

A Sharp-Interface Active Penalty Method for the Incompressible Navier–Stokes Equations

D. Shirokoff · J.-C. Nave

Received: 29 July 2013 / Revised: 10 February 2014 / Accepted: 20 March 2014 /
Published online: 11 April 2014
© Springer Science+Business Media New York 2014

Abstract The volume penalty method provides a simple, efficient approach for solving the incompressible Navier–Stokes equations in domains with boundaries or in the presence of moving objects. Despite the simplicity, the method is typically limited to first order spatial accuracy. We demonstrate that one may achieve high order accuracy by introducing an *active penalty term*. One key difference from other works is that we use a sharp, unregularized mask function. We discuss how to construct the active penalty term, and provide numerical examples, in dimensions one and two. We demonstrate second and third order convergence for the heat equation, and second order convergence for the Navier–Stokes equations. In addition, we show that modifying the penalty term does not significantly alter the time step restriction from that of the conventional penalty method.

Keywords Active penalty method · Sharp mask function · Immersed boundary · Incompressible flow · Navier–Stokes · Heat equation

Mathematics Subject Classification 65N06 · 65N35 · 65N85 · 76M20 · 76M22

1 Introduction

There are many popular methods for numerically solving the incompressible Navier–Stokes equations in complex geometries. For instance, the immersed boundary method [24], the immersed interface method [21] and the ghost fluid method [11] are popular since they allow one to use a regular grid with an immersed domain boundary. Other efficient methods for the Navier–Stokes or heat equation include [13, 14, 23]. These methods not only use a regular grid, but also utilize level set functions to ensure a sharp interface. In all cases, the regular grid

D. Shirokoff (✉) · J.-C. Nave
Department of Mathematics and Statistics, McGill University, Montreal, QC H3A0B9, Canada
e-mail: david.shirokoff@mail.mcgill.ca

J.-C. Nave
e-mail: jcnave@math.mcgill.ca

and level set formulation alleviates many of the numerical difficulties introduced by curved or moving boundaries. In this paper, we focus on the volume penalty method [1, 2, 6, 7, 18], which loosely fits into the same class of methods.

As a result of their simplicity, penalty methods have been used in a wide variety of problems including electromagnetism, magnetohydrodynamics [22], shape optimization [9], fluid–solid interaction problems [10, 17] and even simpler problems such as the heat equation or Poisson equation [25]. In the context of fluids, they provide a simple means for solving the incompressible Navier–Stokes equations in domains with boundaries. The approach relies on replacing the often difficult to implement Dirichlet fluid boundary conditions, with a simpler to implement volumetric forcing term in the advection equation.

Despite the simplicity, the penalty method suffers from i) poor convergence in the penalty parameter, thereby restricting the accuracy of numerical methods and, ii) a lack of regularity in the velocity field which reduces the advantages of spectral methods. For example, solutions to the penalized equations have a discontinuous second derivative which limits the decay rate of the Fourier coefficients, as well as the ability to spectrally compute derivatives. Despite the lack of smoothness, stable and low order spectral methods have been successfully used to solve the penalized fluid equations [17, 19].

The focus of our paper is to introduce a systematic method for improving the accuracy of penalty methods. Current methods which improve accuracy rely on introducing a subgrid numerical construct in the vicinity of the domain boundary [26, 27]. Such approaches, however, are restrictive if one wishes to eventually use high order Fourier methods. One distinct difference with our approach is that we alter the equations at the continuous level to improve the analytic convergence rate of the penalized problem to the original unpenalized problem. The improved analytic convergence rate then allows for higher order numerical schemes.

We first introduce the original volume penalty method, followed by an introduction to the improved *active penalty method*. We then explicitly show how to analytically construct the new penalty term. Following the construction, we then examine a model equation to demonstrate how the active penalization improves the convergence rate for the Poisson equation.

After discussing the improved convergence, we focus on numerical details. First, we examine the stability of the new active penalty term, and show that it does not introduce additional numerical stiffness. We then provide numerical examples for the heat equation, in dimensions one and two, showing second and third order schemes. Lastly, we outline how to handle the divergence constraint for the Navier–Stokes equations and provide numerical examples showing second order convergence (in L^∞) in the velocity field and first order in the pressure.

2 Navier–Stokes and Volume Penalty Equations

The aim of our work is to examine the behavior of a fluid in the vicinity of a solid or a porous medium. For instance, two examples include the motion of a fluid in a bounded domain with hard walls, or the motion around an immersed solid body such as the one shown in Fig. 1. In our case, we consider dimensions $D = 2, 3$ and let $\Omega_p \subset \mathbb{R}^D$ denote the physical fluid domain. For our purposes, Ω_p is an open set with C^2 boundary $\Gamma = \partial\Omega_p$.

2.1 Incompressible Navier–Stokes Equations

Through the conservation of mass and momentum, the incompressible Navier–Stokes equations govern the flow of an incompressible fluid for $\mathbf{x} \in \Omega_p$

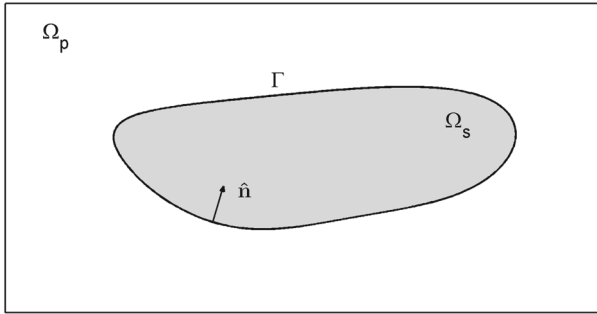


Fig. 1 Illustration of the physical fluid (Ω_p) and solid obstacle (Ω_s) domains

$$\partial_t \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} = -\nabla p + \mu \Delta \mathbf{u} + \mathbf{f} \tag{1}$$

$$\nabla \cdot \mathbf{u} = 0. \tag{2}$$

Here $\mathbf{u}(\mathbf{x}, t)$ is the velocity vector field, $p(\mathbf{x}, t)$ is the pressure, $\mu > 0$ is the kinetic viscosity, and $\mathbf{f}(\mathbf{x}, t)$ is an external forcing such as gravity.

To supplement the bulk equations (1)–(2), the fluid velocity also satisfies prescribed boundary conditions

$$\mathbf{u} = \mathbf{g} \quad \text{for } \mathbf{x} \in \Gamma \tag{3}$$

$$\int_{\Gamma} \mathbf{g} \cdot \mathbf{n} \, dA = 0 \tag{4}$$

Here \mathbf{n} is an outward pointing normal, while Eq. (4) represents a consistency condition on the boundary data. Although we allow $\mathbf{g} = \mathbf{g}(\mathbf{x}, t)$ to be a function of both space and time, the case of $\mathbf{g} = 0$ represents the practical condition of a no-slip and no-flux boundary condition. Together, Eqs. (1)–(2) with boundary data (3) describe the evolution of an initial, divergence-free velocity field $\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0(\mathbf{x})$.

2.2 Volume Penalty Equations

Domains with curved boundaries Γ present several challenges to the numerical solution of Eqs. (1)–(2). For example, curved boundaries or immersed objects limit the use of Fourier methods since solutions are not periodic, or easily extended to periodic functions. One simple solution to handle complicated or moving boundaries is through the use of a volume penalty term in the Navier–Stokes equations. In such a case, one removes the Navier–Stokes boundary condition, and instead adds a drag term to the momentum equation.

To introduce the penalized equation, we first denote Ω_s as the solid domain of the obstacle or wall. Here the obstacle region $\Omega_s = \overline{\Omega}_s$ is a closed set which shares the same boundary as the fluid, $\partial\Omega_s = \Gamma$. The penalized equations are then defined on a computational domain Ω which is the union of the physical and solid domains $\Omega = \Omega_p \cup \Omega_s$. In our case we take Ω to be a rectangular domain with periodic boundary conditions, i.e. $\Omega = \mathbb{T}^D$ where \mathbb{T}^D is the D-dimensional torus.

For a stationary obstacle with a $\mathbf{g} = 0$ boundary condition, the volume penalty equations (see [3–5]) are

$$\partial_t \mathbf{u}_\eta + \mathbf{u}_\eta \cdot \nabla \mathbf{u}_\eta = -\nabla p_\eta + \mu \Delta \mathbf{u}_\eta + \mathbf{f} - \eta^{-1} \chi_s \mathbf{u}_\eta \quad \mathbf{x} \in \Omega \tag{5}$$

$$\nabla \cdot \mathbf{u}_\eta = 0. \tag{6}$$

Here η is a small parameter, and $\chi_s(\mathbf{x})$ is the characteristic function on Ω_s , namely

$$\chi_s(\mathbf{x}) = \begin{cases} 0 & \text{for } \mathbf{x} \in \Omega \setminus \Omega_s \\ 1 & \text{for } \mathbf{x} \in \Omega_s \end{cases}. \tag{7}$$

In the limit $\eta \rightarrow 0$, the drag term in Eq. (5) becomes large and tends to slow the fluid inside Ω_s . Rigorous convergence results by Angot et al. [3], and Carbou and Fabrie [8] show that the penalized velocity \mathbf{u}_η converges to the solution of the Navier–Stokes equations \mathbf{u} with an error rate of $O(\eta^{1/2})$ in the $L^2(\Omega_p)$ norm.

