi_1$ in the continuum analytically. The obstacles are a square, circle, and a line segment. The square has side lengths $\ell = \sqrt{0.04^2 + 0.04^2} \approx 0.0566$, so perimeter is approximately 0.2263. The height of the square is 5, so the surface area contributed by the obstacle would be $S_1 := 1.1315$. The circle has radius $r = \sqrt{0.001} \approx 0.0316$, and so perimeter is $2\pi r \approx 0.1987$. The height of the circle is 4.5, so it contributes surface area $S_2 := 0.8942$. Finally, the line segment has width 0.055 and height 4.5, so the surface area contribution is $S_3 := 0.2475$. The surface area of the solution is then

$$\text{Surface area} = 1 + S_1 + S_2 + S_3 = 3.2732.$$

**Table 3** Runtimes in seconds and number of iterations for the PDE-accelerated solver, primal–dual method [49], and $L^1$ penalty method [38] for solving the *nonlinear* minimal surface obstacle problem with obstacle $\varphi_1$

| Mesh | Obstacle $\varphi := \varphi_1/50$ | | | | | |
| | Our method | | Primal–dual [49] | | $L^1$ penalty [38] | |
| | Time | Iter. | Time | Iter. | Time | Inner (outer) iter. |
| --- | --- | --- | --- | --- | --- | --- |
| $64 \times 64$ | 0.023 | 360 | 0.186 | 370 | 0.4 | 7254 (2380) |
| $128 \times 128$ | 0.144 | 823 | 1.76 | 870 | 5.6 | 29,284 (10,340) |
| $256 \times 256$ | 1.22 | 1863 | 18 | 2070 | | |
| $512 \times 512$ | 12 | 4135 | 163 | 4390 | | |
| $1024 \times 1024$ | 107 | 9074 | 1650 | 10,210 | | |
| Complexity | 1.54 | 0.58 | 1.64 | 0.60 | 1.9 | 0.93 |

The complexity is measured as a function of the number of grid points used in the discretization

We note there is a large discrepancy between the analytic surface area of the obstacle (3.2723) and the computed area (4.1691) on the $64 \times 64$ grid. The reason for this is that finite difference discretizations of surface area for discontinuous functions are not consistent. This phenomenon is well known in the finite-element community, where pathological examples such as the Schwarz lantern [31] demonstrate the care one must take with numerical discretizations of surfaces and surface area to obtain consistent approximations. We give a simple example of this non-consistency in "Appendix."

We now compare runtimes and iteration counts for our PDE acceleration method against the primal–dual method [49] with our improved bisection method for solving the dual problem presented in Sect. 4.2, and against the $L^1$-penalty method from [38]. Table 3 shows the comparison of the runtimes for the obstacle $\varphi := \varphi_1/50$. Our method is more than $10 \times$ faster than primal–dual in terms of CPU time, while both algorithms have similar iteration counts. The difference is that the PDE acceleration updates are explicit, while the dual update for the primal–dual algorithm is implicit and involves solving a nonlinear optimization problem. We note both algorithms converge to a surface that looks to the eye similar to the minimal surface in about half of the iterations reported, and the final iterations are used to resolve the accuracy to the desired tolerance. For the $L^1$-penalty method, we used parameters $\lambda = 100, \mu = 500, L = 2/\mathrm{d}x^2$, and $\mathrm{d}t = 1/4L$, which gave the best performance over the parameters we tried, and we report both the inner and outer iteration counts for completeness. Each inner iteration has similar complexity to a PDE acceleration iteration. The $L^1$-penalty method did not converge to our strict stopping condition (62) for grids of size $256 \times 256$ or larger. (We stopped the experiment at 18 min and 1 million iterations on the $256 \times 256$ grid.)

Our stopping condition (62) is standard in rigorous scientific computing and is simply asking that all methods solve the same problem to the same accuracy. It is also common to use the difference between subsequent iterates as a stopping condition; however, this can have a different meaning for each algorithm and does not provide any guarantee that the algorithms are solving the correct problem (e.g., one can reduce the time step to encourage "faster" convergence). To be complete, we include in Table 4 a comparison of PDE acceleration, primal–dual, and the $L^1$-penalty method for the stopping condition

$$\|u^{n+1} - u^n\|_{L^\infty} \le C\mathrm{d}x^2, \tag{73}$$

for a constant $C$, which is the same condition used in [38]. We took $C = 1/100$, since for any larger value of $C$, the $L^1$-penalty method stopped at a surface that was clearly by eye

