

On efficient multigrid solvers for mixed-degree local discontinuous Galerkin multiphase Stokes problems

Robert I. Saye^{ID}

Lawrence Berkeley National Laboratory, Berkeley, CA, USA

ARTICLE INFO

MSC:

primary 65N55
65N30
65N22
76D07
65F10

Keywords:

Multigrid
Stokes
Local discontinuous Galerkin
Mixed-degree
Multiphase
Local Fourier analysis

ABSTRACT

We design and investigate efficient multigrid solvers for multiphase Stokes problems discretised via mixed-degree local discontinuous Galerkin methods. Using the template of a standard multigrid V-cycle, we develop a smoother analogous to element-wise block Gauss–Seidel, except the diagonal block inverses are replaced with an approximation that balances the smoothing of the velocity and pressure variables, factoring in the unequal scaling of the various Stokes system operators, and optimised via two-grid local Fourier analysis. We evaluate the performance of the multigrid solver across an extensive range of two- and three-dimensional test problems, including steady-state and unsteady, standard-form and stress-form, single-phase and high-contrast multiphase Stokes problems, with multiple kinds of boundary conditions and various choices of polynomial degree. In the lowest-degree case, i.e., that of piecewise constant pressure fields, we observe reliable multigrid convergence rates, though not especially fast. However, in every other case, we see rapid convergence rates matching those of classical Poisson-style geometric multigrid methods; e.g., 5 iterations reduce the Stokes system residual by 5 to 10 orders of magnitude.

1. Introduction

In this work, we design and investigate efficient multigrid solvers for multiphase Stokes problems discretised via mixed-degree local discontinuous Galerkin (LDG) methods. Our motivation stems in part from prior work on multigrid Stokes solvers for equal-degree LDG methods, i.e., those for which the velocity and pressure fields have the same polynomial degree [1,2]. In the equal-degree setting, and for various kinds of Stokes problems, one can build highly effective multigrid solvers via simple multigrid V-cycles and element-wise block Gauss–Seidel smoothers; e.g., to reduce the residual by 8 orders of magnitude, only 6 to 8 solver iterations are needed. However, these multigrid solvers crucially depend on a carefully-tuned mechanism of pressure penalty stabilisation, but for some kinds of Stokes problems—notably, those with prescribed stress boundary conditions, or multiphase problems with interfacial stress jump conditions—there is no good way to implement this mechanism (as discussed further in Section 2.1).

In contrast, a mixed-degree LDG framework does not require any pressure penalty stabilisation, can solve the aforementioned Stokes problems, and, perhaps counterintuitively, can yield significantly more accurate numerical solutions. Here, mixed-degree refers to the piecewise polynomial pressure field being of one degree less than the velocity field. Unfortunately, however, the above-mentioned multigrid solvers are ineffective in a mixed-degree setting—for example, on an elementary Stokes problem, a simple block Gauss–Seidel multigrid solver requires 6 iterations for a 10^8 residual reduction in the equal-degree setting, but fails in the mixed-degree setting unless the smoother is damped; even with optimal damping, convergence is slow and degrades as the mesh is refined (see Appendix C). Consequently, the main aim of this study is to develop a replacement multigrid solver, specifically by targeting the multigrid smoother.

E-mail address: rsaye@lbl.gov.

<https://doi.org/10.1016/j.rinam.2026.100784>

Received 24 June 2026; Received in revised form 11 September 2026; Accepted 8 October 2026

Available online 17 October 2026

2590-0374/© 2026 The Author. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

One possibility is to apply sparse approximate inverses (SAI) methods, as explored in [2]. These techniques have a number of potential benefits, including high parallelism, increased robustness (e.g., in handling a wider range of discretisation parameters), and the potential for black-box adaptivity (e.g., to handle anisotropy, highly unstructured meshes, and complex geometry, etc.). SAI approaches can lead to rapid multigrid convergence; on the other hand, they can be relatively expensive to build and apply. In this work, we leverage some of the techniques developed in [2], but with the aim of devising a smoother whose application cost is the same as block Gauss–Seidel. In essence, the developed smoother is identical to multicoloured block Gauss–Seidel, except the diagonal block inverses are replaced with an approximation that optimally balances the various Stokes system operators (i.e., the viscous, pressure gradient, velocity divergence, and density-weighted temporal derivative operators). Given its similarity to element-wise block Gauss–Seidel, one may also view the smoother as a multiplicative Schwarz method, whereby the subdomains of the Schwarz method correspond to the combined set of degrees of freedom on each mesh element; likewise, the smoother shares various operational similarities with that of multiplicative Vanka-type methods [3–5] and box relaxation methods [6], though differs in how the smoother is constructed.

The paper is structured as follows. In Section 2, we outline the prototype Stokes problem and summarise the mixed-degree LDG framework; we also include numerical examples and convergence tests to compare against the equal-degree setting. In Section 3, we develop the multigrid framework, starting with basic ingredients of the V-cycle and then focusing on the smoother, including: (i) its construction, (ii) balancing methods to treat the unequal scaling of the Stokes system operators, (iii) adaptation to unsteady Stokes problems, (iv) overall assembly and application of the smoother, and (v) optimisation of its parameters. Numerical results are then presented in Section 4, examining multigrid performance across a variety of test problems, including 2D and 3D steady-state and unsteady Stokes problems in standard-form and stress-form, different kinds of boundary conditions, and high-contrast multiphase problems. A concluding perspective is given in Section 5, while additional details of the LDG framework and smoother parameter optimisation are given in Appendices A and B, respectively.

2. LDG scheme

Our prototype (steady-state) Stokes problem consists of solving for a velocity field $\mathbf{u} : \Omega \rightarrow \mathbb{R}^d$ and pressure field $p : \Omega \rightarrow \mathbb{R}$ such that

$$\left. \begin{aligned} -\nabla \cdot (\mu_i(\nabla \mathbf{u} + \gamma \nabla \mathbf{u}^\top)) + \nabla p &= \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= f \end{aligned} \right\} \text{ in } \Omega_i, \quad (1)$$

subject to the interfacial jump conditions

$$\left. \begin{aligned} \llbracket \mathbf{u} \rrbracket &= \mathbf{g}_{ij} \\ \llbracket \mu(\nabla \mathbf{u} + \gamma \nabla \mathbf{u}^\top) \cdot \mathbf{n} - p\mathbf{n} \rrbracket &= \mathbf{h}_{ij} \end{aligned} \right\} \text{ on } \Gamma_{ij}, \quad (2)$$

and boundary conditions

$$\left. \begin{aligned} \mathbf{u} &= \mathbf{g}_\partial & \text{on } \Gamma_D, \\ \mu(\nabla \mathbf{u} + \gamma \nabla \mathbf{u}^\top) \cdot \mathbf{n} - p\mathbf{n} &= \mathbf{h}_\partial & \text{on } \Gamma_N, \end{aligned} \right\} \quad (3)$$

where Ω is a domain in $\mathbb{R}^d$ divided into one or more subdomains/phases Ω_i , $\mu_i > 0$ is a phase-dependent viscosity coefficient, $\Gamma_{ij} := \partial\Omega_i \cap \partial\Omega_j$ is the interface between phase i and j , and Γ_D and Γ_N denote the parts of $\partial\Omega$ upon which velocity Dirichlet or stress boundary conditions are prescribed, respectively. Here, $\gamma \in \{0, 1\}$ is a parameter defining the form of the Stokes equations: they are said to be in *standard form* when $\gamma = 0$ and *stress form* when $\gamma = 1$. Meanwhile, $\mathbf{f}$, f , $\mathbf{g}$, and $\mathbf{h}$ provide the data for the multiphase Stokes problem and are given functions defined on Ω , its boundary, and internal interfaces.

We consider here meshes arising from Cartesian grids, quadrees or octrees; accordingly, a tensor-product polynomial space is adopted. Let $\mathcal{E} = \bigcup_i E_i$ denote the mesh and its elements, let $\wp \geq 0$ be an integer, and let $\mathcal{Q}_\wp$ be the space of tensor-product polynomials of one-dimensional degree $\wp$; e.g., $\mathcal{Q}_2$ is the space of biquadratic or triquadratic polynomials in 2D or 3D, respectively. Define the piecewise polynomial space

$$V_\wp := \{u : \Omega \rightarrow \mathbb{R} \mid u|_E \in \mathcal{Q}_\wp \text{ for every } E \in \mathcal{E}\}.$$

Note that an arbitrary function of $V_\wp$ has $(\wp+1)^d$ degrees of freedom per element. We similarly define $V_\wp^d$ and $V_\wp^{d \times d}$ to be the space of piecewise polynomial vector-valued and matrix-valued fields, respectively.