2.3 Improved Volume Penalty Equations

Although the volume penalty equations do converge to Navier–Stokes as $\eta \rightarrow 0$, the convergence rate is slow and therefore may limit the accuracy of resulting numerical schemes. For example, let $\mathbf{u}_{\eta,num}$ denote a numerical solution for the penalized equations. One is then interested in quantifying the numerical error for $\mathbf{u}_{\eta,num}$ compared to \mathbf{u} , the solution to the original Navier–Stokes problem (1)–(2). Using the triangle inequality,¹ the error can be controlled by

$$\|\mathbf{u} - \mathbf{u}_{\eta,num}\|_2 \leq \|\mathbf{u} - \mathbf{u}_\eta\|_2 + \|\mathbf{u}_\eta - \mathbf{u}_{\eta,num}\|_2. \tag{8}$$

Rigorous convergence results then bound the first term as $\|\mathbf{u} - \mathbf{u}_\eta\|_2 \sim \eta^{1/2}$, while $\|\mathbf{u}_\eta - \mathbf{u}_{\eta,num}\|_2$ depends on the numerical details and order of the scheme. Finally, we note that the addition of the penalty term introduces time scales of $O(\eta)$ and length scales of $O(\eta^{1/2})$ into the solution \mathbf{u}_η . To appropriately resolve the boundary layers in the penalty equations (5)–(6), one then has a grid spacing of $\Delta x \sim \sqrt{\eta}$ leading to a first order bound

$$\|\mathbf{u} - \mathbf{u}_{\eta,num}\|_2 \leq O(\Delta x). \tag{9}$$

In light of the above observations, a high order penalty method must either increase the boundary layer width $O(\sqrt{\eta})$, or improve the analytic convergence rate in the penalty parameter. We adopt the second approach, and outline how Eq. (5) can be modified to better approximate the original Navier–Stokes problem (1)–(2). Furthermore, we note that when modifying the penalty term, it is important to avoid the introduction of additional length or time scales which would hinder the development of high order numerical schemes. To improve the penalized equations, we exploit the fact that \mathbf{u} satisfies the boundary conditions on Γ , and does not represent a physical flow inside Ω_s . Specifically, we modify the penalty term so that the flow tracks an extension function $\tilde{\mathbf{g}}$ defined on $\mathbf{x} \in \Omega_s$. In such a case, the volume penalty equations take the form

$$\partial_t \mathbf{u}_\eta + \mathbf{u}_\eta \cdot \nabla \mathbf{u}_\eta = -\nabla p_\eta + \mu \Delta \mathbf{u}_\eta + \mathbf{f} - \eta^{-1} \chi_s (\mathbf{u}_\eta - \tilde{\mathbf{g}}) \quad \mathbf{x} \in \Omega \tag{10}$$

$$\nabla \cdot \mathbf{u}_\eta = 0 \quad \mathbf{x} \in \Omega_p. \tag{11}$$

At this point, we only specify the divergence constraint within the physical domain and defer a more detailed description of the divergence constraint inside Ω_s for Sect. 7. The idea is to choose $\tilde{\mathbf{g}}$ to reduce the artificial fluid boundary layer generated by the penalized equations in the vicinity of Γ . Specifically, the function $\tilde{\mathbf{g}}$ should be a smooth, at least C^1 , extension

¹ Here $\|\cdot\|_2$ is any appropriate numerical $L^2(\Omega_p)$ norm.

of the prescribed boundary conditions. The extension is constructed for each component of $\tilde{\mathbf{g}}$, and each component of $\tilde{\mathbf{g}}$ should be chosen to match k derivatives of \mathbf{u} in the direction normal to Γ . As a result, we prescribe the following general properties for $\tilde{\mathbf{g}}$

- P1. $\tilde{\mathbf{g}}$ is an extension of the prescribed boundary values: $\tilde{\mathbf{g}} = \mathbf{g}$ for $\mathbf{x} \in \Gamma$.
- P2. $\tilde{\mathbf{g}}$ has the same normal slope as \mathbf{u} : $(\mathbf{n} \cdot \nabla)u_i = (\mathbf{n} \cdot \nabla)\tilde{g}_i$. Here u_i and \tilde{g}_i for $i = 1 \dots D$ are the components of \mathbf{u} and $\tilde{\mathbf{g}}$.
- P3. For higher derivatives, $(\mathbf{n} \cdot \nabla)^k u_i|_{\Gamma_p} = (\mathbf{n} \cdot \nabla)^k \tilde{g}_i$.

Since derivatives of \mathbf{u} may be discontinuous across Γ , the notation Γ_p denotes the limit of the derivative from the physical domain Ω_p .

3 Constructing the Extension $\tilde{\mathbf{g}}$

In this section we discuss one possible construction for the extension function $\tilde{\mathbf{g}}$. The construction procedure is identical for each component g_i of $\tilde{\mathbf{g}}$ for $i = 1 \dots D$.

In our approach, we assume the domain Ω_p has a smooth boundary, at least $\Gamma \in C^2$. As a result, we omit a class of physically important domains such as rectangles. The general idea is to match the normal derivatives of $\tilde{\mathbf{g}}$ to those of \mathbf{u} on Γ . With the appropriate boundary derivatives, we then let $\tilde{\mathbf{g}}$ decay to some constant value \mathbf{G} over a length scale l . In our construction, the maximum length scale l is bounded by the minimum radius of curvature of the interface.

Step 1. First introduce a family of smooth, one-dimensional basis functions $B_j \in C^k$ with $0 \leq j \leq k$ such that

- (i) The functions B_j form a basis for derivatives at $x = 0$

$$\frac{d^i}{dx^i} B_j(0) = \begin{cases} 1 & \text{for } j = i \\ 0 & \text{for } j \neq i \end{cases} \tag{12}$$

- (ii) Each $B_j(x)$ has support on $[0, 1]$. Namely $B_j(x) = 0$ for $x < 0$ and $x > 1$.

One can then use the functions $B_j(x)$ to construct a C^k extension $\tilde{f}(x)$ of a one-dimensional function $f(x)$ on $x \geq 0$ as

$$\tilde{f}(x) = f(0)B_0(x) + f'(0)B_1(x) + \dots + f^{(k)}(0)B_k(x). \tag{13}$$

Note that by construction, the function $\tilde{f}(x)$ matches k derivatives at $x = 0$ and vanishes for $x > 1$. The goal is now to modify the extension (13) to higher dimensions.

Although there are many different choices for $B_j(x)$, we now give an example of one such choice for matching $k = 2$ derivatives. We do this by constructing $B_j(x)$ out of stretched copies of the smoothly decaying function

$$h(x) = \begin{cases} e^{1-\frac{1}{1-x}} & \text{for } 0 \leq x < 1 \\ 0 & \text{for } x \geq 1 \end{cases}. \tag{14}$$

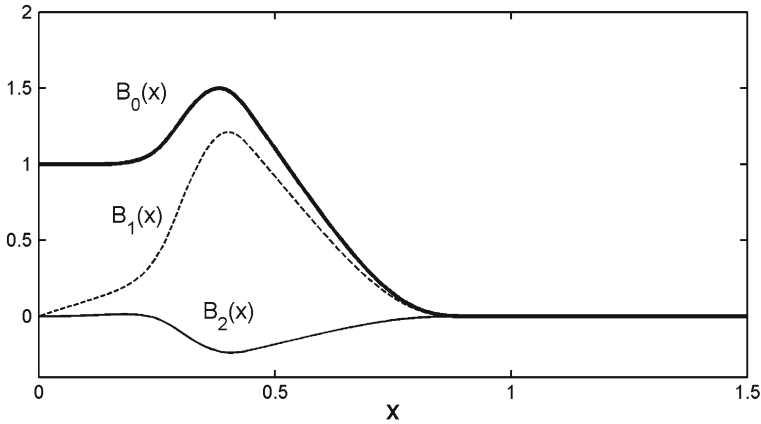


Fig. 2 A plot of the 1D basis functions $B_0(x)$ (thick), $B_1(x)$ (dashed) and $B_2(x)$ (thin)

Using $h(x)$, one can take the functions B_0, B_1, B_2 (Fig. 2) as the weighted sums

$$B_0(x) = 3 h(x) - 3 h(2x) + h(3x) \tag{15}$$

$$B_1(x) = \frac{5}{2} h(x) - 4 h(2x) + \frac{3}{2} h(3x) \tag{16}$$

$$B_2(x) = -\frac{1}{2} h(x) + h(2x) - \frac{1}{2} h(3x). \tag{17}$$

Step 2. Construct a coordinate system inside the obstacle. The coordinate system should be orthogonal at the boundary, and only needs to extend a distance l inside the domain Ω_s .

To construct the coordinates, we follow a standard approach [12] shown in Fig. 3. Let $\xi \in \Gamma$ denote the coordinates of the boundary Γ . Any point \mathbf{x} within a distance l of the boundary may then be written as

$$\mathbf{x} = \xi + s \mathbf{n}(\xi). \tag{18}$$

Here s is the normal distance inside Ω_s from the boundary. Within a small enough region, $s \leq l$, one may invert² equation (18) to write $\xi = \xi(\mathbf{x})$ and $s = s(\mathbf{x})$.

Remark 1 In cases where a level set function $\phi(\mathbf{x})$ describes the boundary $\Gamma = \{\mathbf{x} \in \Omega \mid \phi(\mathbf{x}) = 0\}$, one may identify

$$\mathbf{n} = \nabla\phi \tag{19}$$

$$s = \phi(\mathbf{x}). \tag{20}$$

Here we have assumed $|\nabla\phi| = 1$ and $\phi(\mathbf{x}) > 0$ represents the domain Ω_s while $\phi(\mathbf{x}) < 0$ corresponds to the domain Ω_p . ♠

Step 3. Construct the extension $\tilde{\mathbf{g}}$ using the functions $B_j(x)$ and the coordinates (ξ, s) .

² The coordinates $\xi(\mathbf{x})$ and $s(\mathbf{x})$ are both at least C^1 functions.

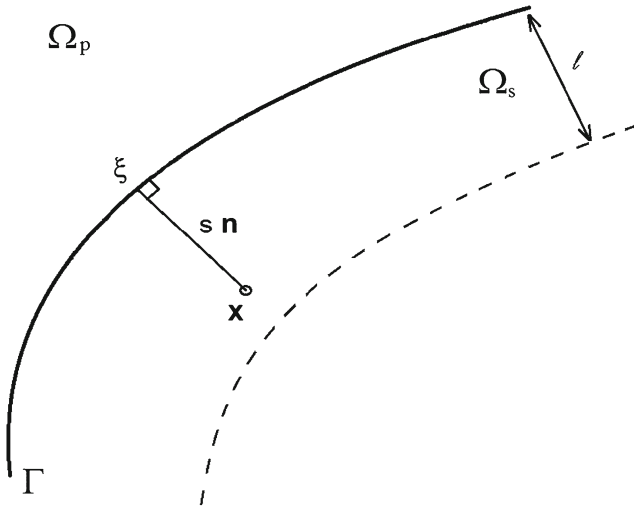


Fig. 3 A local set of normal coordinates. Here $\xi \in \Gamma$ is a point on the boundary, while s is the distance in the normal direction. A coordinate inside a neighborhood of Ω_s has the form $\mathbf{x} = \xi + s \mathbf{n}(\xi)$

For brevity, we introduce notation for the normal derivatives at the boundary Γ .

$$\mathbf{u}_n(\xi) = (\mathbf{n} \cdot \nabla) \mathbf{u}|_{\Gamma} \tag{21}$$

$$\mathbf{u}_{nn}(\xi) = (\mathbf{n} \cdot \nabla)^2 \mathbf{u}|_{\Gamma} \tag{22}$$

\vdots

$$\mathbf{u}_{n_k}(\xi) = (\mathbf{n} \cdot \nabla)^k \mathbf{u}|_{\Gamma} \tag{23}$$

Again, we note that higher derivatives of \mathbf{u} are discontinuous across the boundary. Therefore, \mathbf{u}_{n_k} is evaluated as the limit approaching the boundary from the physical domain. The extension is then

$$\begin{aligned} \tilde{\mathbf{g}}(\mathbf{x}) = & (\mathbf{g}(\xi) - \mathbf{G}) B_0(s l^{-1}) + l \mathbf{u}_n(\xi) B_1(s l^{-1}) \\ & + l^2 \mathbf{u}_{nn}(\xi) B_2(s l^{-1}) + \dots \\ & + l^k \mathbf{u}_{n_k}(\xi) B_k(s l^{-1}) + \mathbf{G}. \end{aligned} \tag{24}$$