**Table 4** **Runtimes in the same setting as Table 3, except with the relaxed stopping condition** $\|u^{n+1} - u^n\|_{L^\infty} \le dx^2/100$

| Mesh | Obstacle $\varphi := \varphi_1/50$ | | | | | |
|---|---|---|---|---|---|---|
| | Our method | | Primal–dual [49] | | $L^1$ penalty [38] | |
| | Time | Iter. | Time | Iter. | Time | Inner (outer) iter. |
| $64 \times 64$ | 0.014 | 288 | 0.23 | 294 | 0.23 | 4969 (206) |
| $128 \times 128$ | 0.1 | 618 | 2.4 | 684 | 2.94 | 18,228 (385) |
| $256 \times 256$ | 0.9 | 1323 | 23.6 | 1556 | 50.9 | 70,177 (1221) |
| Complexity | 1.5 | 0.55 | 1.67 | 0.6 | 1.95 | 0.96 (0.64) |

In all other experiments, we use the stopping condition (62)

very far from the true minimal surface. We only computed the table up to a $256 \times 256$ grid, since the $L^1$-penalty method did not converge in a reasonable amount of time for larger grids. We see that PDE acceleration and primal–dual have similar complexity, though PDE acceleration is roughly ten times faster, and the $L^1$-penalty method is an order of magnitude slower.

Table 3 (and all future tables) also shows computational complexity, which is computed as the exponent $p > 0$ for which the curve $N^p$ most closely fits the CPU time or iteration count, where $N$ is the number of grid points used in the discretization. We see the complexity of PDE acceleration is $p \approx 1.55$, which agrees with the discussion in Sect. 2.2 for 2D problems. The primal–dual method has slightly worse complexity ($p \approx 1.65$), and the $L^1$ penalty method has complexity $p \approx 2$. We note that complexity for the primal–dual and $L^1$-penalty methods are different than those reported in [49] and [38], respectively. The reason for this is that [49] and [38] report complexity for the *linearized* minimal surface problem, and in particular, do not report runtimes or complexity for nonlinear problems. For the linearized minimal surface problem, the $L^1$-penalty method is reported in [38] to have nearly linear complexity in the number of grid points, which is faster than both primal–dual and our PDE acceleration. Our experiments show that while the $L^1$-penalty method is very fast for linear problems, it is not well suited for nonlinear minimal surface problems.

Let us explain briefly why the nonlinear minimal surface problem is more computationally complex to solve via $L^1$-penalty and primal–dual methods. The $L^1$-penalty method [38] for linear problems involves solving a linear Poisson equation at each outer iteration, which can be done in linear time with multigrid methods, for example, while for nonlinear problems the outer iteration involves solving a nonlinear minimal surface problem, which is more expensive. Interestingly, the authors of [38] use Nesterov acceleration to solve the nonlinear problem at each outer iteration. However, the use of Nesterov acceleration does not employ our optimal damping parameter from Sect. 3, and the momentum is reset at each outer iteration, which we find inhibits the acceleration obtained from momentum methods. Regarding the primal–dual method [49], when solving the linearized problem one can use a handful of IRLS steps to solve the dual problem, as described in [49]. However, for nonlinear problems the IRLS method fails to converge and our new bisection method (see Sect. 4.2) is required, which requires approximately 30–50 iterations. This makes the dual problem more expensive to solve for nonlinear minimal surface problems and explains the difference between computation time for linear and nonlinear problems. To be clear, the IRLS method is unusable for nonlinear problems,
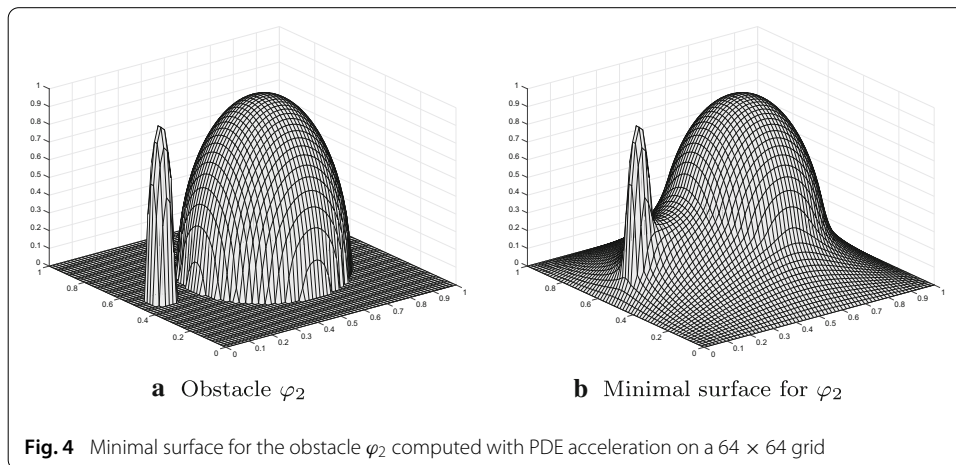
**a** Obstacle $\varphi_2$               **b** Minimal surface for $\varphi_2$