The LDG framework considered here solves Eqs. (1)–(3) for a discrete solution $\mathbf{u}_h \in V_\wp^d$ and $p_h \in V_{\wp-1}$ according to a five-step process: (i) using viscosity-upwinded one-sided numerical fluxes, define a discrete gradient $\boldsymbol{\tau}_h \in V_\wp^{d \times d}$ equal to the discretisation of $\nabla \mathbf{u}_h$, taking into account Dirichlet source data $\mathbf{g}$; (ii) using $\boldsymbol{\tau}_h$, define a viscous stress tensor $\boldsymbol{\sigma}_h \in V_\wp^{d \times d}$ equal to the L^2 projection of $\mu(\boldsymbol{\tau}_h + \gamma \boldsymbol{\tau}_h^\top) - p_h \mathbb{I}$ onto $V_\wp^{d \times d}$; (iii) using the adjoint of the discrete gradient operator from the first step, compute the negative discrete divergence of $\boldsymbol{\sigma}_h$, taking into account Neumann-like data $\mathbf{h}$, and set the result equal to the L^2 projection of $\mathbf{f}$ onto $V_\wp^d$; (iv) compute a mixed-degree negative discrete divergence of $\mathbf{u}_h$, taking into account Dirichlet data $\mathbf{g} \cdot \mathbf{n}$, and set the result equal to the L^2 projection of f onto $V_{\wp-1}$; last, (v) amend the overall system by appropriately including penalty stabilisation terms. Our construction is partly based on the LDG schemes initially developed by Cockburn et al. [7], with extensions to multiphase variable-viscosity problems

analogous to the techniques developed in prior work [1]; details are deferred to [Appendix A](#). The outcome is a symmetric linear system of the following form (here illustrated for the 3D case, with that of the 2D case being an obvious reduction),

$$\begin{pmatrix} \mathcal{A}_{11} & \mathcal{A}_{12} & \mathcal{A}_{13} & -\tilde{G}_1^T \bar{M} \\ \mathcal{A}_{21} & \mathcal{A}_{22} & \mathcal{A}_{23} & -\tilde{G}_2^T \bar{M} \\ \mathcal{A}_{31} & \mathcal{A}_{32} & \mathcal{A}_{33} & -\tilde{G}_3^T \bar{M} \\ -\bar{M} \tilde{G}_1 & -\bar{M} \tilde{G}_2 & -\bar{M} \tilde{G}_3 & 0 \end{pmatrix} \begin{pmatrix} u_h \\ v_h \\ w_h \\ p_h \end{pmatrix} = \begin{pmatrix} b_u \\ b_v \\ b_w \\ b_p \end{pmatrix}, \quad (4)$$

where $u_h, v_h, w_h \in V_\varphi$ denote the components of $\mathbf{u}_h$, while (b_u, b_v, b_w, b_p) collects the entire influence of the source data $\mathbf{f}$, f , $\mathbf{g}$, and $\mathbf{h}$ onto the right-hand side.¹ Here, $\tilde{G} : V_\varphi \rightarrow V_{\varphi-1}^d$ is a mixed-degree discrete gradient operator, $\bar{M}$ is a block-diagonal mass matrix relative to $V_{\varphi-1}$, and $\mathcal{A} : V_\varphi^d \rightarrow V_\varphi^d$ implements the viscous part of the Stokes momentum equations, whose (i, j) th block is given by

$$\mathcal{A}_{ij} = \delta_{ij} \left(\sum_{k=1}^d G_k^T M_\mu G_k \right) + \gamma G_j^T M_\mu G_i + \delta_{ij} \mathcal{E},$$

where δ_{ij} is the Kronecker delta, $G : V_\varphi \rightarrow V_\varphi^d$ is a full-degree discrete gradient operator, M_μ is a μ -weighted block-diagonal mass matrix relative to V_φ , and $\mathcal{E}$ is a block-sparse matrix implementing velocity penalty stabilisation. Eq. (4) gives the system in component form; we will also refer to it through an abridged notation,

$$\begin{pmatrix} \mathcal{A} & \mathcal{G} \\ D & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}_h \\ p_h \end{pmatrix} = \begin{pmatrix} \mathbf{b}_u \\ b_p \end{pmatrix}, \quad (5)$$

where $\mathcal{G} : V_{\varphi-1} \rightarrow V_\varphi^d$ is an effective pressure gradient operator and $D : V_\varphi^d \rightarrow V_{\varphi-1}$ is an effective negated velocity divergence operator. The row $(\mathcal{A} \quad \mathcal{G})$ is referred to as the stress divergence operator and the row $(D \quad 0)$ as the divergence constraint operator. In terms of multigrid design, we shall also refer to the Stokes system Eqs. (4) and (5) through the condensed form

$$A_h x_h = b_h,$$

where $x_h \in \mathcal{V}_h$ collects $\mathbf{u}_h$ and p_h into one set of unknowns, $\mathcal{V}_h := V_\varphi^d \otimes V_{\varphi-1}$, and A_h is the entire symmetric saddle point operator. (The subscript h may be dropped when context permits.)

2.1. Comparison to equal-degree framework

Before we shift focus to the development of multigrid solvers, we briefly compare the mixed-degree and equal-degree LDG approaches. The equal-degree framework is essentially the same except for three main aspects: (i) the pressure field is of full degree, i.e., $p_h \in V_\varphi$; (ii) the operators $\mathcal{G}$ and D in Eq. (5) are replaced by analogous full-degree counterparts; and (iii) a pressure penalty stabilisation operator $\mathcal{P}$ is included in the bottom-right block of the Stokes system, so that it takes the form $\begin{pmatrix} \mathcal{A} & \mathcal{G} \\ D & -\mathcal{P} \end{pmatrix}$. The purpose of $\mathcal{P}$ is to ensure stability in the discretised Stokes system by weakly enforcing continuity of the pressure field; for equal-degree LDG discretisations it is defined such that [7]

$$q^T \mathcal{P} p = \int_{\Gamma_o} \tau_p \llbracket q \rrbracket \llbracket p \rrbracket, \quad (6)$$

for every $p, q \in V_\varphi$, where Γ_o denotes the set of all interior mesh faces and τ_p is a pressure penalty parameter (proportional to the element size h and inversely proportional to the local viscosity).² This form of pressure penalisation is applicable whenever the pressure field is expected to be continuous, or, via a simple modification, whenever interfacial jumps in p can be specified *prior* to solving the Stokes problem. However, for multiphase problems involving interfacial stress jump conditions, the interfacial jump in p is not known a priori, because it cannot be inferred from the given stress jump conditions $\llbracket \sigma \cdot \mathbf{n} \rrbracket$ without first knowing $\llbracket \mu(\nabla \mathbf{u} + \nabla \mathbf{u}^T) \cdot \mathbf{n} \rrbracket$. Similarly, for problems with prescribed stress boundary conditions, we cannot weakly enforce pressure boundary conditions via Eq. (6) (were it to include boundary faces) because $p|_{\partial\Omega}$ is not known a priori. In general, some kind of pressure stabilisation is needed for equal-degree LDG schemes in order to ensure stability of the discretised Stokes system [7]. As part of this work, some alternative pressure penalisation approaches were investigated but none were found satisfactory,³ hence the adoption here of a mixed-degree framework that requires no pressure penalisation whatsoever.

¹ Scattered throughout this paper we often apply a few convenient abuses of notation. A piecewise polynomial function (e.g., in V_φ) may carry the same notation as its corresponding coefficient vector relative to the chosen basis of V_φ , with the precise meaning understood from context. For example, in the identity $u^T M v = \int_\Omega u v$, the left-hand side uses vectors and matrices, whereas the right-hand side applies the functional form. Likewise, operator notation may refer to the functional form or to its matrix representation relative to the chosen basis. For example, the i th component of the broken gradient operator $\tilde{G} : V_\varphi \rightarrow V_\varphi^d$ satisfies $u^T M \tilde{G}_i v = \int_\Omega u \tilde{G}_i v = \sum_{E \in \mathcal{E}} \int_E u \partial_i v$, for every $u, v \in V_\varphi$.

² One may also interpret $\mathcal{P}$ as penalising the difference between $\mathcal{G}p$ and the broken gradient of p .