Note that $\tilde{\mathbf{g}}$ decays to \mathbf{G} , i.e. $\tilde{\mathbf{g}} \rightarrow \mathbf{G}$, as $s \rightarrow l$. Therefore \mathbf{G} can be any time-dependent constant vector, however, for numerical purposes one should choose \mathbf{G} close to the boundary average of \mathbf{g}

$$\mathbf{G} = A^{-1} \int_{\Gamma} \mathbf{g} dA \tag{25}$$

$$A = \int_{\Gamma} dA. \tag{26}$$

Remark 2 Since values of $\tilde{\mathbf{g}}$ inside Ω_s depend on derivatives of \mathbf{u} on the boundary, the function $\tilde{\mathbf{g}}$ described in (24) depends linearly on \mathbf{u} . ♠

Remark 3 We can check that the construction (24) satisfies the properties (P1)–(P3). For $\mathbf{x} \in \Gamma$, we have $s = 0$, so that $\tilde{\mathbf{g}} = \mathbf{g}$, thereby satisfying (P1). To check higher derivatives, we first note that differentiating (18) with respect to s yields $\frac{\partial \mathbf{x}(\boldsymbol{\xi}, s)}{\partial s} = \mathbf{n}(\boldsymbol{\xi})$. As a result any function independent of s , $y(\boldsymbol{\xi}, s) = y(\boldsymbol{\xi})$, has the property that

$$(\mathbf{n} \cdot \nabla)y(\boldsymbol{\xi}) = \sum_j \frac{\partial x_j}{\partial s} \frac{\partial y(\boldsymbol{\xi})}{\partial x_j} \tag{27}$$

$$= \frac{\partial y(\boldsymbol{\xi})}{\partial s} \tag{28}$$

$$= 0. \tag{29}$$

Meanwhile, we also have

$$(\mathbf{n} \cdot \nabla)^i B_j(s l^{-1})|_{s=0} = \left(\frac{\partial}{\partial s}\right)^i B_j(s l^{-1})|_{s=0} \tag{30}$$

$$= l^{-i} \delta_{ij} \tag{31}$$

where δ_{ij} is the Kronecker delta. Combining the two properties above, we have

$$(\mathbf{n} \cdot \nabla)^i \tilde{\mathbf{g}}|_{\mathbf{x} \in \Gamma} = \mathbf{u}_{n_i}(\boldsymbol{\xi}). \tag{32}$$

Therefore, we recover properties (P2)–(P3). ♠

4 A Model Equation

In this section we examine solutions to the steady-state heat equation to provide some explanation for how the extension function \tilde{g} improves the analytic convergence rate of the penalized equations to the original problem. In particular, we seek to quantify the error that results from the additional penalty forcing. As a non-penalized problem, consider

$$\partial_{xx} v = 0 \quad x \in [-1, 0] \tag{33}$$

with boundary conditions: $v(-1) = 1$, $v(0) = 0$. The solution is then $v(x) = -x$ for $x \in [-1, 0]$.

We note that solving explicit examples of the steady-state equations do not give general sharp convergence estimates, however, they do provide a rigorous lower bound on the convergence rate of the penalized equation to the exact non-penalized equation. The equivalent one dimensional steady-state penalized problem is then

$$\partial_{xx} u = \eta^{-1} H(x)(u - \tilde{g}) \quad x \in [-1, \infty) \tag{34}$$

with boundary conditions $u(-1) = 1$, $u(\infty) = 0$. Here $H(x)$ is the Heaviside function

$$H(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases} \tag{35}$$

We now examine the convergence of solutions $u \rightarrow v$ in the limit $\eta \rightarrow 0$ for different extensions \tilde{g} .

Remark 4 As a result of the discontinuous Heaviside function $H(x)$, the solution u to Eq. (34) has one continuous derivative ($u \in C^1$). Higher derivatives are discontinuous across $x = 0$. ♠

In light of remark (4), we take \tilde{g} to have derivatives matching u from the physical domain $x = 0^-$ and not $x = 0^+$.

Proposition 1 *Suppose that \tilde{g} is a bounded C^{k+1} function that matches k derivatives of u at $x = 0^-$. Namely*

1. $\tilde{g}(0) = 0$
2. $\tilde{g}'(0) = u'(0)$
3. $\tilde{g}^{(m)}(0) = u^{(m)}(0^-) = 0$ for $2 \leq m \leq k$.

Then the solution u converges to v as

$$\max_{x \in [-1, 0]} |u - v| = O(\eta^{(k+1)/2}) \tag{36}$$

Proof In the region $-1 \leq x \leq 0$, u has the solution

$$u = (1 + c) + cx \tag{37}$$

for some constant c . To construct the solution on $x \geq 0$, we note that one may write \tilde{g} as

$$\tilde{g}(x) = cx + x^{k+1}R(x) \tag{38}$$

for some remainder function $R(x)$, where in general $R(0) \neq 0$. By construction (38) matches the first k derivatives of u at $x = 0^-$. In addition, we assume that $\tilde{g}(x)$ and $\tilde{g}'(x)$ are bounded, so that $R(x)$ and $R'(x)$ are also bounded functions. On $x \geq 0$, u then solves

$$\partial_{xx}u - \eta^{-1}u = -\eta^{-1}(cx + x^{k+1}R(x)) \tag{39}$$

To obtain the correct scaling, we rescale $x = \eta^{1/2}X$ to obtain

$$\partial_{XX}u - u = -c\eta^{1/2}X + \eta^{(k+1)/2}X^{k+1}R(\eta^{1/2}X) \tag{40}$$

The general solution is then

$$u(X) = \begin{cases} (1 + c) + c\eta^{1/2}X & \text{if } X < 0 \\ be^{-X} + c\eta^{1/2}X + \eta^{(k+1)/2}Q(X) & \text{if } X \geq 0 \end{cases} \tag{41}$$

where we have excluded the term e^X since it diverges as $X \rightarrow \infty$. In addition, $Q(X)$ is a particular solution (which stays bounded as $X \rightarrow \infty$) to

$$Q_{XX} - Q = X^{k+1}R(\eta^{1/2}X) \tag{42}$$

For instance, one may write a particular solution as

$$Q(X) = \frac{1}{2} \int_0^X (e^{X-y} - e^{-X+y})y^{k+1}R(\eta^{1/2}y) dy - Ae^X \tag{43}$$

$$A = \frac{1}{2} \int_0^\infty e^{-y}y^{k+1}R(\eta^{1/2}y) dy \tag{44}$$

Letting $R_m = \max_y |R(y)|$, we also have the bound

$$|Q(0)| = |A| \leq \frac{R_m}{2} \int_0^\infty y^{k+1}e^{-y} dy = Q_0. \tag{45}$$

The same type of argument holds for bounding $|Q'(0)| \leq Q_1$.

To solve for the unknown constants, c and b , we use the fact that u and u' are continuous across $x = 0$. We therefore obtain the two equations

$$1 + c = b + \eta^{(k+1)/2} Q(0) \tag{46}$$

$$\eta^{1/2} c = -b + \eta^{1/2} c + \eta^{(k+1)/2} Q'(0) \tag{47}$$

Upon solving for b and c , the error between u and v on the physical domain $-1 \leq x \leq 0$ is

$$\max_{x \in [-1, 0]} |u - v| = 1 + c \tag{48}$$

$$= (Q(0) + Q'(0))\eta^{(k+1)/2} \tag{49}$$

$$\leq (Q_0 + Q_1)\eta^{(k+1)/2} \tag{50}$$

$$= O(\eta^{(k+1)/2}) \tag{51}$$

Hence, for the model problem, matching k derivatives of \tilde{g} yields a convergence rate of $(k + 1)/2$. In particular, when $k = 0$, we recover the known convergence rate $\eta^{1/2}$ of the standard penalty method. \square

Remark 5 Using higher derivatives in the construction of \tilde{g} which are taken as limits from the domain $x \searrow 0^+$, does not yield the convergence rate stated in proposition (1). As an example, we take $\tilde{g}^+ = u'(0)B_1(x) + u''(0^+)B_2(x)$ where

$$B_1(x) = \frac{5}{2}e^{-x} - 4e^{-2x} + \frac{3}{2}e^{-3x} \tag{52}$$

$$B_2(x) = \frac{1}{2}e^{-x} - e^{-2x} + \frac{1}{2}e^{-3x}. \tag{53}$$

For such a \tilde{g}^+ , the solution u to problem (34) yields only a first order error

$$\max_{[-1, 0]} |u - v| \sim \frac{11}{6} \eta. \tag{54}$$

In contrast, taking $\tilde{g}^- = u'(0)B_1(x) + u''(0^-)B_2(x)$ yields a convergence rate in agreement with (1)

$$\max_{[-1, 0]} |u - v| \sim 11\eta^{3/2}. \tag{55}$$



5 Stability

In this section we establish numerical stability criteria for the 1D penalized heat equation. To examine stability, we work with the domain $\Omega_p = (0, \pi)$, $\Omega_s = [\pi, 2\pi]$ and periodic boundary conditions. Moreover, we take $g(0) = g(\pi) = 0$ to capture a $u = 0$ boundary condition at the fluid-solid boundary. A simple Euler scheme matching one derivative of u at the interface is then

$$u^{n+1} = (I + \Delta t \Delta)u^n - \Delta t \eta^{-1} \chi_s (u^n - \tilde{g}^n) \tag{56}$$

$$\tilde{g}^n = u_x^n(\pi)B_1(x - \pi) - u_x^n(2\pi)B_1(2\pi - x) \tag{57}$$

In general, adding derivatives of u to $\tilde{g}(x)$ can reduce, by factors of h , the time step restriction for an explicit scheme. In the case at hand, however, the structure of $\tilde{g}(x)$ results in the same time step restriction as the original volume penalty method, namely

$$\Delta t < \min\{O(h^2), O(\eta)\}. \tag{58}$$

Here h is either the grid spacing of a finite difference scheme, or alternatively h^{-1} scales as the largest wavenumber in a Fourier method.

We note that although (56) is a linear recursion relation, the right hand side is not a normal operator. As a result, a rigorous proof of (58) requires bounding the eigenvalues for the spatially discrete system (56). The analysis is further complicated by the fact that the operators (or matrices) on the right hand side of (56) do not commute.

In this section we establish the time step restriction (58). To do so, we first compute the eigenvalues for the penalty term using a finite difference scheme. We show that although the penalty term contains derivatives of u , the eigenvalues remain $O(\eta^{-1})$ and do not become $O(\eta^{-1}h^{-1})$.

Secondly, to show that the addition of the Laplacian does not alter the restriction (58), we numerically compute the eigenvalues for Eq. (56) using a finite difference scheme.