**Fig. 4** Minimal surface for the obstacle $\varphi_2$ computed with PDE acceleration on a $64 \times 64$ grid

**Table 5** Runtimes in seconds and number of iterations for the PDE-accelerated solver, primal–dual method [49], and $L^1$ penalty method [38] for solving the *nonlinear* minimal surface obstacle problem with obstacle $\varphi_2$

| Mesh | Obstacle $\varphi := \varphi_2$ | | | | | |
|---|---|---|---|---|---|---|
| | Our method | | Primal–dual [49] | | $L^1$ penalty [38] | |
| | Time | Iter. | Time | Iter. | Time | Inner (outer) iter. |
| $64 \times 64$ | 0.012 | 300 | 0.182 | 330 | 0.31 | 7065 (60) |
| $128 \times 128$ | 0.138 | 704 | 1.82 | 780 | 3.4 | 19,712 (70) |
| $256 \times 256$ | 1.08 | 1620 | 17.8 | 1720 | 39.8 | 58,788 (170) |
| $512 \times 512$ | 10.2 | 3642 | 180 | 4320 | 551.1 | 199,323 (470) |
| $1024 \times 1024$ | 95.1 | 8117 | 1880 | 9710 | 8401 | 660,908 (1030) |
| Complexity | 1.61 | 0.59 | 1.66 | 0.61 | 1.84 | 0.82 (0.55) |

since it returns complex numbers after a few iterations, so it does not even provide an approximate solution to the problem.

The second obstacle we consider is

$$\varphi_2(x) = \sqrt{(1 - |x - P|^2/0.09)_+} + \sqrt{(1 - |x - Q|^2/0.0025)_+}, \tag{74}$$

where $P = (0.55, 0.5)$ and $Q = (0.1, 0.5)$. Figure 4 shows the obstacle and minimal surface. The runtimes for the PDE acceleration, primal–dual, and the $L^1$-penalty method are given in Table 5. We again see that PDE acceleration is approximately $10\times$ faster in terms of CPU time, and the $L^1$-penalty method is an order of magnitude slower.

### 5.2 Double obstacle with forcing

Here, we consider the double obstacle problem with forcing from [49] (originally from [39]; see also [4] for double obstacle problems ). The nonlinear version of the problem is given by

$$\min \left\{ \int_\Omega \sqrt{1 + |\nabla u|^2} - uv \, \mathrm{d}x \ : \ u \in H_0^1(\Omega) \text{ and } \psi \geq u \geq \varphi \right\}. \tag{75}$$

Scheme (61) is simple to modify for the double obstacle problem by setting $u^{n+1} = \max\{\min\{v, \psi\}, \varphi\}$ at each iteration. We test PDE acceleration on the elasto-plastic torsion problem, originally from [39]. In this setting, $\Omega = [0, 1]^2$, $\varphi_3(x) = -\mathrm{dist}(x, \partial\Omega)$, $\psi_3(x) =$

**Table 6** Runtimes in seconds for the PDE-accelerated solver compared to the primal–dual method from [49] for the *linear* and *nonlinear* minimal surface double obstacle problem with forcing (the elasto-plastic torsion problem)

| Mesh | Linear double obstacle problem | | | | Nonlinear double obstacle problem | | | |
|---|---|---|---|---|---|---|---|---|
| | Our method | | Primal–dual [49] | | Our method | | Primal–dual [49] | |
| | Time | Iter. | Time | Iter. | Time | Iter. | Time | Iter. |
| $64^2$ | 0.012 | 378 | 0.01 | 356 | 0.016 | 382 | 0.156 | 360 |
| $128^2$ | 0.1 | 835 | 0.086 | 814 | 0.133 | 862 | 1.59 | 810 |
| $256^2$ | 0.69 | 1807 | 0.785 | 1884 | 1.23 | 1937 | 15.2 | 1810 |
| $512^2$ | 5 | 3937 | 7.56 | 4092 | 11.9 | 4297 | 143 | 4050 |
| $1024^2$ | 62.9 | 8459 | 81.7 | 9113 | 108 | 9409 | 1.540 | 9000 |
| Comp. | 1.52 | 0.56 | 1.62 | 0.58 | 1.6 | 0.58 | 1.65 | 0.58 |

0.2, $u = 0$ on $\partial\Omega$, and the force $v$ is given by

$$v(x) = \begin{cases} 300, & \text{if } x \in S := \{|x_1 - x_2| \leq 0.1 \text{ and } x_1 \leq 0.3\}, \\ -70e^{x_2}g(x), & \text{if } s \notin S \text{ and } x_1 \leq 1 - x_2, \\ 15e^{x_2}g(x), & \text{if } s \notin S \text{ and } x_1 > 1 - x_2, \end{cases} \tag{76}$$

where

$$g(x) = \begin{cases} 6x_1, & \text{if } 0 \leq x_1 \leq 1/6, \\ 2(1 - 3x_1), & \text{if } 1/6 < x_1 \leq 1/3, \\ 6(x_1 - 1/3), & \text{if } 1/3 < x_1 \leq 1/2, \\ 2(1 - 3(x_1 - 1/3)), & \text{if } 1/2 < x_1 \leq 2/3, \\ 6(x_1 - 2/3), & \text{if } 2/3 < x_1 \leq 5/6, \\ 2(1 - 3(x_1 - 2/3)), & \text{if } 5/6 < x_1 \leq 1. \end{cases} \tag{77}$$