³ We mention here two approaches. (i) One possibility is to include a discrete viscous stress tensor σ_h as an additional solution variable, thereby expanding the coupled Stokes system to solve for velocity (a vector field), pressure (a scalar field), and stress (a matrix-valued field); then, instead of penalising $\llbracket p_h \rrbracket$, one may penalise $\llbracket \sigma_h \cdot \mathbf{n} \rrbracket$. This approach results in a stable discretisation and brief experimentation suggests that one can build fast-converging multigrid solvers. However, the substantial increase in system size renders this approach too computationally expensive, especially for high-order 3D problems. (ii) Another approach is to generalise Eq. (6) so as to infer $\llbracket p \rrbracket$ as part of the solution process itself. Two possibilities that preserve the overall symmetry of the Stokes operator are to replace the bilinear penalty functional with either

$$\int_{\Gamma_o} \llbracket \sigma(\mathbf{v}, q) \cdot \mathbf{n} \rrbracket \cdot \llbracket \sigma(\mathbf{u}, p) \cdot \mathbf{n} - \mathbf{h} \rrbracket \quad \text{or} \quad \int_{\Gamma_o} \llbracket \mathbf{n} \cdot \sigma(\mathbf{v}, q) \cdot \mathbf{n} \rrbracket \llbracket \mathbf{n} \cdot \sigma(\mathbf{u}, p) \cdot \mathbf{n} - \mathbf{h} \cdot \mathbf{n} \rrbracket,$$

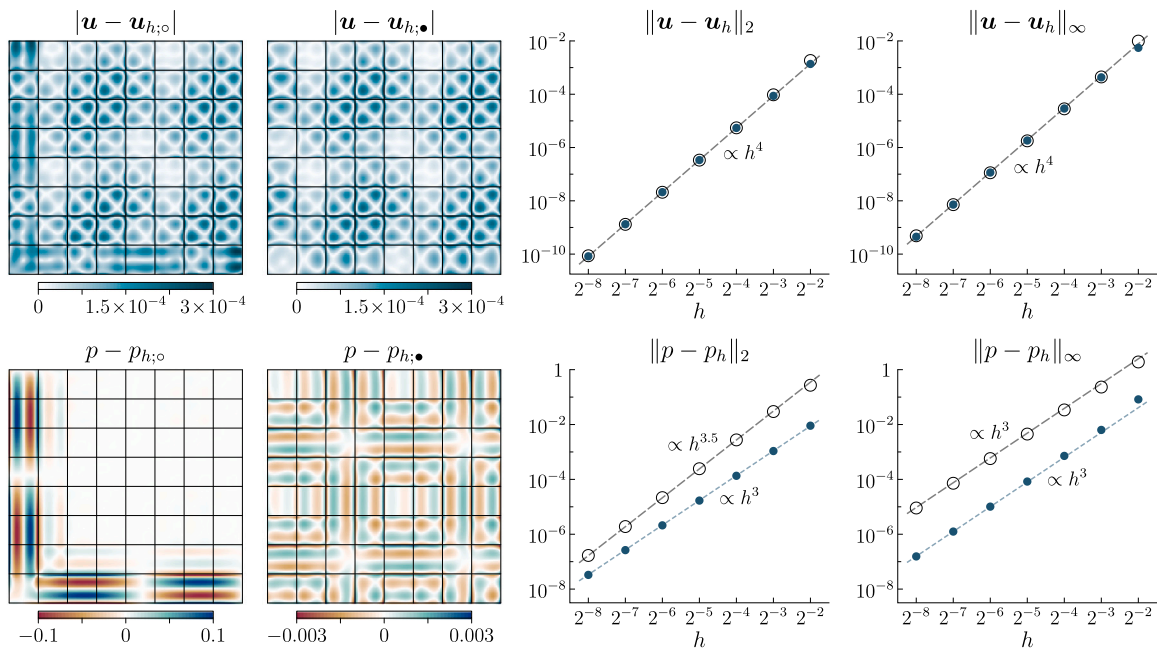


Fig. 1. Comparing solution accuracy for the two main LDG approaches: equal-degree (far-left and white circles $\circ$) and mixed-degree (middle-left and filled circles $\bullet$).

Besides the issue of pressure penalisation, another key difference between the mixed-degree and equal-degree frameworks is that of solution accuracy. In particular, the equal-degree approach often leads to significant pressure errors near boundaries and interfaces, whereas the mixed-degree approach yields cleaner and more accurate pressure fields. To demonstrate this, the left portion of Fig. 1 compares discrete solutions arising from a single-phase stress-form Stokes problem, with $\mu \equiv 1$ on $\Omega = (0, 1)^2$, stress boundary conditions, and source data generated from a sinusoidal exact solution.⁴ The far-left (resp., middle-left) figures plot the solution error of the equal-degree (resp., mixed-degree) solution, as computed via $\varphi = 3$ on an 8×8 mesh. Note that in the equal-degree case, the pressure field contains significant boundary errors, around two orders of magnitude greater than the interior; although the differences in velocity field error are less pronounced, one may also observe a weak numerical boundary layer in the top-left plot. It is important to note that similar outcomes occur for Dirichlet boundary conditions and for embedded interfaces in multiphase problems—as such, the large numerical boundary layers of the equal-degree approach are ingrained in the discretisation itself (and not so much the nuances of penalisation). Using the same test problem, the right-half of Fig. 1 contains the results of a grid convergence test, examining the maximum and L^2 error norms over grid sizes 4×4 up to 256×256 . Note that both approaches yield nearly identical velocity errors of optimal order accuracy, i.e., order $\varphi + 1$. Regarding the pressure error, once again we see a substantial difference—both approaches yield order φ in the maximum norm (for mixed-degree, this is optimal; for equal-degree, this is suboptimal), but the mixed-degree pressure error is smaller by a factor of ≈ 60 . Comparison in the L^2 norm is more subtle: roughly speaking, it is possible for the equal-degree setting to yield optimal order pressure fields sufficiently far from boundaries/interfaces, which means it is possible for $\|p - p_h\|_2 \in \mathcal{O}(h^{\varphi+1/2})$, as observed in the bottom-middle-right plot of Fig. 1. Nevertheless, the mixed-degree case yields better L^2 -norm pressure errors (of order φ). Overall, to various extents the same observations hold for other kinds of test problems and other choices of polynomial degrees φ , making the mixed-degree approach a compelling choice for LDG-based Stokes solvers.

where $\mathbf{v}$ and q are test functions and $\sigma(\cdot, \cdot)$ is an LDG viscous stress tensor operator. In either case, the new Stokes system takes the form

$$\begin{pmatrix} \mathcal{A} - \mathcal{P}_{11} & \mathcal{G} + \mathcal{P}_{12} \\ \mathcal{D} + \mathcal{P}_{12}^\top & -\mathcal{P}_{22} \end{pmatrix},$$

where both $\mathcal{P}_{11}$ and $\mathcal{P}_{22}$ are symmetric positive semidefinite. Unfortunately, neither form of stress penalisation is effective in practice—to stabilise pressure, the penalisation needs to be sufficiently strong; on the other hand, it needs to be sufficiently weak so as to preserve enough of the smoothing properties of $\mathcal{A}$ needed for effective multigrid solvers; in fact, if it is too strong then $\mathcal{A} - \mathcal{P}_{11}$ becomes indefinite and stability is lost entirely—numerical experiments indicate there is no good middle ground between these competing needs.

⁴ The exact solution used to generate the data of Fig. 1 consists of unit-amplitude sine waves; specifically $\mathbf{u} = (u, v)$ where $u = u(x, y) = \cos(2\pi x + 0.25) \cos(2\pi y + 1.25)$, $v = v(x, y) = \cos(2\pi x + 4.25) \cos(2\pi y + 5.25)$, and $p = p(x, y) = \cos(2\pi x - 3.75) \cos(2\pi y - 2.75)$.

2.2. Choice of basis

One may implement the above LDG framework using any reasonable basis; for example, a tensor-product Gauss–Lobatto nodal basis or a tensor-product Legendre polynomial orthogonal basis. We have used the latter approach in this work as it simplifies the design and implementation of the multigrid smoother.

3. Multigrid schemes

The multigrid framework considered here applies the standard concepts of a V-cycle operating on a nested mesh hierarchy. To summarise the key ingredients:

- *Mesh hierarchy.* In this work we use quadrees and octrees to define the finest mesh; the corresponding tree structure naturally defines a hierarchical process by which to agglomerate mesh elements into a nested mesh hierarchy $\mathcal{E}_h, \mathcal{E}_{2h}, \mathcal{E}_{4h}, \dots$
- *Interpolation operator.* The interpolation operator $I_{2h}^h : \mathcal{V}_{2h} \rightarrow \mathcal{V}_h$ transfers coarse mesh corrections to a fine mesh and is defined naturally via polynomial injection: $(I_{2h}^h u)|_{E_f} := u|_{E_c}$, where E_f is a fine mesh element and $E_c \supseteq E_f$ is its corresponding coarse mesh element. Note that I_{2h}^h independently acts on each component of the state: on velocity (resp., pressure) components it is a degree $\wp$ (resp., $\wp - 1$) interpolation operator.
- *Restriction operator.* The restriction operator $R_h^{2h} : \mathcal{V}_h \rightarrow \mathcal{V}_{2h}$ transfers a fine mesh residual onto the coarse mesh and is defined via L^2 Galerkin projection onto $\mathcal{V}_{2h}$. Equivalently, it can be defined via the adjoint of the interpolation operator, $R_h^{2h} := \mathcal{M}_{2h}^{-1} (I_{2h}^h)^\top \mathcal{M}_h$, where $\mathcal{M}_h$ is the block-diagonal mass matrix relative to $\mathcal{V}_h$.