5.1 Eigenvalues of the Penalty Term (Finite Differences)

In practice, one does not observe the time step restriction governed by the norm of \tilde{g} , but rather the larger bound in (58). Here we provide a stability criteria by analytically computing the penalty term eigenvalues for a finite difference scheme.

Let $x_k = kh$ for $0 \leq k \leq N - 1$ with grid spacing $h = 2\pi/N$. Furthermore, denote the discrete vector $\mathbf{u} = [u(x_0) u(x_1) \dots u(x_{N-1})]^T$.

We are then interested in evaluating the eigenvalues of the penalty term

$$\mathbf{B}\mathbf{u} = \lambda\mathbf{u} \tag{59}$$

$$\mathbf{B} = -\eta^{-1}(\mathbf{I}_x - \mathbf{v}_1\mathbf{d}_1^T - \mathbf{v}_2\mathbf{d}_2^T) \tag{60}$$

where \mathbf{B} is the finite difference matrix corresponding to the penalty term. Here \mathbf{I}_x is the identity matrix restricted to $x \in \Omega_s$ while \mathbf{v}_1 and \mathbf{v}_2 , are vectors with components

$$(\mathbf{v}_1)_k = \chi_s(x_k)B_1(x_k - \pi) \tag{61}$$

$$(\mathbf{v}_2)_k = -\chi_s(x_k)B_1(2\pi - x_k) \tag{62}$$

In addition, \mathbf{d}_1 and \mathbf{d}_2 are column vectors which approximate the derivatives of a vector \mathbf{u} as

$$u_x(\pi) \approx \mathbf{d}_1^T \mathbf{u} \tag{63}$$

$$u_x(2\pi) \approx \mathbf{d}_2^T \mathbf{u} \tag{64}$$

For instance, a centered difference approximation to the derivative $u_x(2\pi)$ would have $(\mathbf{d}_2)_{N-1} = -(2h)^{-1}$, $(\mathbf{d}_2)_1 = (2h)^{-1}$ and $(\mathbf{d}_2)_k = 0$ for $k = 0$ and $1 < k < N - 1$. Lastly, since the support of $B_1(x)$ is restricted to $x < 1$, the function $B_1(x - \pi) = 0$ for $x > \pi + 1$. Hence, the numerical derivative of $B_1(x - \pi)$ at $x = 2\pi$ is zero (or similarly with $B_1(2\pi - x)$ at $x = 0$)

$$\mathbf{d}_2^T \mathbf{v}_1 = 0 \tag{65}$$

$$\mathbf{d}_1^T \mathbf{v}_2 = 0. \tag{66}$$

Combining the orthogonality conditions (65)–(66) with the fact that $\mathbf{I}_x \mathbf{v}_{1,2} = \mathbf{v}_{1,2}$, implies that $\mathbf{u} = \mathbf{v}_{1,2}$ are eigenvectors with corresponding eigenvalues

$$\lambda_1 = -\eta^{-1}(1 - \mathbf{d}_1^T \mathbf{v}_1) \tag{67}$$

$$\lambda_2 = -\eta^{-1}(1 - \mathbf{d}_2^T \mathbf{v}_2). \tag{68}$$

All other eigenvalues of \mathbf{B} then lie in the space perpendicular to \mathbf{v}_1 and \mathbf{v}_2 resulting in either $\lambda = 0$ or $\lambda = -\eta^{-1}$. The eigenvalues (67)–(68) are therefore directly a result of the modified penalty term and depend specifically on the component values of $\mathbf{d}_{1,2}$. As a result, the products $\mathbf{d}_{1,2}^T \mathbf{v}_{1,2} \in (0, 1]$ depending on how one builds the numerical derivative vector $\mathbf{d}_{1,2}$.

As an example, taking a centered difference approximation to the derivative $u_x(2\pi)$ yields

$$\mathbf{b}_2^T \mathbf{v}_2 = \frac{1}{2h}((\mathbf{v}_2)_1 - (\mathbf{v}_2)_{N-1}) \tag{69}$$

$$\approx 0.5. \tag{70}$$

The second line follows since $(\mathbf{v}_2)_{N-1} = 0$ while $(\mathbf{v}_2)_1 \approx h$ because the function $B'_1(0) = 1$.

In general, the product $\mathbf{d}_{1,2}^T \mathbf{v}_{1,2}$ will be a weighted average of the derivatives of $B_1(x)$ on the left and right of the interface. As a result, all eigenvalues λ of \mathbf{B} satisfy $-\eta^{-1} \leq \lambda \leq 0$. Therefore, modifying the penalty term does not change the time step restriction $\Delta t < 2\eta$ for a simple Euler scheme $\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t \mathbf{B}\mathbf{u}^n$.

5.2 Numerical Eigenvalues

In the follow section we numerically compute the eigenvalues of (56)–(57) using a finite difference scheme for the spatial derivatives.³ The scheme then has the form

$$\mathbf{u}^{n+1} = [\mathbf{I} + \Delta t (\mathbf{L} + \mathbf{B})]\mathbf{u}^n \tag{71}$$

where \mathbf{L} is the standard 3-point stencil discrete Laplacian. As a result, the eigenvalues of the linear system (71) approach the real values associated with the Laplacian Δ when $\eta \rightarrow \infty$, and the values associated with the penalty term when $\eta \rightarrow 0$.

To compute the eigenvalues numerically, we fix a grid with N points ($256 \leq N \leq 4096$) and examine the range $10^{-9} \leq \eta \leq 1$.

Proposition 2 (Practical stability) *In practice, the numerical scheme (71) is stable provided one takes the time step restriction*

$$\Delta t < \min\{0.5h^2, 1.2\eta\}. \tag{72}$$

Remark 6 The exact constant 1.2 in (72) depends on numerical details such as how one interpolates derivatives to the interface or the nature of the functions $B_0(x), B_1(x) \dots$ ♠

Here Fig. 4 shows that the numerical eigenvalues for $N = 2048$ and $\eta = 10^{-7}$ are stable with a time step restriction (72).

6 Numerical Example: Heat Equation

In the following section we provide numerical examples for the heat equation in dimension $D = 1, 2$. Specifically, we combine the analytic convergence and stability results from the

³ Although not shown, a similar result of $\Delta t < \min\{N^{-2}, 1.1\eta\}$ holds for a Fourier scheme.

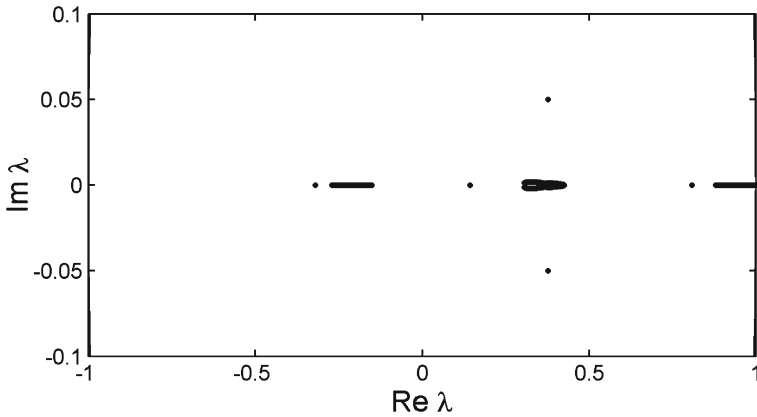


Fig. 4 Scatter plot of the numerical eigenvalues for (71). Here $N = 2048$, $\eta = 10^{-7}$ and Δt is taken from (72)

previous sections to show that one may achieve high order numerical schemes. As a starting point, we demonstrate high order convergence in $D = 1$ dimension. We then move to $D = 2$, and outline additional details that arise from the numerical construction of the extension $\tilde{g}(\mathbf{x})$.

6.1 1D Heat Equation

To test the convergence rates for the penalized heat equation, we use a manufactured solution approach. We note that the forced heat equation on $x \in [0, 2\pi]$,

$$\partial_t u = u_{xx} + f \tag{73}$$

$$f = e^{\sin(x+t)} [\cos(x+t) + \sin(x+t) - \cos^2(x+t)] \tag{74}$$

$$u(x, 0) = e^{\sin(x)} \tag{75}$$

has an exact solution $u_e = e^{\sin(x+t)}$. To quantify the total error, we penalize equation (73) as

$$\partial_t u_\eta = (u_\eta)_{xx} + f - \eta^{-1} \chi_s (u_\eta - \tilde{g}) \tag{76}$$

and compare the numerical solution of (76) to the exact one from (73).⁴

To discretize in space, we use an equispaced grid with fourth order stencils for all derivatives. In addition, we treat all terms explicitly in time with a second order (improved) Euler scheme. When constructing the extension \tilde{g} , we first compute the derivatives of u at each grid point, i.e. $u_x(x_k)$ or $u_{xx}(x_k)$. We then interpolate the values of u_x and u_{xx} from the regular grid points to the points x_Γ on the interface.

Remark 7 The solution to the penalized heat equation u has a discontinuous second derivative u_{xx} across the interface. As a result, interpolating u_{xx} using regular grid points on both sides of the interface will produce a weighted average of right and left derivatives $u_{xx}(0^-)$ and $u_{xx}(0^+)$ in the construction of \tilde{g} . We note that in practice, such a procedure does not appear to alter the final numerical convergence rate. ♠

For our tests, we choose a solid region centered at π to be $\Omega_s = [\pi - 0.7, \pi + 0.7]$. To satisfy the stability restriction, we then take $\Delta t = 0.2 h^2$, $h = 2\pi/N$ and slave $\eta = 5 \Delta t$ so

⁴ One can also restrict the forcing $\tilde{f} = f(1 - \chi_s)$ to the physical domain, and obtain similar results.