We then set $\varphi := \varphi_3/10$, $\psi = \psi_3/10$ and $v := v_3/10$ to get similar results to [49] where the linearization is studied. For comparison with [49], we also consider the linear double obstacle problem

$$\min\left\{\int_{\Omega} \frac{1}{2}|\nabla u|^2 - uv \, dx \ : \ u \in H_0^1(\Omega) \text{ and } \psi \geq u \geq \varphi\right\}. \tag{78}$$

We report the CPU runtimes and iteration counts for the PDE acceleration method and the improved primal–dual method for both the linear and nonlinear double obstacle problems in Table 6. We see that for the nonlinear problem, PDE acceleration is again roughly $10\times$ faster than primal–dual, while only 2x faster for the linear obstacle problem. The difference is that the dual update is explicit for linear problems, which leads to a substantial acceleration. Figure 5 shows the computed membrane on a $64 \times 64$ grid, and the double obstacle contact regions computed on a $512 \times 512$ grid. These agree well with the results in [49, Fig. 5].

### 5.3 Stochastic homogenization with obstacles
We consider the stochastic obstacle problem

$$\min_{u \in H_0^1(\Omega)} \left\{\int_{\Omega} \frac{1}{2}|A(\tfrac{x}{\varepsilon})\nabla u|^2 - fu \, dx \ : \ u \geq \varphi \text{ in } \Omega\right\}, \tag{79}$$

where $A(x)$ is sampled from a $\mathbb{Z}^d$-stationary probability measure with unit range dependence. We consider here a random checkerboard, where we let $(b(z))_{z \in \mathbb{Z}^2}$ be independent
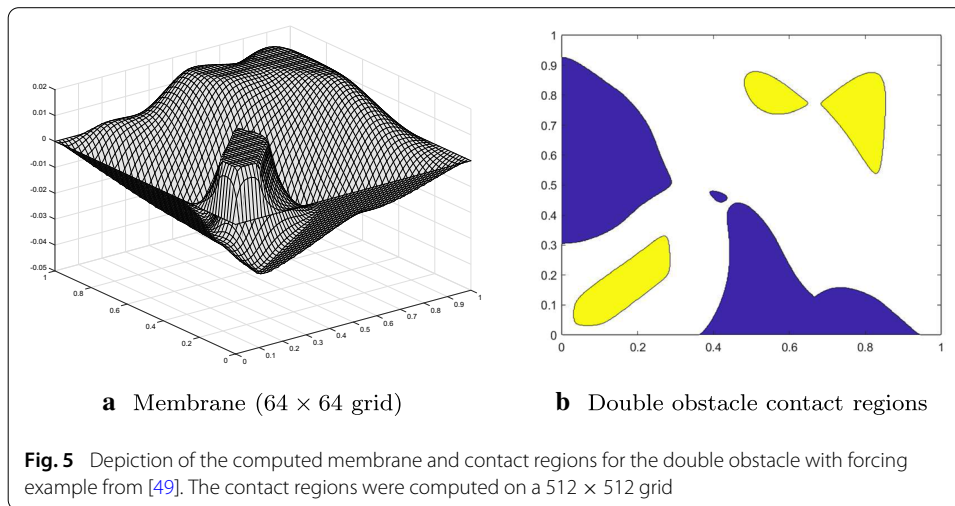
**a**  Membrane (64 × 64 grid)                    **b**  Double obstacle contact regions

**Fig. 5**  Depiction of the computed membrane and contact regions for the double obstacle with forcing example from [49]. The contact regions were computed on a 512 × 512 grid

**Table 7  Runtimes for the PDE-accelerated solver on the stochastic homogenization obstacle problem with damping parameters $a = 2\pi, 6\pi, 9\pi$**