To define the primary Stokes operator A_h on every level of the hierarchy, there are two possible approaches: (i) one may use a “pure geometric multigrid” method in which every level is explicitly meshed (e.g., listing out the elements and faces, their connectivity and quadrature, etc.) upon which $A_{2h}, A_{4h}, \dots$ is built via the LDG discretisation; alternatively, (ii) one may use an operator-coarsening approach that recursively and conveniently automates this process and guarantees the coarse grid problems are discretised in a way that harmoniously matches the fine grid’s discretisation. We have used the latter approach in this work—for more details, see [8], which introduces LDG operator-coarsening methods for Poisson problems, and [1], for their extension to multiphase Stokes problems.

Algorithm 1 Multigrid V-cycle $V(\mathcal{E}_h, x_h, b_h)$ with v_1 pre- and v_2 post-smoothing steps on mesh $\mathcal{E}_h$ of the hierarchy.

```

1: if  $\mathcal{E}_h$  is the bottom level then
2:   Solve  $A_h x_h = b_h$  using the bottom level direct solver.
3: else
4:   Apply pre-smoother  $v_1$  times.
5:   Compute restricted residual,  $r_{2h} := (I_{2h}^h)^\top (b_h - A_h x_h)$ .
6:   Solve coarse grid problem,  $x_{2h} := V(\mathcal{E}_{2h}, 0, r_{2h})$ .
7:   Interpolate and correct,  $x_h \leftarrow x_h + I_{2h}^h x_{2h}$ .
8:   Apply post-smoother  $v_2$  times.
9: return  $x_h$ .
```

We consider here the template of a standard V-cycle, as shown in Algorithm 1.⁵ The design of the smoother is one of our main focuses in this work. Besides the smoother, we have made the following implementation choices:

- *Bottom solver.* On the bottom level of the hierarchy (often consisting of just one element spanning the entire domain), a symmetric eigendecomposition of A is precomputed and used as a direct solver. For Stokes systems, oftentimes there is a nontrivial kernel matching that of the non-discretised continuum problem, e.g., constant velocity and/or pressure fields. In these cases, the bottom solver applies a standard pseudoinverse approach to appropriately handle kernel eigenmodes.
- *Pre- and post-smoother iteration counts.* There is a balance between choosing v relatively high, thus needing fewer (but more expensive) V-cycles, versus choosing v relatively low, thus needing more (but cheaper) V-cycles. Prior work in fast LDG multigrid methods [1,2,8,9] has consistently indicated that $v_1 \approx 3$ and $v_2 \approx 3$ generally yields the best results in terms of fastest solver time to reduce the residual by a fixed factor. Consequently, to keep the focus on key results, we have fixed $v_1 = v_2 = 3$.
- *Solver acceleration.* Although multigrid can be used as a standalone iterative method, its convergence can be accelerated by using it as a preconditioner in a Krylov method such as Generalised Minimal Residual (GMRES) [10,11]. For the numerical study of this work, we apply multigrid-preconditioned GMRES as the main solver driver, using a single application of the V-cycle of Algorithm 1 as left-preconditioner.

⁵ Note that $(I_{2h}^h)^\top$ appears on line 5 rather than the operator R_h^{2h} ; this follows from the Galerkin approach, namely that $A_h x_h = b_h$ implements the PDE problem in the “mass basis”, i.e., the discretised operator, mapping functions from V_h to V_h , is given by $M_h^{-1} A_h$. As a piecewise polynomial function in V_h , the residual of the fine mesh problem is given by $M_h^{-1} (b_h - A_h x_h)$. In the multigrid V-cycle, this residual is multiplied by R_h^{2h} to define the right-hand side data for the coarse mesh problem: $M_{2h}^{-1} A_{2h} x_{2h} = R_h^{2h} M_h^{-1} (b_h - A_h x_h)$, i.e., $A_{2h} x_{2h} = (I_{2h}^h)^\top (b_h - A_h x_h) =: r_{2h}$.

In the remainder of this section we discuss the design of the multigrid smoother, adaptation to unsteady Stokes problems, application in the multigrid V-cycle, and finally the optimisation of its parameters.

3.1. Smoother preliminaries

The multigrid smoother developed in this work is in many ways analogous to a multicoloured block Gauss–Seidel smoother, where each *block* corresponds to the collective set of degrees of freedom on each mesh element, i.e., velocity and pressure combined. Accordingly, partition $Ax = b$ so that $\sum_j A_{ij}x_j = b_i$, where $(\cdot)_j$ denotes the set of coefficients on element i and $A_{ij} \in \mathbb{R}^{n \times n}$ denotes the corresponding (i, j) th block of A , each of size $n := d(\wp + 1)^d + \wp^d$. Given an input approximation $x \in \mathcal{V}_h$, one iteration of block Gauss–Seidel corresponds to sweeping over the elements in some particular order so as to compute an updated approximation $\bar{x} \in \mathcal{V}_h$ whereby

$$\bar{x}_i := x_i + A_{ii}^{-1} \left(b_i - \sum_j A_{ij} \hat{x}_{j,i} \right) \quad \text{where} \quad \hat{x}_{j,i} := \begin{cases} \bar{x}_j & \text{if } j < i, \\ x_j & \text{otherwise,} \end{cases} \quad (7)$$

where “ $j < i$ ” means block j is updated before block i . As mentioned in the introduction, for equal-degree LDG Stokes problems, block Gauss–Seidel leads to highly effective multigrid solvers (see, e.g., [1,2]); however, it is ineffective on mixed-degree LDG problems, even with the best possible damping/overrelaxation parameter—V-cycle iteration counts are significantly higher and convergence rates strongly decay as the grid is refined. One of the goals in this work is to develop a replacement smoother that has the same application cost as block Gauss–Seidel; in essence, this is achieved by replacing the inverse block-diagonal term A_{ii}^{-1} in (7) by a regularised version, denoted Q_i , partly inspired by balanced SAI smoothers [2]. In turn, Q_i is computed through a weighted least squares approximation so that $A \operatorname{diag}(Q_i) \approx \mathbb{I}$, except relatively few off-diagonal components of A enter the least squares calculation, the right-hand side is altered to include damping/overrelaxation coefficients, and the weighting is chosen following a balancing process designed to smooth the velocity and pressure variables at the same rate. These aspects are controlled by four positive parameters: ζ_σ , ζ_u , ω_u , and ω_p . The parameters ζ_σ (resp., ζ_u) control the weighting of the rows corresponding to the stress divergence operator (resp., divergence constraint operator) on the diagonal block of A . Meanwhile, ω_u (resp., ω_p) effectuate the damping or overrelaxation of the velocity (resp., pressure) degrees of freedom, thereby allowing these variables to be damped/overrelaxed by different amounts. The value of these parameters are tunable but depend only on $\wp$, γ , and d and are otherwise constant; typically, ω_u and ω_p are close to unity, while ζ_u is unit-order in magnitude, and we shall see later that ζ_σ can in essence be eliminated from consideration.

To detail the construction of the smoother Q , some additional notation is useful. For an arbitrary coefficient vector $x \in \mathcal{V}_h = V_{\wp}^d \otimes V_{\wp-1}$, let $x_{i,u,\ell,k}$ denote the component of x associated with the k th modal coefficient of the ℓ th component of the velocity field on the i th element; similarly, let $x_{i,p,k}$ denote the k th degree of freedom of the pressure field on the i th element.⁶ In particular, let $k = 0$ correspond to the constant-mode lowest-degree modal coefficient; for example, if $x \equiv 0$ except for $x_{i,p,0} \neq 0$, then x corresponds to an identically zero velocity field and piecewise constant pressure field.

For the moment, suppose that we have appropriately “balanced” the Stokes operator. Then, corresponding to the construction of Q_i on the i th element, define the weighting vector $\zeta_i \in \mathcal{V}_h$ such that $\zeta_i \equiv 0$, except for the following components: $\zeta_{i,u,\ell,k} = \zeta_\sigma$ for every ℓ and k ; $\zeta_{i,u,p,k} = \zeta_u$ for every k ; and $\zeta_{i,j,u,\ell,0} = 1$ for every ℓ and $j \neq i$. In addition, define the damping/overrelaxation vector $\omega \in \mathcal{V}_h$ such that $\omega \equiv 0$, except for the following components: $\omega_{j,u,\ell,k} = \omega_u$ for every j, ℓ, k ; and $\omega_{j,p,k} = \omega_p$ for every j, k . Then, Q_i is defined as the solution of the following minimisation problem:

$$Q_i = \operatorname{argmin}_{Q \in \mathbb{R}^{n \times n}} \left\| \operatorname{diag}(\zeta_i) (A_{:,i} Q - \operatorname{diag}(\omega) \mathbb{I}_{:,i}) \right\|_F^2, \quad (8)$$

where $A_{:,i}$ denotes the i th block column of A and $\|\cdot\|_F$ denotes the Frobenius norm. Eq. (8) implements an overdetermined weighted least squares problem, whose matrix is essentially a trimmed version of the i th block column of A , with right-hand side data equal to the i th block column of the ω -weighted identity matrix $\mathbb{I}_\omega := \operatorname{diag}(\omega) \mathbb{I}$. Importantly, note that among the nonzero off-diagonal blocks of $A_{:,i}$, only a small subset of rows are actually kept: the weighting vector ζ_i , which multiplies the rows of $A_{:,i}$, is zero everywhere except for the rows of the stress divergence operator associated with constant-mode lowest-degree modal coefficients. There are only d such rows per off-diagonal block and all have unit weighting, while the weighting of the rows in the diagonal block are controlled by ζ_σ and ζ_u .