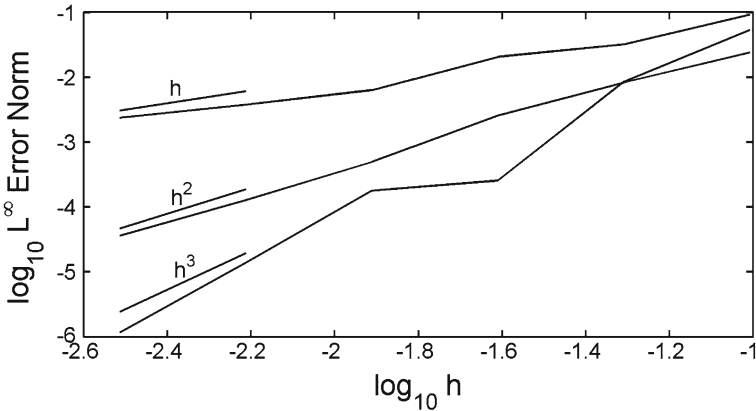


Fig. 5 Plot of numerical errors $\|u - u_{\eta,num}\|_{\infty}$ for different values of N . The three curves correspond to building \tilde{g} using 0, 1, 2 derivatives of u and result with convergence rates of 1, 2, 3, respectively

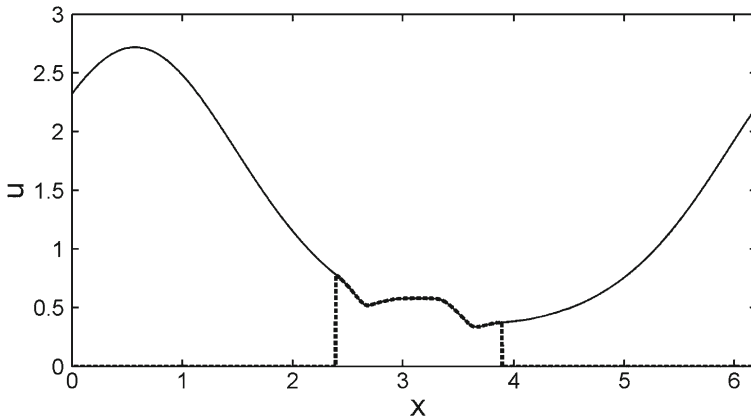


Fig. 6 Plot of the numerical solution $u_{\eta,num}$ (thin line) with the extension \tilde{g} (dashed line) for $N = 2048$. Here $\Omega_s = [\pi - 0.7, \pi + 0.7]$ is the solid domain

that all parameter values are fixed by the number of grid points N . For each $N = 2^k$, with $6 \leq k \leq 12$, we then numerically integrate (76) up to a final time of $T = 1$. We repeat the procedure using 0, 1 and 2 derivatives of u in constructing the extension \tilde{g} and compare the numerical solution to the exact one (i.e. that of the unpenalized problem). Here Fig. 5 shows the convergence rates for matching different derivatives, while Figs. 6 and 7 show a typical solution and the corresponding error, respectively.

6.2 2D Heat Equation

In the following subsection, we outline the numerical details for a $D = 2$ scheme. Here we work with an equispaced, regular grid with $N \times N$ points ($64 \leq N \leq 512$), and immerse the boundary Γ . The main difference when moving to higher dimensions is how one computes the extension $\tilde{g}(\mathbf{x})$. To illustrate the construction, we refer to Fig. 8. To build $\tilde{g}(\mathbf{x})$ we first compute all appropriate derivatives at each grid point (both on Ω_s and Ω_p). For each grid point $\mathbf{x} \in \Omega_s$ within a distance l of Γ , we compute $\xi(\mathbf{x})$ as the orthogonal projection of \mathbf{x} onto Γ and $s(\mathbf{x}) = \|\xi(\mathbf{x}) - \mathbf{x}\|_2$. Using a regular 9 point stencil, we then perform a polynomial

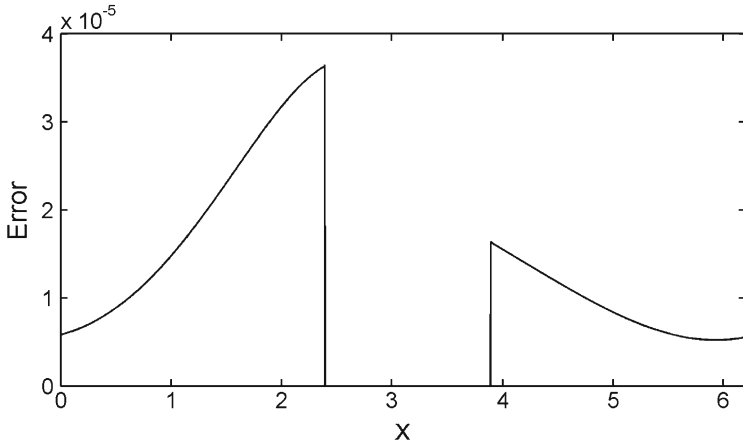


Fig. 7 Plot of the total error in solving the penalized equations matching 1 derivative of u in the extension. Here, $N = 2048$, $T = 1$ is the integration time, while $\Omega_s = [\pi - 0.7, \pi + 0.7]$ is the solid domain

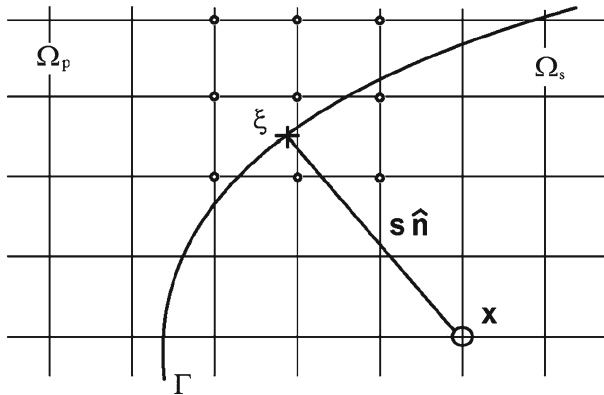


Fig. 8 Regular grid with interpolation points

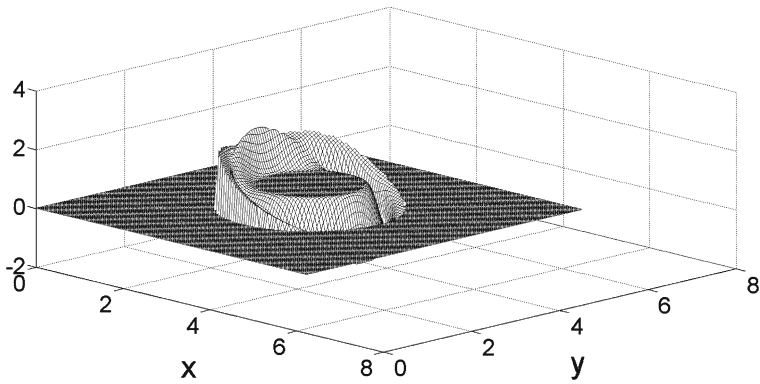


Fig. 9 A sample 2D plot of \tilde{g} matching 2 normal derivatives of $u(x)$. The plot is taken at $t = 0$ for the heat equation tests

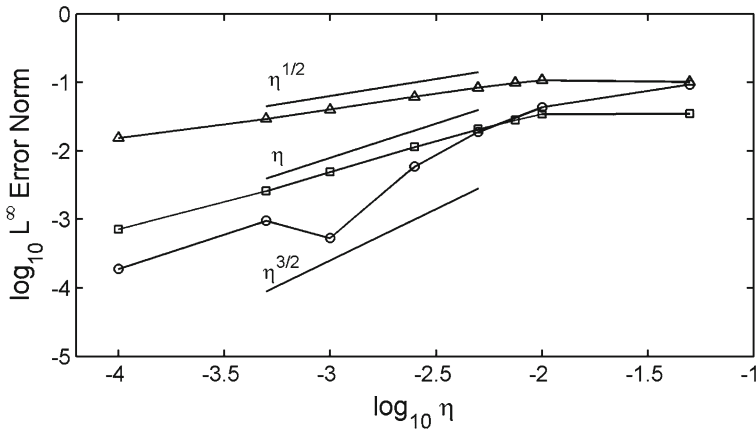


Fig. 10 Convergence plot of the L^∞ error at $T = 0.1$ for the heat equation tests. The plot shows curves for matching 0 derivatives (triangles), 1 derivative (squares), and 2 derivatives (circles). The straight lines compare the expected convergence rates of $O(\eta^{0.5})$, $O(\eta)$ and $O(\eta^{1.5})$, respectively

interpolation of all required derivatives from the grid points to ξ . Using the interpolated derivatives at ξ , one can then compute the normal derivatives of \mathbf{u} required in Eq. (24) to construct $\tilde{\mathbf{g}}(\mathbf{x})$ at each grid-point inside Ω_s . Figure 9 illustrates a typical construction of $\tilde{\mathbf{g}}(\mathbf{x})$.

Remark 8 For computational efficiency, one can precompute and store the values of ξ as well as the appropriate coefficients required to extrapolate derivatives to the interface Γ . ♠

As an example in $D = 2$, we take the computational domain Ω to be a periodic square with side length 2π . For the penalized domain Ω_s , we take a circle of radius $r = 1/2$ and center $(x_c, y_c) = (\pi, \pi)$. The physical domain is then $\Omega_p = \Omega \setminus \Omega_s$. To perform convergence tests, we again use a manufactured solution where $u_e = [e^{\sin(x)} + \cos(y)] \cos(t)$. Here we perform a convergence test for the penalty parameter η . To compute the convergence rate, we fix $N = 512$ and vary $5 \times 10^{-5} \leq \eta \leq 10^{-1}$, so that discrete numerical errors are smaller than the η -dependent error obtained by introducing the penalty term. For different values of η , we then integrate the penalty equation for a time $T = 0.1$ and compute the error. Figure 10 shows the L^∞ error between the penalized equation and the exact heat equation as a function of η .

7 Numerical Example: 2D Incompressible Navier–Stokes

The primary difficulty when transitioning from a penalized heat equation to the penalized incompressible Navier–Stokes equations is the addition of the velocity divergence constraint. Other differences, such as moving from a scalar to a vector equation, or adding a nonlinear convective term do not pose new additional challenges to the penalized equations. Intuitively, the difficulty with the divergence can be outlined as follows. For the penalized heat equation, the active penalty term forces the function u to closely track the extension function \tilde{g} . When moving to a set of vector equations, the velocity vector \mathbf{u}_η will closely track the term $\tilde{\mathbf{g}}$ inside the penalty region Ω_s . However, the component-wise construction of $\tilde{\mathbf{g}}$ will in general be such that $\nabla \cdot \tilde{\mathbf{g}} \neq 0$. Consequently, to remain consistent, one should not force $\nabla \cdot \mathbf{u}_\eta = 0$ inside Ω_s but rather allow $\nabla \cdot \mathbf{u}_\eta$ to loosely track $\nabla \cdot \tilde{\mathbf{g}}$.

One approach for handling the divergence constraint is to replace $\nabla \cdot \mathbf{u}_\eta = 0$ with a Pressure Poisson equation (PPE) [15, 16, 28, 29]. Such an approach can provide a consistent method

to compute the pressure and obtain high order schemes. Since a PPE approach requires the additional solution of a Poisson equation with Neumann boundary conditions, we defer the implementation to future work. In our case, we utilize a projection method where we project the velocity divergence to zero inside the fluid domain. We now discretize equations (10)–(11) in time.

7.1 Discretization in Time

Here we outline a pseudo-spectral scheme for solving the Navier–Stokes equations. For a second order scheme in h , we take a first order discretization in time with a time step restriction of the form outlined in (72). Since the domain is 2π periodic, we can use the Fourier transform to invert the Poisson equation. In the following algorithm we take a regular $N \times N$ grid. We also denote the discrete Fourier transform by \mathcal{F} so that $\hat{p}^n(\mathbf{k}) = \mathcal{F}[p^n]$ with $\mathbf{k} = (k_x, k_y)$ and $k = |\mathbf{k}|$.