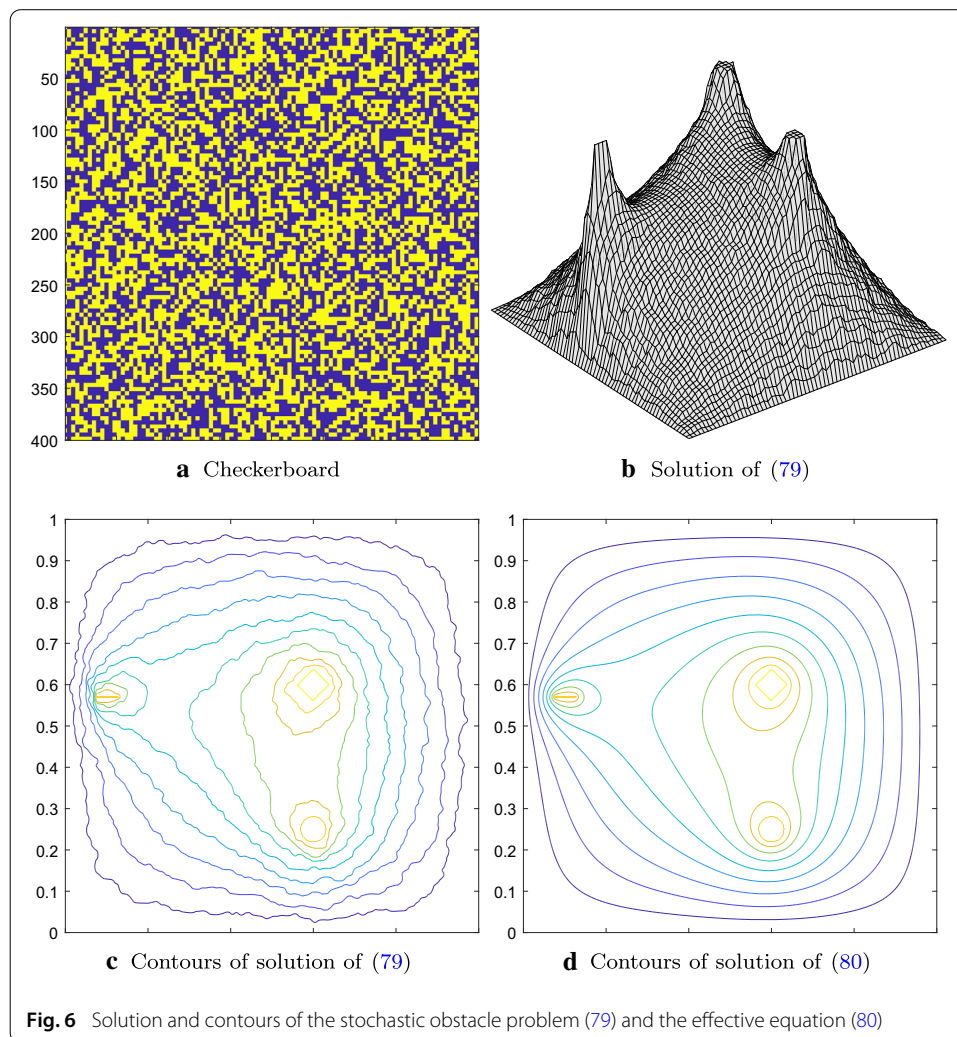| Damping | | $a = 2\pi$ | | $a = 6\pi$ | | $a = 9\pi$ | |
|---|---|---|---|---|---|---|---|
| Checkerboard | Mesh | Time (s) | Iter. | Time (s) | Iter. | Time (s) | Iter. |
| 16 × 16 | $64^2$ | 0.037 | 1665 | 0.012 | 572 | 0.012 | 569 |
| 32 × 32 | $128^2$ | 0.315 | 3924 | 0.115 | 1340 | 0.12 | 1469 |
| 64 × 64 | $256^2$ | 3.35 | 8919 | 1.16 | 3087 | 1.34 | 3588 |
| 128 × 128 | $512^2$ | 31.2 | 20,224 | 9.91 | 6908 | 10.7 | 7482 |
| 256 × 256 | $1024^2$ | 339 | 45,003 | 109 | 15,197 | 118 | 16,425 |
| Complexity | | 1.65 | 0.59 | 1.64 | 0.59 | 1.65 | 0.6 |

random variables such that

$$\mathbb{P}(b(z) = 1) = \mathbb{P}(b(z) = 9) = \frac{1}{2},$$

and set $A(x) = b(z)$ for $x \in z + [0, 1)^2$. We also set $f = 1$. By the Dynkin formula [2, Ex. 2.3], solutions of (79) converge almost surely to solutions of the homogenized problem

$$\min_{u \in H_0^1(\Omega)} \left\{ \int_\Omega \frac{1}{2}|3\nabla u|^2 - u \, dx \, : \, u \geq \varphi \text{ in } \Omega \right\}, \tag{80}$$

as $\varepsilon \to 0^+$. Experiments with this example (without the obstacle) are also presented in [1].

We ran some experiments using the PDE acceleration method for solving this stochastic obstacle problem. Table 7 shows the runtimes for different values of the damping parameter $a$. Figure 6 shows a random checkerboard, the solution of the stochastic obstacle problem (79), and the solution of the homogenized problem (80). We mention that knowledge of the effective (homogenized) equation (80) can help in selecting the optimal damping parameter for the stochastic problem (79). Indeed, by (39) the optimal damping for the homogenized equation (without the obstacle) is $a = 2\sqrt{3}\pi$, which is larger than the damping $a = 2\pi$ we have been using in this paper so far. However, since the discussion in Sect. 2.3 does not consider the obstacle, the true optimal damping parameter will depend on the smallest eigenvalue of the effective operator on the domain $\{u > \varphi\}$, which is initially unknown. Since this domain is strictly smaller than $\Omega$, monotonicity of eigenvalues implies that the optimal damping is larger than our computed $a = 2\sqrt{3}\pi$. We find (see Table 7) that $a = 6\pi$ is close to optimal for this problem.