3.2. Balancing

We next consider the balancing of the Stokes operator, via a simple diagonal pre- and post-scaling of A . This balancing approach is similar to that developed in prior work [2], and treats two key aspects of Stokes problems:

⁶ Beyond the grouping into element-wise blocks and the consistent matching of the degrees of freedom of blockwise matrices and vectors, none of the algorithms described in this work depend on the particular ordering thereof.

- *Mesh-dependent operator scaling.* Note that the operators within Eq. (5) scale with h and μ in different ways: $\mathcal{A} \sim \mu h^{d-2}$, $\mathcal{G} \sim h^{d-1}$, and $D \sim h^{d-1}$. In particular, note the dependence on h means that the length scales of the domain, or even the particular level of the multigrid hierarchy, can change the relative strengths of these operators, even when the entire collection of elements have the same viscosity coefficient. Without treatment, this may skew the solution of the least squares problem in Eq. (8), by over- or under-penalising the off-diagonal components, or by counteracting the role of the weighting parameters ζ_σ and ζ_u that are meant to (universally, for any μ and any h) balance the smoothing characteristics of velocity and pressure variables.
- *High-contrast problems.* Whenever the viscosity coefficient μ exhibits large variations or discontinuities, the various (sub)blocks A_{ij} may then have considerably different magnitudes—left untreated, this may again skew the above least squares formulation and lead to poor multigrid smoothing performance, e.g., because nearby elements are smoothed at different rates.

Fortunately, these aspects can be treated via a simple balancing process. In a form analogous to Eq. (5), the i th diagonal block of A is equal to

$$\begin{pmatrix} A_{ii} & \mathcal{G}_{ii} \\ D_{ii} & 0 \end{pmatrix}. \quad (9)$$

Our balancing approach performs a simple diagonal pre- and post-scaling of A in order to calibrate the norms of the (newly-scaled) blocks in Eq. (9). There are many possible approaches⁷; here we apply a scheme based on the block-diagonal broken gradient operator $\tilde{G} = (\tilde{G}_1, \dots, \tilde{G}_d) : V_\varphi \rightarrow V_\varphi^d$ and how it factors into the operators of Eq. (9), as follows. We define the “scale” $S_{A,i}$ of A_{ii} as

$$S_{A,i} := \left(d \sum_{\ell=1}^d \left\| (\tilde{G}_{\ell,ii})^\top M_{\mu,ii} (\tilde{G}_{\ell,ii}) \right\|_F^2 \right)^{\frac{1}{2}}.$$

This definition reflects that, except for lifting operators, the viscous operator $\mathcal{A}$ is roughly d copies of a μ -weighted Laplacian operator. Similarly, we define the scale $S_{D,i}$ of D_{ii} as

$$S_{D,i} := \left(\sum_{\ell=1}^d \left\| M_{ii} \tilde{G}_{\ell,ii} \right\|_F^2 \right)^{\frac{1}{2}}.$$

Using these scaling factors, let $\alpha \in \mathcal{V}_h$ denote a global scaling vector such that $\alpha_{i,u,\ell,k} := (S_{A,i})^{-1/2}$ and $\alpha_{i,p,k} := (S_{A,i})^{1/2} (S_{D,i})^{-1}$ for all i, ℓ, k . The calibration of the blocks in Eq. (9) is then implemented via the rescaled Stokes operator $\tilde{A} := \text{diag}(\alpha) A \text{diag}(\alpha)$; in particular, all of the newly-scaled subblocks have unit-magnitude Frobenius norm.

This balancing approach is applied as follows. Define $\tilde{A} := \text{diag}(\alpha) A \text{diag}(\alpha)$ and invoke the least squares methods discussed in Section 3.1 on $\tilde{A}$ (instead of A) thereby computing the approximate inverse $\tilde{Q}$ satisfying $\tilde{A}\tilde{Q} \approx \mathbb{I}_\omega$. The true approximate inverse of A is then defined by $Q := \text{diag}(\alpha)\tilde{Q}\text{diag}(\alpha)$ so that $AQ \approx \mathbb{I}_\omega$. One may view this process as a simple diagonal preconditioning of A to treat high-contrast viscosity coefficients as well as the mesh-dependent scaling of the viscous stress, pressure gradient, and velocity divergence operators. One may also view this process as augmenting the weighted least squares problem via additional weighting factors that “cancel out” the scaling effects of μ and h .

3.3. Unsteady Stokes problems

Thus far we have focused on steady-state Stokes problems. For unsteady/nonstationary Stokes problems, the introduction of a density-weighted temporal derivative term can alter the characteristics of the Stokes system and therefore the multigrid solver. The governing equations consist of keeping the interfacial jump conditions Eq. (2) and boundary conditions Eq. (3) but amending Eq. (1) to include a density-weighted term that would arise from, e.g., a time-stepping method applied to the incompressible Navier–Stokes equations whereby the advection term is treated explicitly and the viscous term implicitly. The amended form reads

$$\left. \begin{aligned} \frac{\rho_i}{\delta} \mathbf{u} - \nabla \cdot (\mu_i (\nabla \mathbf{u} + \gamma \nabla \mathbf{u}^\top)) + \nabla p &= \mathbf{f} \\ -\nabla \cdot \mathbf{u} &= f \end{aligned} \right\} \text{ in } \Omega_i, \quad (10)$$

where $\rho_i > 0$ is the density of phase i and $\delta > 0$ is a parameter proportional to the time step of a temporal integration method. The corresponding LDG discretisation is a straightforward modification to the steady-state problem and leads to

$$\begin{pmatrix} \frac{1}{\delta} M_\rho + \mathcal{A} & \mathcal{G} \\ D & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}_h \\ p_h \end{pmatrix} = \begin{pmatrix} \mathbf{b}_u \\ b_p \end{pmatrix}, \quad (11)$$

where M_ρ is a block diagonal ρ -weighted mass matrix. Beyond this simple modification, what makes the unsteady problem subtle is the relative strengths of the viscous operator and the newly-added density-weighted term [1,2]. When $\mu\delta/\rho$ is sufficiently large,

⁷ For example, one could simply compute the Frobenius norms of the matrices appearing in Eq. (9) and use that to determine a diagonal scaling. This approach is easy, except for some edge cases concerning $\varphi = 1$ for which it is possible that $D_{ii} = 0$ and $\mathcal{G}_{ii} = 0$ on a small subset of mesh elements (depending on boundary conditions, numerical flux choices, etc.). We instead opted for an approach that works in all settings, for all φ , and this led to the method described here based on the scaling of the broken gradient operator.

we essentially have a steady-state Stokes system with a small ρ/δ -weighted identity shift added to the viscous operator; in this case, a good steady-state multigrid Stokes solver can be effective. Conversely, when $\rho/(\mu\delta)$ is sufficiently large, Eq. (11) approximately reduces to a Poisson problem for pressure (written in flux-form), closely related to Chorin's projection method for incompressible fluid dynamics [12]. The role of the pressure variable p changes in these two extremes, from being a Lagrange multiplier enforcing the divergence constraint (in the former), to being the primary solution variable in an elliptic interface problem (in the latter). Crucially, note also that the relative strengths of these operators can change across the levels of a multigrid hierarchy—on highly-refined meshes, viscous effects may dominate; on coarse grids, the density-weighted term may dominate.

These aspects can be treated via a refinement to the balancing process, together with a blending procedure applied to the smoother parameters $(\zeta_\sigma, \zeta_u, \omega_u, \omega_p)$, as follows.

- The “scale” of the temporal derivative term on the i th element is defined via

$$S_{\rho,i} := \frac{1}{\delta} \|M_{\rho,ii}\|_F.$$

In the balancing process of Section 3.2, the scaling of the top-left block of Eq. (11) is defined as $(S_{\mathcal{A},i}^2 + S_{\rho,i}^2)^{1/2}$, and we appropriately redefine the scaling vector α .