Algorithm 1 (Navier–Stokes)

1. Given the velocity \mathbf{u}_η^n , compute an intermediate velocity $\tilde{\mathbf{u}}_\eta^{n+1}$

$$\frac{\tilde{\mathbf{u}}_\eta^{n+1} - \mathbf{u}_\eta^n}{\Delta t} = \mathbf{F}^n - \frac{1}{\eta} \chi_s(\mathbf{x})(\mathbf{u}_\eta^n - \tilde{\mathbf{g}}^n) \tag{77}$$

$$\mathbf{F}^n = -\mathbf{u}_\eta^n \cdot \nabla \mathbf{u}_\eta^n + \mu \Delta \mathbf{u}_\eta^n + \mathbf{f}^n \tag{78}$$

2. Compute the pressure

$$\Delta p_\eta^{n+1} = \frac{1}{\Delta t} (\nabla \cdot \tilde{\mathbf{u}}_\eta^{n+1})(1 - \chi_s) - \mathcal{A}. \tag{79}$$

For $k = 0$ set $\hat{p}_\eta^{n+1}(0) = 0$, while for $k \neq 0$ take

$$\hat{p}_\eta^n(\mathbf{k}) = -\frac{1}{k^2} \mathcal{F}\left[\frac{1}{\Delta t} (\nabla \cdot \tilde{\mathbf{u}}_\eta^{n+1})(1 - \chi_s)\right]. \tag{80}$$

Note that \mathcal{A} does not appear in the Fourier transform and at no time does one ever compute \mathcal{A} . The value of \mathcal{A} is hidden as a consistency condition in setting $\hat{p}_\eta^{n+1}(0) = 0$.

3. Update the velocity \mathbf{u}_η^{n+1}

$$\mathbf{u}_\eta^{n+1} = \tilde{\mathbf{u}}_\eta^{n+1} - (\Delta t) \mathcal{F}^{-1}[-i\mathbf{k}\hat{p}_\eta^n]. \tag{81}$$

Note that either Fourier transforms or a second order finite difference scheme can be used when computing the derivatives $\partial_{x_j} \mathbf{u}_\eta$ in algorithm (1).

Since the second derivatives are discontinuous, we compute $\Delta \mathbf{u}_\eta$ using finite differences.

In the Poisson equation for the pressure, p_η is only determined up to a constant. To uniquely determine p_η we enforce $\int_\Omega p_\eta \, dV = 0$. Meanwhile the value of \mathcal{A} is chosen so

that the Poisson equation satisfies the standard solvability condition. Namely,

$$\mathcal{A} = \frac{\Delta t^{-1}}{V} \int_{\Omega_p} \nabla \cdot \tilde{\mathbf{u}}_\eta \, dV \tag{82}$$

$$= \frac{\Delta t^{-1}}{V} \int_\Gamma \tilde{\mathbf{u}}_\eta \cdot \mathbf{n} \, dA \tag{83}$$

$$= \frac{\Delta t^{-1}}{V} \int_\Gamma (\tilde{\mathbf{u}}_\eta - \mathbf{g}) \cdot \mathbf{n} \, dA \tag{84}$$

where in the last line we have used the fact that $\int_\Gamma \mathbf{g} \cdot \mathbf{n} = 0$. Here $V = \int_{\Omega_p} dV$ is the volume of the physical domain. The last line also shows that \mathcal{A} is not nearly as large as Δt^{-1} since the jump $(\mathbf{u}_\eta - \mathbf{g}) \cdot \mathbf{n}$ is expected to be small. Finally, we make a remark on the projection of \mathbf{u} . Inside Ω_p , we have

$$\nabla \cdot \mathbf{u} = \nabla \cdot \tilde{\mathbf{u}} - (\Delta t)\Delta p \tag{85}$$

$$= (\Delta t)\mathcal{A} \tag{86}$$

$$\leq C \max_{\mathbf{x} \in \Gamma} |\tilde{\mathbf{u}}_\eta - \mathbf{g}| \tag{87}$$

where C is an appropriate constant. Hence any error in the divergence of \mathbf{u} is directly controlled by the error in the velocity boundary condition. In particular, for matching 1 derivative, we expect $\nabla \cdot \mathbf{u} = O(\eta) + O(\Delta t)$ inside Ω_f . As a result, we can recover a second order scheme, however systematically moving to a higher order method will require an alternative formulation, such as a PPE scheme, for computing the pressure.

Remark 9 In order to guarantee second order spatial accuracy in algorithm (1), $\tilde{\mathbf{g}}$ should match at least 1 derivative of \mathbf{u}_η . ♠

Remark 10 One could also consider solving the Poisson equation (79) with $\mathcal{A} = 0$ and instead impose an interface condition on the normal pressure gradient:

$$[\mathbf{n} \cdot \nabla p_\eta]_\Gamma = \frac{1}{\Delta t} \mathbf{n} \cdot (\tilde{\mathbf{u}}_\eta - \mathbf{g}) \tag{88}$$

$$[f]_\Gamma := \lim_{\epsilon \rightarrow 0} (f(\mathbf{x} + \epsilon \mathbf{n}) - f(\mathbf{x} - \epsilon \mathbf{n})) \quad \mathbf{x} \in \Gamma. \tag{89}$$

In the definition (89), \mathbf{n} is taken as the unit normal directed outward from Ω_p . Such an approach greatly simplifies the analysis for the behavior of the divergence ($\nabla \cdot \mathbf{u}_\eta$) in the resulting PPE scheme. However, we note that numerically solving (79) with (88) and $\mathcal{A} = 0$ is harder than simply solving (79)–(82). Furthermore, the equations also allow for a direct solution using pseudo-spectral methods, while the interface problem does not. ♠

To test the order of accuracy of the active penalty method, we again use a manufactured solution of the form $\mathbf{u}_e = (u_e, v_e)$ and p_e where

$$u_e = \cos(x) \sin(y) \cos(t) \tag{90}$$

$$v_e = -\sin(x) \cos(y) \cos(t) \tag{91}$$

$$p_e = \sin(2x) \cos(y) \cos(t). \tag{92}$$

Given initial data corresponding to the exact solution, we numerically evolve the velocity \mathbf{u}_η and pressure p_η using the pseudo-spectral method outlined in algorithm 1. Here we match

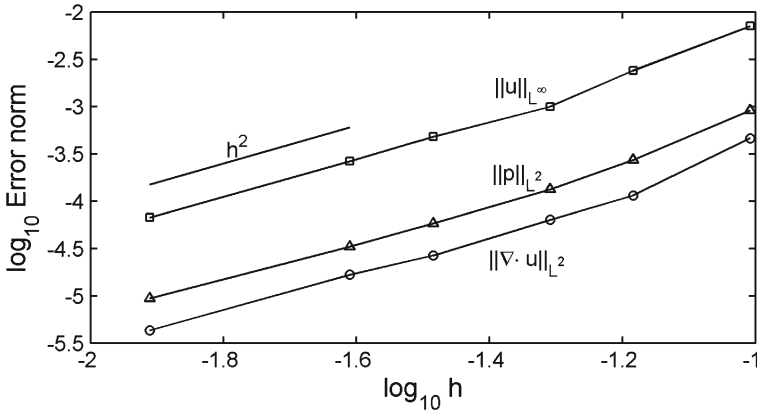


Fig. 11 Navier–Stokes convergence plot. Second order convergence in $L^\infty(\Omega_p)$ for the velocity field (squares), and in $L^2(\Omega_p)$ for the pressure (triangles) and velocity divergence (circles)

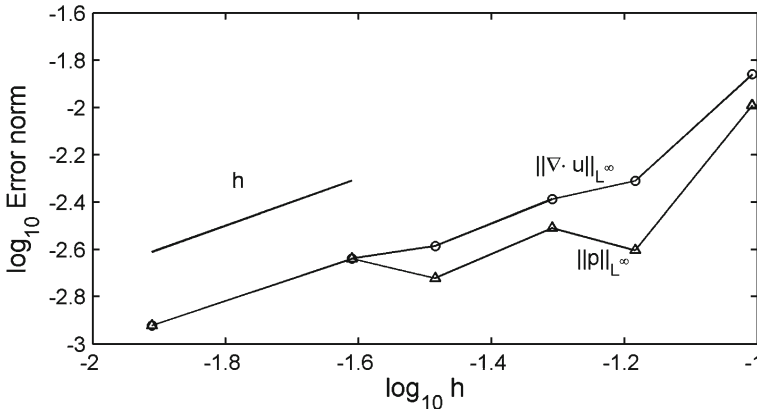


Fig. 12 Navier–Stokes convergence plot. First order convergence in $L^\infty(\Omega_p)$ for the pressure (triangles) and velocity divergence (circles). The weaker convergence in $L^\infty(\Omega_p)$ is due to the boundary layer in the pressure and divergence. The divergence was computed using second order finite differences

1 derivative of \mathbf{u}_η in the construction of $\tilde{\mathbf{g}}$ and take time steps, with the appropriate restriction, of $\Delta t = O(h^2) = O(\eta)$. Figure 11 shows second order convergence of the velocity field (in $L^\infty(\Omega_p)$), as well as the pressure and divergence (in $L^2(\Omega_p)$). Meanwhile, the pressure and the divergence converge at one order less in $L^\infty(\Omega_p)$ (Fig. 12). As an example, Fig. 13a, b show the typical error for velocity and pressure while Fig. 14a, b show the velocity divergence. In addition, Fig. 15a, b show the horizontal velocity field along with the horizontal component of the extension $\tilde{\mathbf{g}} \cdot \hat{\mathbf{x}}$. Note that \mathbf{u}_η is again very close to $\tilde{\mathbf{g}}$ inside Ω_s .