**a** Checkerboard

**b** Solution of (79)

**c** Contours of solution of (79)

**d** Contours of solution of (80)

**Fig. 6** Solution and contours of the stochastic obstacle problem (79) and the effective equation (80)

## 6 Conclusion

We studied the recently introduced variational framework, called PDE acceleration, for applying accelerated gradient descent (or momentum descent) to problems in the calculus of variations. For a large class of convex optimization problems, the descent equations for PDE acceleration correspond to a nonlinear damped wave equation, which can be solved by a simple explicit forward Euler scheme. The acceleration is realized as a relaxation of the CFL condition for a wave equation ($dt \sim dx$) compared to a diffusion equation ($dt \sim dx^2$). We proved convergence with a linear rate for this class of accelerated PDEs and applied the method to minimal surface obstacle problems, including a double obstacle problem with forcing and a stochastic homogenization problem with obstacle constraint. In every case, PDE acceleration is faster than existing state-of-the-art methods for nonlinear minimal surface problems.

We mention briefly some ideas for future work. First, we use the damping parameter $a = 2\pi$ throughout the whole paper, which is surely not optimal for every problem. We can achieve faster convergence for many experiments in the paper by hand tuning the damping. The difficulty in selecting the optimal damping is that it depends on the first Dirichlet eigenvalue (in the linear case) on the free boundary domain $\{u > \varphi\}$, which is $a$

*priori* unknown. A way to improve performance could be to compute the solution first on a coarse grid, and then estimate the optimal damping from the computed free boundary and use the optimal damping parameter when solving the equation on a finer mesh.

There are other natural ways to speed up convergence, such as considering a multigrid approach, or varying the damping parameter over time. The damping parameter controls the damping profile in the frequency domain; larger choices of the damping parameter give preference to damping higher frequencies at the expense of leaving lower frequencies underdamped. This is reminiscent of how the choice of grid resolution affects the damping in multigrid methods, and a smart choice of a schedule for varying the damping parameter may result in a significant speedup.

Finally, the methods here are not restricted to second-order equations and can be applied almost directly to higher-order equations, such as the fourth-order PDEs that have proven popular in image processing [37,45,47]. In this case, PDE acceleration will relax the very stiff CFL condition ($dt \sim dx^4$) for fourth-order equations to $dt \sim dx^2$. It is also possible to make other choices for the kinetic energy, which would lead to other flows that may be of interest. (However, due to Ostrogradsky instability [43], the kinetic energy should only contain first derivatives of $u$ in time.) Problems in the calculus of variations arise in virtually all fields of science and engineering and include problems like image segmentation and noise removal [13,28], minimal surfaces [12], and materials science [6], among many others. The results of this paper suggest that PDE acceleration can be a useful tool for solving many of these other problems, and we intend to pursue such applications, and others, in future work.

**Author details**
[1] School of Mathematics, University of Minnesota, Minneapolis, USA, [2] School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, USA.

## A Note on finite difference approximations to surface area

We give an example here to show that finite difference approximations to the surface area of discontinuous functions are not, in general, consistent. Define $\varphi : [0, 1]^2 \to \mathbb{R}$ by

$$\varphi(x) = \begin{cases} 1, & \text{if } 2x_1 - x_2 \geq 0, \\ 0, & \text{otherwise.} \end{cases}$$

The surface area of the graph of $\varphi$ is $1 + \frac{\sqrt{5}}{2}$, the quantity $\frac{\sqrt{5}}{2}$ being the length of the line segment $2x_1 - x_2 = 0$ contained in the box $[0, 1]^2$. Consider an $m \times m$ grid with spacing $h = 1/m > 0$, and discrete surface area

$$S_m := \frac{1}{m^2} \sum_{i,j=1}^{m-1} \sqrt{1 + |\nabla \varphi_{i,j}|^2}, \tag{81}$$

where

$$|\nabla \varphi_{i,j}|^2 = \frac{(\varphi(hi + h, hj) - \varphi(ih, jh))^2}{h^2} + \frac{(\varphi(hi, hj + h) - \varphi(hi, hj))^2}{h^2}.$$