- The smoother parameters are blended via a harmonic-weighted convex combination. In one extreme, suppose that $\mu \neq 0$ and $\rho \equiv 0$ leads to a particular set of optimised smoother parameters $(\zeta_\sigma^0, \zeta_u^0, \omega_u^0, \omega_p^0)$; in the other extreme, suppose that $\mu \equiv 0$ and $\rho \neq 0$ leads to optimised smoother parameters $(\zeta_\sigma^\infty, \zeta_u^\infty, \omega_u^\infty, \omega_p^\infty)$. (The superscripts denote the effective Reynolds number.) Then, to build the smoother Q_i on element i , for any combination of $\rho > 0$ and $\mu > 0$, we apply the following smoother parameters:

$$(\zeta_\sigma, \zeta_u, \omega_u, \omega_p) = (1 - \lambda_i)(\zeta_\sigma^0, \zeta_u^0, \omega_u^0, \omega_p^0) + \lambda_i(\zeta_\sigma^\infty, \zeta_u^\infty, \omega_u^\infty, \omega_p^\infty)$$

where

$$\lambda_i := \frac{S_{\rho,i}}{S_{\rho,i} + S_{\mathcal{A},i}}.$$

Here, $\lambda_i \in [0, 1]$ is a blending amount that depends on the local, mesh-dependent relative strengths of the density-weighted temporal operator and the viscosity-weighted viscous operator. Note that the blending amount can vary across the multigrid hierarchy, and even between elements on the same level.

3.4. Smoother construction and application

Assembling the above steps, Algorithm 2 details the overall approach to construct the block-diagonal smoother Q . We note several aspects:

- The implementation of Algorithm 2 is perhaps simpler than it may first appear. The main step is that of assembling and solving the least squares problems. In essence, assembly involves visiting every nonzero block in the i th block column of $\tilde{A}$ —upon visiting the diagonal block, rescale the rows arising from the stress divergence (resp., velocity constraint) operator by ζ_σ (resp., ζ_u) and append them to a temporary matrix; upon visiting an off-diagonal block, keep only the rows arising from the stress divergence operator associated with constant-mode lowest-degree modal coefficients, and append them to the same temporary matrix. The resulting matrix is of size $m \times n$ where $n = d(\wp + 1)^d + \wp^d$ and $m = n + \ell d$, where ℓ is the number of off-diagonal blocks (often just four in 2D or six in 3D). A similar temporary matrix is built to define the right-hand side of the least squares problem, i.e., an $m \times n$ matrix with either $\zeta_\sigma \omega_u$ or $\zeta_u \omega_p$ appearing on the diagonal entries. The Frobenius norm minimisation problem (line 12) translates to a standard least squares problem with n right-hand side vectors; in this work we utilise a standard QR method from LAPACK for its solution.
- Each individual least squares problem involves a nearly square matrix with computational cost similar to inverting a diagonal block of A ; further comments on the size of these matrices are given in the concluding remarks. Moreover, each least squares problem is well-conditioned and almost always full-rank.⁸
- Note that every diagonal block of Q can be (pre)computed independently of (and possibly concurrently with) every other diagonal block; i.e., the algorithm is trivially parallelisable.

We now discuss the application of Q as a multigrid smoother, analogous to multicoloured block Gauss–Seidel. Let the input approximation be denoted by $x \in \mathcal{V}_h$ and the output by $\bar{x} \in \mathcal{V}_h$. In general, Q is asymmetric, so there are two alternatives to the Gauss–Seidel sweep of Eq. (7): the first uses Q ,

$$\bar{x}_i := x_i + Q_i(b_i - \sum_j A_{ij} \hat{x}_{j,i}) \quad \text{where} \quad \hat{x}_{j,i} := \begin{cases} \bar{x}_j & \text{if } j < i, \\ x_j & \text{otherwise,} \end{cases} \quad (12)$$

⁸ The exception is when a single mesh element spans the entire domain, plus some idiosyncratic cases involving $\wp = 1$; in these cases the rank deficiency is straightforward to handle, simply by adopting the minimum-norm minimum-residual least squares solution.

Algorithm 2 Construction of Q , given smoother parameters $(\zeta_\sigma^0, \zeta_u^0, \omega_u^0, \omega_p^0)$ and $(\zeta_\sigma^\infty, \zeta_u^\infty, \omega_u^\infty, \omega_p^\infty)$ for the extremal cases of $\{\mu \neq 0, \rho \equiv 0\}$ and $\{\mu \equiv 0, \rho \neq 0\}$, respectively.

-
- 1: **for** every element i (possibly in parallel) **do**
 - 2: $S_{\mathcal{A},i} := \left(d \sum_{\ell=1}^d \left\| (\hat{G}_{\ell,ii})^\top M_{\mu,ii} (\hat{G}_{\ell,ii}) \right\|_F^2 \right)^{\frac{1}{2}}.$
 - 3: $S_{D,i} := \left(\sum_{\ell=1}^d \left\| M_{ii} \hat{G}_{\ell,ii} \right\|_F^2 \right)^{\frac{1}{2}}.$
 - 4: $S_{\rho,i} := \begin{cases} 0 & \text{for steady-state Stokes problems,} \\ \frac{1}{\delta} \|M_{\rho,ii}\|_F & \text{for unsteady Stokes problems.} \end{cases}$
 - 5: Define the rescaling vector $\alpha \in \mathcal{V}_h$ such that $\forall i, \ell, k$,

$$\alpha_{i,u,\ell,k} := (S_{\mathcal{A},i}^2 + S_{\rho,i}^2)^{-1/4},$$

$$\alpha_{i,p,k} := (S_{\mathcal{A},i}^2 + S_{\rho,i}^2)^{1/4} (S_{D,i})^{-1}.$$
 - 6: Define the balanced Stokes operator $\tilde{A} := \text{diag}(\alpha) A \text{diag}(\alpha)$.
 - 7: **for** every element i (possibly in parallel) **do**
 - 8: $\lambda_i := S_{\rho,i} / (S_{\rho,i} + S_{\mathcal{A},i}).$
 - 9: $(\zeta_\sigma, \zeta_u, \omega_u, \omega_p) := (1 - \lambda_i)(\zeta_\sigma^0, \zeta_u^0, \omega_u^0, \omega_p^0) + \lambda_i(\zeta_\sigma^\infty, \zeta_u^\infty, \omega_u^\infty, \omega_p^\infty).$
 - 10: Define $\zeta_i \in \mathcal{V}_h$ such that $\zeta_i \equiv 0$, except for

$$\zeta_{i;i,u,\ell,k} := \zeta_\sigma \quad \forall \ell, k,$$

$$\zeta_{i;i,p,k} := \zeta_u \quad \forall k,$$

$$\zeta_{i;j,u,\ell,0} := 1 \quad \forall \ell, j \neq i.$$
 - 11: Define $\omega_i \in \mathcal{V}_h$ such that $\omega_i \equiv 0$, except for

$$\omega_{i;i,u,\ell,k} := \omega_u \quad \forall \ell, k,$$

$$\omega_{i;i,p,k} := \omega_p \quad \forall k.$$
 - 12: Solve the least squares minimisation problem

$$\tilde{Q}_i := \underset{Q \in \mathbb{R}^{n \times n}}{\text{argmin}} \left\| \text{diag}(\zeta_i) (\tilde{A}_{:,i} Q - \text{diag}(\omega_i) \mathbb{I}_{:,i}) \right\|_F^2.$$
 - 13: **return** $Q := \text{diag}(\alpha) \tilde{Q} \text{diag}(\alpha).$
-

while the second uses the transpose of Q ,

$$\bar{x}_i := x_i + Q_i^\top (b_i - \sum_j A_{ij} \bar{x}_{j,i}) \quad \text{where} \quad \bar{x}_{j,i} := \begin{cases} \bar{x}_j & \text{if } j < i, \\ x_j & \text{otherwise,} \end{cases} \quad (13)$$

where as before, “ $j < i$ ” means block j is updated before block i . One could therefore apply any combination of Eq. (12) or Eq. (13) as a pre-smoother or post-smoother. Numerical experiments indicate that the fastest multigrid solvers arise from applying Eq. (12) as a pre-smoother and Eq. (13) as a post-smoother—consequently, the same choice is used throughout the numerical experiments presented below; see also the results of [2] that arrive at a similar conclusion for smoothers based on general SAI methods. Regarding the element sweep order, we consider here a multicoloured approach. Specifically, in a setup phase, a graph-colouring algorithm is applied to the connectivity graph associated with the block sparsity pattern of the Stokes operator A , thereby assigning a colour to each element.⁹ (On Cartesian grids with one-sided numerical fluxes, this recovers the well-known red-black colouring associated with the 5-point (2D) or 7-point (3D) Laplacian stencil.) In the pre-smoothing sweep of Eq. (12) and post-smoothing sweep of Eq. (13), every element of the same colour is processed (possibly in parallel), before moving onto the next colour.¹⁰

3.5. Optimal smoothing parameters

To summarise the discussion so far, we have built a smoother analogous to multicoloured block Gauss–Seidel, taking into account key aspects of multiphase Stokes problems such as mesh-dependent operator scaling, as well as the application of the smoother in a multigrid V-cycle. The smoother has four tunable parameters: ζ_σ and ζ_u , controlling the relative strengths of velocity and pressure

⁹ For example, one could apply the DSATUR graph-colouring algorithm [13].