8 Flow Around an Impulsively Started Cylinder

In this section we test our method for the model problem of an impulsively started cylinder [20]. In this case, we solve the following initial value problem where the fluid starts at rest

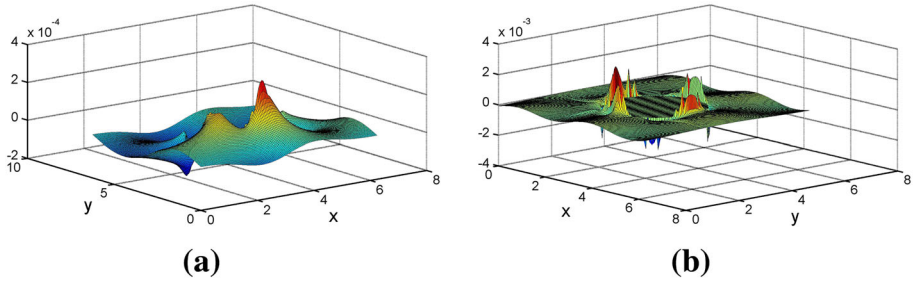


Fig. 13 Error fields with $N = 128$ for the velocity and pressure after $T = 1$. **a** Velocity error. **b** Pressure error

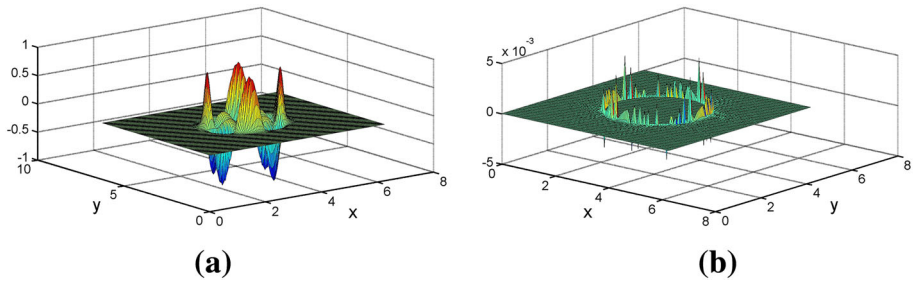


Fig. 14 Plots of the divergence $\nabla \cdot \mathbf{u}_\eta$ in Ω (left) and in Ω_p (right) with $N = 128$ after $T = 1$. The plot in Ω_p shows the $\|\nabla \cdot \mathbf{u}_\eta\|_{L^\infty(\Omega_p)}$ error occurs at a point in a boundary layer near Γ . **a** Full divergence. **b** Divergence error

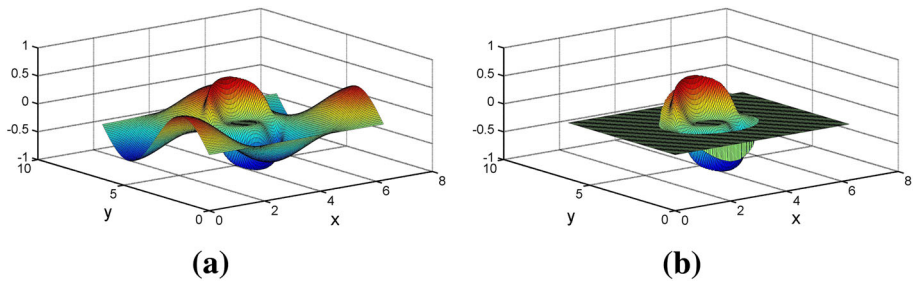


Fig. 15 The numerical velocity field for horizontal component $u_{\eta,Num}$ along with the extension function. Here $N = 128$ and $T = 1$. **a** Velocity $u_{\eta,Num}$. **b** The extension $\tilde{g} \cdot \hat{x}$

$$\mathbf{u}_\eta(\mathbf{x}, 0) = 0 \quad \text{for } \mathbf{x} \in \Omega. \tag{93}$$

The impulsively started cylinder is then modeled by a moving mask function with a time dependent set $\Omega_s(t)$ and the appropriate Dirichlet boundary condition. For $t > 0$ we have

$$\Omega_s(t) = \{\mathbf{x} : |\mathbf{x} - \mathbf{x}_0 - \mathbf{u}_0 t| \leq R\} \tag{94}$$

$$\mathbf{u} = \mathbf{u}_0 \quad \text{for } \mathbf{x} \in \Gamma. \tag{95}$$

Here $\mathbf{u}_0 = u_0 \hat{e}_x$ is the velocity of the cylinder, and (\mathbf{x}_0, R) denotes the center and radius of the cylinder.

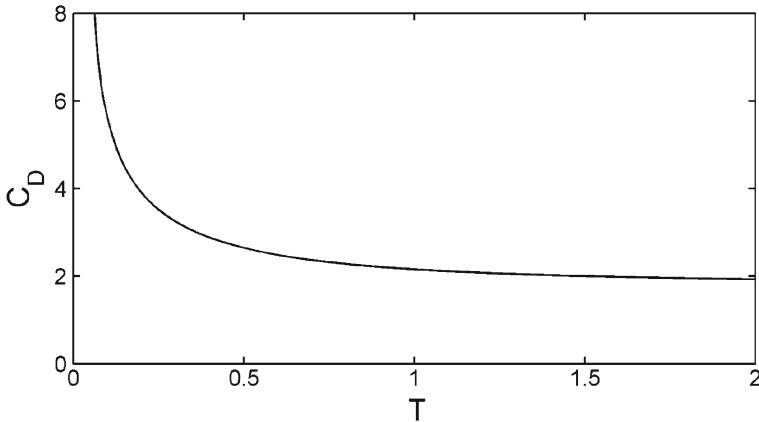


Fig. 16 Drag versus time for RE = 40. Here $\eta = 2 \times 10^{-4}$, $N = 512$, $l = 0.45$

To simplify the numerical calculation, we perform a Galilean transformation on the coordinates and solve the penalized equations with a stationary mask. The velocity field then solves the equation

$$\partial_t \mathbf{u}_\eta + (\mathbf{u}_\eta - \mathbf{u}_0) \cdot \nabla \mathbf{u}_\eta = -\nabla p_\eta + \mu \Delta \mathbf{u}_\eta + \mathbf{f} - \eta^{-1} \chi_s (\mathbf{u}_\eta - \mathbf{u}_0 - \tilde{\mathbf{g}}), \tag{96}$$

with initial data $\mathbf{u}(\mathbf{x}, 0) = 0$. Here $\chi_s(\mathbf{x})$ is a stationary mask with $\Omega_s(0)$, while $\tilde{\mathbf{g}}$ is the active penalty term with a zero boundary condition $\mathbf{g} = 0$.

To compare our results with pre-existing numerical tests, we adopt the following definition of the Reynolds number and time scales from [20]

$$RE = \frac{2Ru_0}{\mu} \tag{97}$$

$$T = \frac{u_0}{R} t. \tag{98}$$

Using Eq. (96), we then solve for the velocity field in time, and compute the drag force and lift for the impulsively started cylinder. To compute the force we numerically evaluate the momentum transfer to the fluid

$$\mathbf{F}_b = -\frac{d}{dt} \int_{\Omega_f} \mathbf{u} dV \tag{99}$$

$$= -\frac{d}{dt} \int_{\Omega} \mathbf{u} (1 - \chi_s(\mathbf{x})) dV. \tag{100}$$

The lift (C_L) and drag (C_D) coefficients are then evaluated as the non-dimensionalized components of the force

$$C_D = \frac{\mathbf{F}_b \cdot \hat{\mathbf{e}}_x}{Ru_0^2} \quad C_L = \frac{\mathbf{F}_b \cdot \hat{\mathbf{e}}_y}{Ru_0^2}. \tag{101}$$

In our numerical tests, we examine the impulsively started cylinder for RE = 40 and RE = 550. In both cases, we use $R = 1$, $u_0 = 10$ and the appropriate values of μ to obtain RE. Here Figs. 16 and 17 show the drag versus time for an impulsively started cylinder with RE

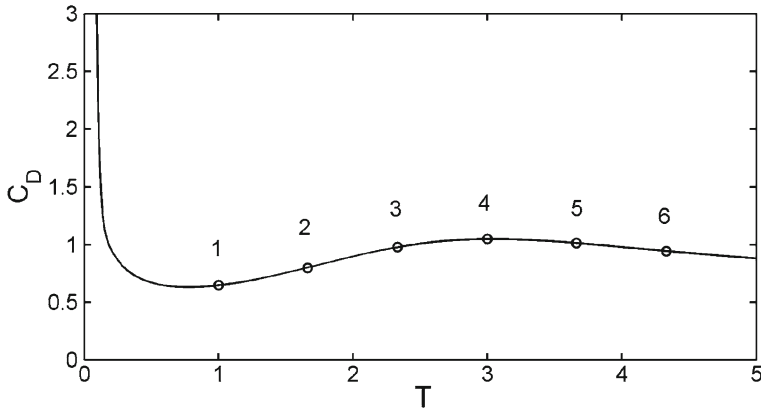


Fig. 17 Drag versus time for $RE = 550$. Here $\eta = 10^{-3}$, $N = 768$, $l = 0.05$. Circles correspond to snapshots of the vorticity shown in Fig. 18

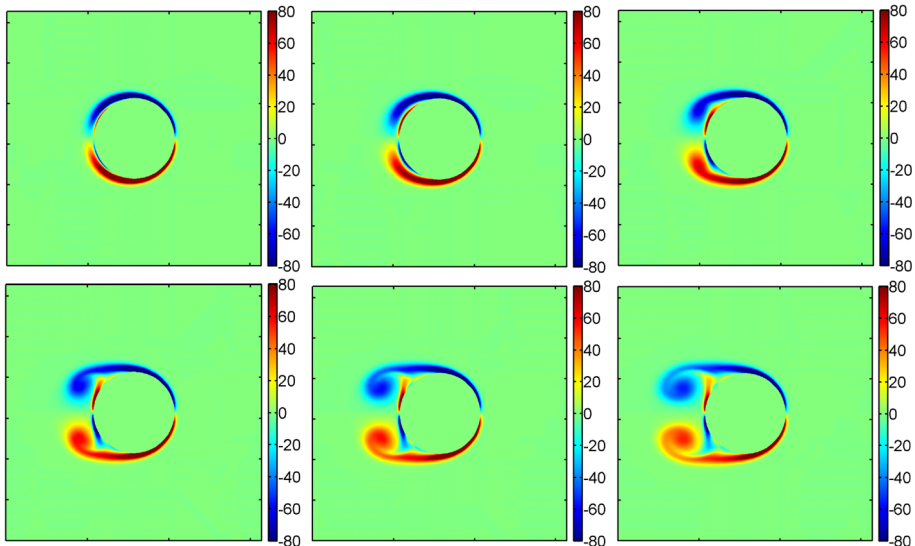


Fig. 18 Snapshots of the vorticity for an impulsively started cylinder with $RE = 550$. Images are taken at times $(1 - r) T = 1, 1.66, 2.33, 3, 3.66, 4.33$ and correspond to the circles in Fig. 17

$= 40$ and $RE = 550$, respectively. Note that qualitatively the curves match the benchmark results from [20]. In particular, for the $RE = 40$, the drag coefficient monotonically decays to a value slightly below 2. Meanwhile, for $RE = 550$, the drag first drops, followed by a peak at $T = 3.05$. Here Fig. 18 shows the early development of vorticity for the impulsive cylinder. We also extend the computation for a much longer time to verify the onset of vortex shedding. Here Fig. 19 shows the oscillations in the lift coefficient versus time, while Fig. 20 shows the vorticity at various times in the evolution. We note that due to the periodicity of the domain, the simulation effectively models an array of cylinders, as opposed to the conventional von Kármán street which arises from flow past one cylinder.