Suppose $m$ is even. Then, there are exactly $\frac{3}{2}m - 1$ grid points for which $|\nabla \varphi_{i,j}|^2 \neq 0$. These are all the grid points $(i, j)$ that intersect the boundary $2i - j = 0$, and their two northern neighbors $(i, j + 1)$ and $(i, j + 2)$. For all of these grid points, $|\nabla \varphi_{i,j}|^2 = \frac{1}{h^2} = m^2$. Therefore,

$$S_m = \frac{1}{m^2}(m^2 - \tfrac{3}{2}m + 1) + \frac{1}{m^2}(\tfrac{3}{2}m - 1)\sqrt{1 + m^2}.$$

As $m \to \infty$, we have $S_m \to 1 + \frac{3}{2} = 2.5$, which is not the correct surface area of $1 + \frac{\sqrt{5}}{2} \approx 2.118$. It is easy to check numerically, by computing $S_m$ via (81), that $S_{10^4} = 2.4997$. Pathological examples like this are well known in the finite-element community and are similar to the famous Schwarz lantern [31].

## References

1. Armstrong, S., Hannukainen, A., Kuusi, T., Mourrat, J.C.: An iterative method for elliptic problems with rapidly oscillating coefficients. arXiv:1803.03551 (2018)
2. Armstrong, S., Kuusi, T., Mourrat, J.C.: Quantitative stochastic homogenization and large-scale regularity. arXiv:1705.05300 (2017)
3. Attouch, H., Goudou, X., Redont, P.: The heavy ball with friction method, I. The continuous dynamical system: global exploration of the local minima of a real-valued function by asymptotic analysis of a dissipative dynamical system. Commun. Contemp. Math. **2**(01), 1–34 (2000)
4. Badea, L., Tai, X.C., Wang, J.: Convergence rate analysis of a multiplicative Schwarz method for variational inequalities. SIAM J. Numer. Anal. **41**(3), 1052–1073 (2003)
5. Bähr, M., Breuß, M., Wunderlich, R.: Fast explicit diffusion for long-time integration of parabolic problems. In: AIP Conference Proceedings, vol. 1863, p. 410002. AIP Publishing (2017)
6. Ball, J.M.: The calculus of variations and materials science. Q. Appl. Math. **56**(4), 719–740 (1998)
7. Bardi, M., Capuzzo-Dolcetta, I.: Optimal Control and Viscosity Solutions of Hamilton–Jacobi–Bellman Equations. Springer, Berlin (2008)
8. Benyamin, M., Calder, J., Sundaramoorthi, G., Yezzi, A.: Accelerated variational PDEs for efficient solution of regularized inversion problems. J. Math. Imaging Vis. (2019). https://doi.org/10.1007/s10851-019-00910-2
9. Bottou, L.: Large-scale machine learning with stochastic gradient descent. In: Proceedings of COMPSTAT'2010, pp. 177–186. Springer, Berlin (2010)
10. Braess, D., Carstensen, C., Hoppe, R.H.: Convergence analysis of a conforming adaptive finite element method for an obstacle problem. Numer. Math. **107**(3), 455–471 (2007)
11. Brezis, H., Sibony, M.: Méthodes d'approximation et d'itération pour les opérateurs monotones. Arch. Ration. Mech. Anal. **28**(1), 59–82 (1968)
12. Caffarelli, L.A.: The obstacle problem revisited. J. Fourier Anal. Appl. **4**(4–5), 383–402 (1998)
13. Chan, T.F., Vese, L.A.: Active contours without edges. IEEE Trans. Image Process. **10**(2), 266–277 (2001)
14. Crandall, M.G., Ishii, H., Lions, P.L.: User's guide to viscosity solutions of second order partial differential equations. Bull. Am. Math. Soc. **27**(1), 1–67 (1992)
15. Evans, L.: Partial Differential Equations. Graduate Studies in Mathematics, vol. 19, 2nd edn. AMS, Providence (2002)
16. Goudou, X., Munier, J.: The gradient and heavy ball with friction dynamical systems: the quasiconvex case. Math. Program. **116**(1–2), 173–191 (2009)
17. Grewenig, S.: FSI schemes: fast semi-iterative solvers for PDEs and optimisation methods. In: Pattern Recognition: 38th German Conference, GCPR 2016, Hannover, Germany, September 12–15, 2016, Proceedings, vol. 9796, p. 91. Springer, Berlin (2016)
18. Hafner, D., Ochs, P., Weickert, J., Reißel, M., Grewenig, S.: FSI schemes: fast semi-iterative solvers for PDEs and optimisation methods. In: German Conference on Pattern Recognition, pp. 91–102. Springer, Berlin (2016)
19. Hintermüller, M., Ito, K., Kunisch, K.: The primal-dual active set strategy as a semismooth Newton method. SIAM J. Optim. **13**(3), 865–888 (2002)
20. Hintermüller, M., Kovtunenko, V.A., Kunisch, K.: Obstacle problems with cohesion: a hemivariational inequality approach and its efficient numerical solution. SIAM J. Optim. **21**(2), 491–516 (2011)
21. Hoppe, R.H.: Multigrid algorithms for variational inequalities. SIAM J. Numer. Anal. **24**(5), 1046–1065 (1987)
22. Johnson, C.: Adaptive finite element methods for the obstacle problem. Math. Models Methods Appl. Sci. **2**(04), 483–487 (1992)
23. LeVeque, R.J.: Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems, vol. 98. SIAM, Philadelphia (2007)
24. Lions, P.L., Mercier, B.: Splitting algorithms for the sum of two nonlinear operators. SIAM J. Numer. Anal. **16**(6), 964–979 (1979)
25. Majava, K., Tai, X.C.: A level set method for solving free boundary problems associated with obstacles. Int. J. Numer. Anal. Model **1**(2), 157–171 (2004)
26. Nesterov, Y.: A method of solving a convex programming problem with convergence rate o (1/k2). Soviet Math. Dokl. **27**, 372–376 (1983)
27. Polyak, B.T.: Some methods of speeding up the convergence of iteration methods. USSR Comput. Math. Math. Phys. **4**(5), 1–17 (1964)
28. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D **60**(1–4), 259–268 (1992)
29. Schaeffer, H., Hou, T.Y.: An accelerated method for nonlinear elliptic pde. J. Sci. Comput. **69**(2), 556–580 (2016)
30. Scholz, R.: Numerical solution of the obstacle problem by the penalty method. Computing **32**(4), 297–306 (1984)
31. Schwarz, H.A.: Sur une définition erronée de l'aire d'une surface courbe. Gesammelte Math. Abh. **1**, 309–311 (1890)
32. Su, W., Boyd, S., Candes, E.: A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. In: Advances in Neural Information Processing Systems 27 (NIPS 2014), pp. 2510–2518 (2014)