¹⁰ In particular, the pre- and post-smoothing sweeps use the same colour ordering, a choice motivated by the findings of [2], where this configuration typically yielded faster convergence than colour reversal during post-smoothing.

variables in the approximate inverses, and ω_u and ω_p , controlling the damping/overrelaxation of velocity and pressure degrees of freedom. To conclude this section, we discuss the optimisation of these parameters.

Numerical experiments universally indicate that the fastest multigrid solvers are obtained when ζ_σ is sufficiently large ($\zeta_\sigma \gtrsim 10$), beyond which the precise value of ζ_σ plays no further role in achieving the best performance; in other words, one can essentially make ζ_σ arbitrarily large. Since ζ_σ implements a weighting of the stress divergence operator on the diagonal block, when ζ_σ is sufficiently large the corresponding equations are nearly exactly satisfied by the least squares solution—intuitively, the smoothing effects of the Laplace-like viscous operator $\mathcal{A}$ should be prioritised. Note that this implies that the corresponding unknowns could be eliminated (essentially via the Schur complement of $\mathcal{A}_{ii}$), thereby reducing computational cost of the least squares solver, albeit with more algorithmic bookkeeping. On the other hand, in our numerical experiments we observed that the (pre)computation of the smoother Q has negligible cost compared to the overall multigrid solver. Consequently, and mainly for algorithmic simplicity, we have kept the non-eliminated form and apply a large value of ζ_σ . In particular, we universally set $\zeta_\sigma = 128$, across every parameter combination and every test problem, and exclude ζ_σ from further consideration.¹¹

To optimise for the remaining parameters $(\zeta_u, \omega_u, \omega_p)$, here we adopt an approach based on two-grid local Fourier analysis [14–16], detailed further in Appendix B. In essence, this procedure evaluates the performance of a two-level multigrid method, as applied to an infinite uniform Cartesian grid spanning all of $\mathbb{R}^d$, on an input function space spanned by functions of the form $\mathbf{x} \mapsto \exp(i\theta \cdot \mathbf{x}/h)$, where $i^2 = -1$. The output is a “two-grid LFA convergence factor” $\rho(\zeta_u, \omega_u, \omega_p)$, a predicted multigrid convergence rate such that the error in solving $A\mathbf{x} = \mathbf{b}$ is reduced by a factor of ρ per iteration. A corresponding search procedure then seeks out the set of parameters for which convergence is fastest, i.e., $\rho(\zeta_u, \omega_u, \omega_p)$ is smallest. Given the nature of these parameters—specifically that ω_u and ω_p are damping/overrelaxation parameters, while the balancing algorithms are designed so that ζ_u is unit magnitude—we expect a (local) minimum near the point $(1, 1, 1)$; the search algorithm applies a heuristic coordinate descent method to approximately find this minimum, denoted in the following by $\rho_\star$, and the corresponding tuned parameters $(\zeta_u, \omega_u, \omega_p)$. Note that two-grid LFA serves primarily as a guiding heuristic, since it optimises performance under the idealised assumptions of an infinite, constant-coefficient uniform grid and an exact coarse-grid solve. Nevertheless, the numerical experiments in Section 4 demonstrate that these LFA-tuned parameters transfer effectively to more complex settings, including nontrivial boundary conditions, high-contrast variable viscosity coefficients, and nonuniform meshes.

For every combination of polynomial degree $\varphi \in \{1, 2, 3\}$ and spatial dimension $d \in \{2, 3\}$, we consider three cases: (i) a steady-state standard-form Stokes problem, in which $\rho \equiv 0$, $\mu \equiv 1$, and $\gamma = 0$; (ii) a steady-state stress-form Stokes problem, in which $\rho \equiv 0$, $\mu \equiv 1$, and $\gamma = 1$; and (iii) an unsteady Stokes problems of infinite Reynolds number, in which $\rho \equiv 1$ and $\mu \equiv 0$ (γ thus irrelevant). For each of these 18 configurations, we apply the two-grid LFA method and at the same time, evaluate ρ in a neighbourhood of the minimum $\rho_\star$. The results of this study are shown in Figs. 2 and 3 for the 2D and 3D cases, respectively:

- The illustrated surfaces correspond to the level set $\{\rho(\zeta_u, \omega_u, \omega_p) = \rho_\star^{1/1.1}\}$ and indicate a region of near-optimal parameters. Any set of parameters inside this region yields a two-grid LFA-predicted multigrid convergence rate no more than $\rho_\star^{1/1.1}$; equivalently, they yield a multigrid iteration count at most 10% above the LFA-predicted optimal.
- The bounding box of the near-optimal region provides an indication of how finely-tuned the parameters need to be. For example, the parameter ζ_u often has a relatively wide window, whereas ω_u is often more tightly controlled around unity. This bounding box is indicated at the top of each figure; e.g., “ $\boxtimes = (0.4, 0.9, 0.7) \nearrow (0.9, 1.1, 1.1)$ ” means a parameter range $0.4 \leq \zeta_u \leq 0.9$, $0.9 \leq \omega_u \leq 1.1$, and $0.7 \leq \omega_p \leq 1.1$.
- Ultimately, we need to choose a set of parameters for use in practical settings. One approach could be to adopt the LFA-predicted best set of parameters. However, this optimum is sometimes situated off to one side and close to the boundary of the near-optimal region. Instead, we have chosen smoother parameters that better represent the average over all good parameters, i.e., at most 5% above optimal; as such, we define the overall optimum as the centroid of the region $\{\rho(\zeta_u, \omega_u, \omega_p) \leq \rho_\star^{1/1.05}\}$. In Figs. 2 and 3, this optimum is illustrated by the ball $\bullet$, whose coordinates are given at the top of each figure.

Appendix B provides further details on the LFA optimisation algorithm that adaptively explores the parameter space, prioritising areas where ρ is small. More sophisticated approaches include the robust optimisation methods recently developed by Brown et al. [16]; these methods specifically target multigrid optimisation problems, including the possibility of incorporating derivative and subgradient relations specific to LFA, and could lead to substantial speedups compared to the simple search algorithm used in this work.

We conclude this section by referring back to Algorithm 2 and noting that, for the remainder of this paper and its numerical experiments, we apply the optimal smoothing parameters as defined by the $\bullet$ coordinates in Figs. 2 and 3. In particular, the left (resp., middle) column of Figs. 2 and 3 defines the parameters $(\zeta_u^0, \omega_u^0, \omega_p^0)$ for steady-state standard-form (resp., stress-form) Stokes problems, while the right column defines $(\zeta_u^\infty, \omega_u^\infty, \omega_p^\infty)$.

¹¹ The value of 128 carries no particular meaning, other than being one order magnitude greater than the aforementioned cutoff and not so excessive that it impacts numerical conditioning.