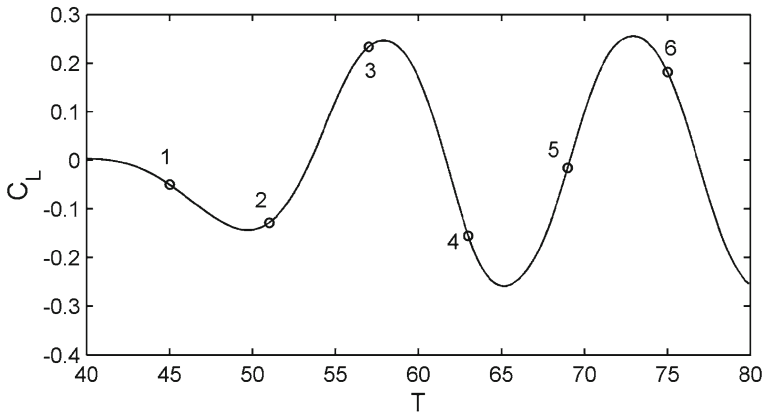


Fig. 19 Lift versus time for the onset of the von Kármán street at $RE = 550$. The oscillations correspond to vortex shedding. Circles correspond to snapshots of the vorticity shown in Fig. 20

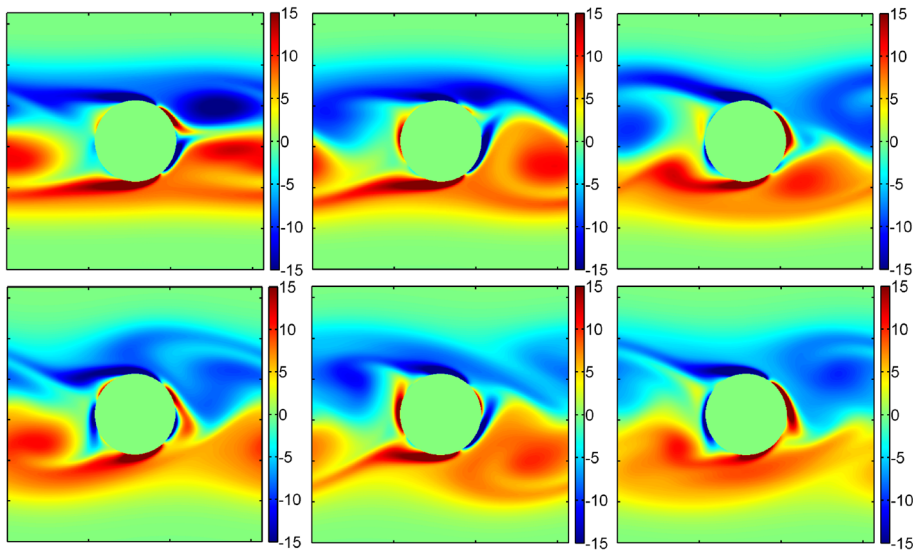


Fig. 20 Snapshots of the vorticity for an impulsively started cylinder showing the onset of full vortex shedding and von Kármán type street [30]. Images are for $RE = 550$, and taken at times $T = 45, 51, 57, 63, 69, 75$ corresponding to the circles in Fig. 19

9 Conclusion

In this paper, we outline how to construct high order penalty methods. We do so by first introducing an active penalty term for the heat equation. When we increase the number of matched derivatives, we show that the penalty term improves the analytic convergence rate in terms of the penalty parameter. Secondly, we examine the numerical stability of the active penalty term. We show that it does not introduce additional stiffness into the equations or additional length scales that would need to be resolved. The combination of the high order convergence in the penalty parameter along with the numerical stability then leads to

higher order numerical schemes. Lastly, we extend the penalized term from the heat equation to the incompressible Navier–Stokes equations. In particular, we show how to handle the divergence constraint on the velocity field. We also conclude with an application of flow around an impulsively started cylinder for $RE = 40$ and $RE = 550$. In the case of $RE = 550$, we demonstrate the onset of a von Kármán street.

Although we have outlined a high order approach, there are still remaining issues that limit the practical feasibility of the method. For instance, at no point do we improve the smoothness of the solution \mathbf{u}_η . In fact the second derivatives of \mathbf{u}_η remain discontinuous across the curve Γ , although matching more derivatives in the active penalty term may reduce the size of the discontinuity. As a result, Fourier methods still have a slow decay in the Fourier modes thereby limiting the ability to spectrally compute derivatives. In addition, interpolation of high order derivatives in the construction of $\tilde{\mathbf{g}}$ should be one-sided (i.e. from Ω_p) while in practice one would prefer to use points on both sides of Γ . As a result, ongoing research includes improving the global smoothness of \mathbf{u}_η while retaining the high order convergence.

Acknowledgments The authors would like to thank Kirill Shmakov and Geneviève Bourgeois for additional preliminary computations not currently presented. The authors have also greatly benefited from conversations with Dmitry Kolomenskiy, Kai Schneider, Ruben Rosales and Tsogtgerel Gantumur. This work was supported by an NSERC Discovery Grant and the NSERC DAS.

References

1. Angot, P.: A unified fictitious domain model for general embedded boundary conditions. *C. R. Math. Acad. Sci. Paris* **341**, 683–688 (2005)
2. Angot, P.: A fictitious domain model for the stokes/brinkman problem with jump embedded boundary conditions. *C. R. Math. Acad. Sci. Paris* **348**, 697–702 (2010)
3. Angot, P., Bruneau, C.-H., Fabrie, P.: A penalization method to take into account obstacles in incompressible viscous flows. *Numer. Math.* **81**, 497–520 (1999)
4. Angot, P., Caltagirone, J.-P.: New graphical and computational architecture concept for numerical simulation on supercomputers. In: *Proceedings of 2nd World Congress on Computational Mechanics*, vol. 1, pp. 973–976 (1990)
5. Arquis, E., Caltagirone, J.-P.: Sur les conditions hydrodynamiques au voisinage d’une interface milieu fluide—milieux poreux: application à la convection naturelle. *Comptes Rendus de l’Academie des Science Paris* **II**(299), 1–4 (1984)
6. Bruneau, C.-H.: Boundary conditions on artificial frontiers for incompressible and compressible Navier–Stokes equations. *M2AN Math. Model. Numer. Anal.* **34**, 303–314 (2000)
7. Bruneau, C.-H., Fabrie, P.: New efficient boundary conditions for incompressible navier-stokes equations: a well-posedness result. *M2AN Math. Model. Numer. Anal.* **30**, 815–840 (1996)
8. Carbou, G., Fabrie, P.: Boundary layer for a penalization method for viscous incompressible flow. *Adv. Diff. Equat.* **8**(12), 1409–1532 (2003)
9. Chantalat, F., Bruneau, C.-H., Galusinski, C., Iollo, A.: Level-set, penalization and cartesian meshes: a paradigm for inverse problems and optimal design. *J. Comput. Phys.* **228**, 6291–6315 (2009)
10. Coquerelle, M., Cottet, G.-H.: A vortex level set method for the two-way coupling of an incompressible fluid with colliding rigid bodies. *J. Comput. Phys.* **227**, 9121–9137 (2008)
11. Fedkiw, R., Aslam, T., Merriman, B., Osher, S.: A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method). *J. Comput. Phys.* **152**, 457–492 (1999)
12. Folland, G.: *Introduction to Partial Differential Equations*, 2d ed. Princeton University Press, Princeton (1995)
13. Gibou, F., Chen, L., Nguyen, D., Banerjee, S.: A level set based sharp interface method for the multiphase incompressible Navier–Stokes equations with phase change. *J. Comput. Phys.* **222**, 536–555 (2007)
14. Gibou, F., Fedkiw, R.: A Fourth order accurate discretization for the Laplace and Heat equations on arbitrary domains with applications to the Stefan problem. *J. Comput. Phys.* **202**, 577–601 (2005)
15. Henshaw, William D.: A fourth-order accurate method for the incompressible Navier–Stokes equations on overlapping grids. *J. Comput. Phys.* **113**(1), 13–25 (July 1994)

16. Johnston, H., Liu, J.-G.: Accurate, stable and efficient navier-stokes solvers based on explicit treatment of the pressure term. *J. Comput. Phys.* **199**, 221–259 (2004)
17. Kadoch, B., Kolomenskiy, D., Angot, P., Schneider, K.: A volume penalization method for incompressible flows and scalar advection-diffusion with moving obstacles. *J. Comput. Phys.* **231**, 4365–4383 (2012)
18. Khadra, K., Angot, P., Parneix, S., Caltagirone, J.P.: Fictitious domain approach for numerical modelling of Navier–Stokes equations. *Int. J. Numer. Methods Fluids* **34**, 651–684 (2000)
19. Kolomenskiy, D., Schneider, K.: A fourier spectral method for the Navier–Stokes equations with volume penalization for moving solid obstacles. *J. Comput. Phys.* **228**, 5687–5709 (2009)
20. Koumoutsakos, P., Leonard, A.: High-resolution simulations of the flow around an impulsively started cylinder using vortex methods. *J. Fluid Mech.* **296**, 1–38 (1995)
21. Le, B., Khoo, B., Peraire, J.: An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries. *J. Comput. Phys.* **220**, 109–138 (2006)
22. Morales, J., Leroy, M., Bos, W., Schneider, K.: Simulation of confined magnetohydrodynamic flows using a pseudo-spectral method with volume penalization. *J. Comput. Phys.* (under review), hal-00719737, version 1. (2012)
23. Ng, Y.T., Min, C., Gibou, F.: An efficient fluid–solid coupling algorithm for single-phase flows. *J. Comput. Phys.* **228**, 8807–8829 (2009)
24. Peskin, C.: The immersed boundary method. *Acta Numerica* **11**, 479–517 (2000)
25. Ramière, I., Angot, P., Belliard, M.: A general fictitious domain method with immersed jumps and multilevel nested structured meshes. *J. Comput. Phys.* **225**, 1347–1387 (2007)
26. Sarthou, A., Vincent, S., Caltagirone, J.P.: Consistent velocity-pressure for second-order L^2 -penalty and direct-forcing methods, hal-00592079, Version 1. (2011)
27. Sarthou, A., Vincent, S., Caltagirone, J.P., Angot, P.: Eulerian–Lagrangian grid coupling and penalty methods for the simulation of multiphase flows interacting with complex objects. *Int. J. Numer. Methods Fluids* **56**, 1093–1099 (2008)
28. Shirokoff, D.I.: A Pressure Poisson Method for the Incompressible Navier–Stokes Equations: II. Long Time Behavior of the Klein–Gordon Equations. PhD thesis, Massachusetts Institute of Technology (2011)
29. Shirokoff, D., Rosales, R.R.: An efficient method for the incompressible Navier–Stokes equations on irregular domains with no-slip boundary conditions, high order up to the boundary. *J. Comput. Phys.* **230**, 8619–8646 (2011)
30. von Kármán, T.: *Aerodynamics*. McGraw-Hill, New York (1963)