33. Sundaramoorthi, G., Yezzi, A.: Accelerated optimization in the PDE framework: formulations for the manifold of diffeomorphisms. arXiv:1804.02307 (2018)
34. Sundaramoorthi, G., Yezzi, A.: Variational PDE's for acceleration on manifolds and applications to diffeomorphisms. In: Advances in Neural Information Processing Systems 31 (NIPS 2018) (2018)
35. Sutskever, I., Martens, J., Dahl, G., Hinton, G.: On the importance of initialization and momentum in deep learning. In: International Conference on Machine Learning, pp. 1139–1147 (2013)
36. Tai, X.C.: Rate of convergence for some constraint decomposition methods for nonlinear variational inequalities. Numer. Math. **93**(4), 755–786 (2003)
37. Tai, X.C., Hahn, J., Chung, G.J.: A fast algorithm for Euler's elastica model using augmented Lagrangian method. SIAM J. Imaging Sci. **4**(1), 313–344 (2011)
38. Tran, G., Schaeffer, H., Feldman, W.M., Osher, S.J.: An L1 penalty method for general obstacle problems. SIAM J. Appl. Math. **75**(4), 1424–1444 (2015)
39. Wang, F., Cheng, X.L.: An algorithm for solving the double obstacle problems. Appl. Math. Comput. **201**(1–2), 221–228 (2008)
40. Ward, C., Whitaker, N., Kevrekidis, I., Kevrekidis, P.: A toolkit for steady states of nonlinear wave equations: continuous time Nesterov and exponential time differencing schemes. arXiv:1710.05047 (2017)
41. Weickert, J., Grewenig, S., Schroers, C., Bruhn, A.: Cyclic schemes for PDE-based image analysis. Int. J. Comput. Vis. **118**(3), 275–299 (2016)
42. Wibisono, A., Wilson, A.C., Jordan, M.I.: A variational perspective on accelerated methods in optimization. Proc. Natl. Acad. Sci. **113**(47), E7351–E7358 (2016)
43. Woodard, R.P.: The theorem of Ostrogradsky. arXiv:1506.02210 (2015)
44. Yezzi, A., Sundaramoorthi, G.: Accelerated optimization in the PDE framework: formulations for the active contour case. arXiv:1711.09867 (2017)
45. You, Y.L., Kaveh, M.: Fourth-order partial differential equations for noise removal. IEEE Trans. Image Process. **9**(10), 1723–1730 (2000)
46. Zhang, Y.: Multilevel projection algorithm for solving obstacle problems. Comput. Math. Appl. **41**(12), 1505–1513 (2001)
47. Zhu, W., Chan, T.: Image denoising using mean curvature of image surface. SIAM J. Imaging Sci. **5**(1), 1–32 (2012)
48. Zosso, D., Osting, B.: A minimal surface criterion for graph partitioning. Inverse Probl. Imaging **10**(4), 1149–1180 (2016)
49. Zosso, D., Osting, B., Xia, M.M., Osher, S.J.: An efficient primal-dual method for the obstacle problem. J. Sci. Comput. **73**(1), 416–437 (2017)

## Publisher's Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.