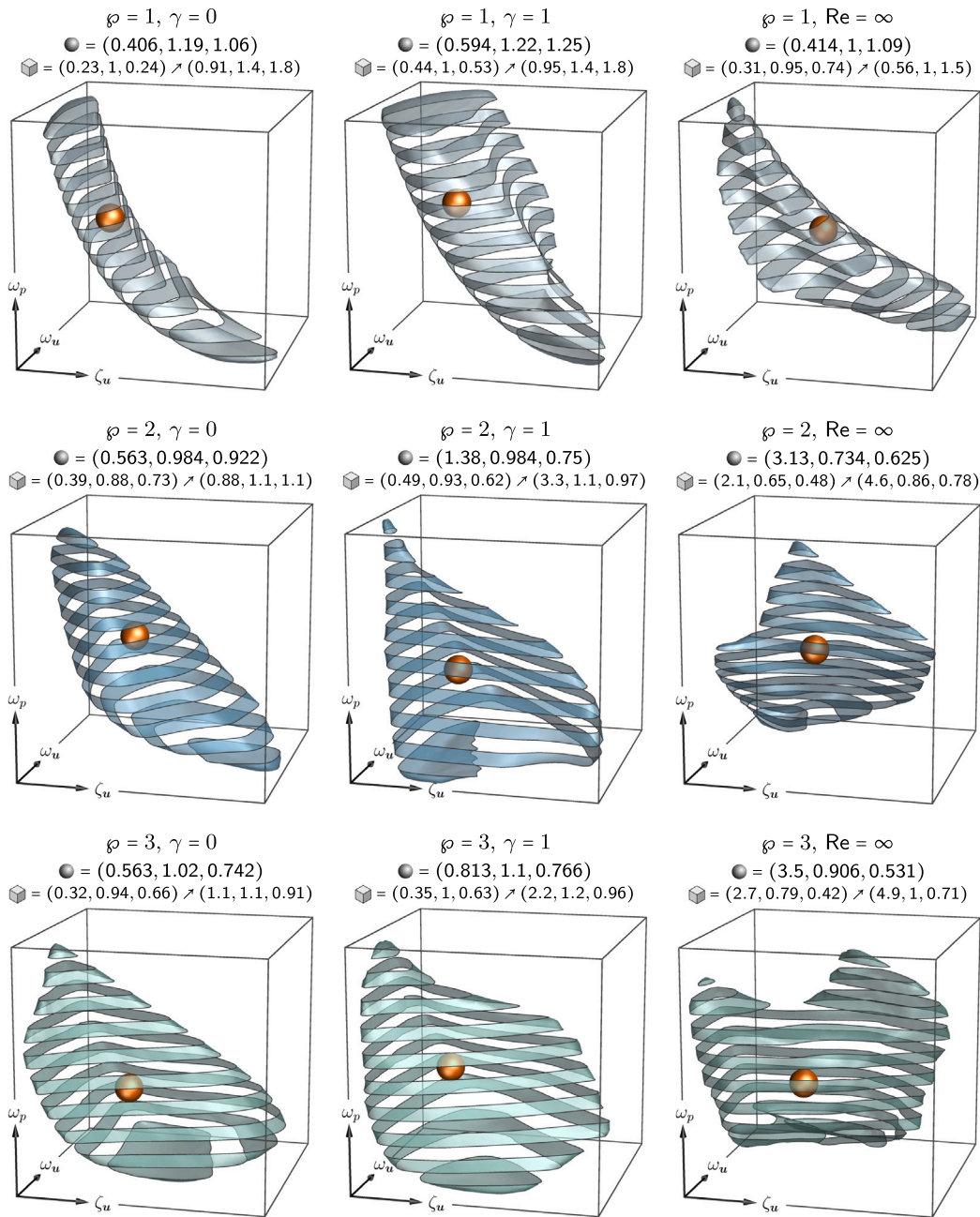


Fig. 2. Regions of near-optimal smoother parameters for 2D steady-state standard-form Stokes problems (left column), steady-state stress-form Stokes problems (middle column), and unsteady vanishing-viscosity Stokes problems (right column), for each $\phi \in \{1, 2, 3\}$. Here, “near-optimal” means the multigrid iteration count is at most 10% above optimal, according to the predictions of two-grid local Fourier analysis, as outlined in Section 3.5 and detailed further in Appendix B. The approximate centroid $\bullet$ and bounding box $\square$ of each region is indicated at the top of each panel; see Section 3.5 for their definition.

4. Numerical experiments

In this section, we examine multigrid performance on a variety of Stokes problems in two and three dimensions and across multiple grid sizes, for a variety of polynomial degrees ϕ and boundary condition types. The metric used to assess multigrid efficiency concerns the convergence rate of the corresponding multigrid V-cycle left-preconditioned GMRES solver. Specifically, for each test case, we randomly generate a vector b and solve $Ax = b$ with zero initial guess, $x_0 \equiv 0$. Each iteration k of GMRES yields an improved

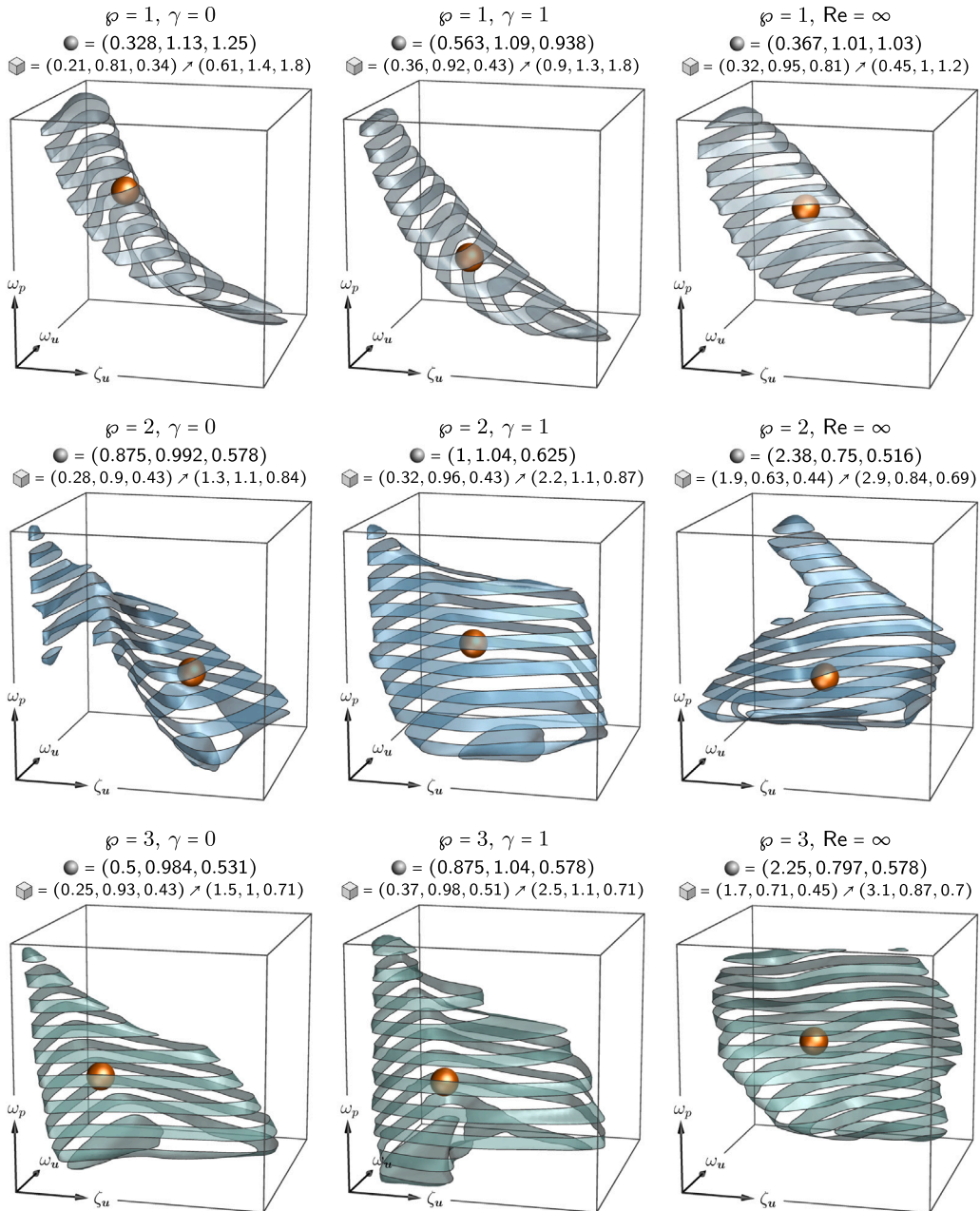


Fig. 3. Regions of near-optimal smoother parameters for 3D steady-state standard-form Stokes problems (left column), steady-state stress-form Stokes problems (middle column), and unsteady vanishing-viscosity Stokes problems (right column), for each $\phi \in \{1, 2, 3\}$. Here, “near-optimal” means the multigrid iteration count is at most 10% above optimal, according to the predictions of two-grid local Fourier analysis, as outlined in Section 3.5 and detailed further in Appendix B. The approximate centroid $\bullet$ and bounding box $\boxtimes$ of each region is indicated at the top of each panel; see Section 3.5 for their definition.

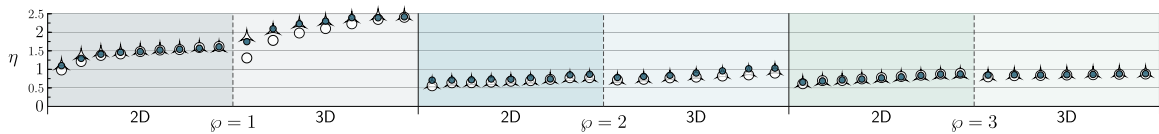


Fig. 4. Multigrid solver performance for the single-phase steady-state standard-form Stokes problem considered in Section 4.1. For each configuration (2D or 3D, polynomial degree φ , and boundary condition type), the corresponding sequence of symbols plot the numerically determined speed η on grid sizes $n \times n (\times n)$ where $n = 4, 8, 16, 32, \dots$, from left-to-right. Periodic, Dirichlet, and stress boundary conditions are denoted by Δ , $\bullet$, and $\circ$, respectively.

solution x_k and corresponding residual norm r_k ; typically, r_k is a nearly-constant fraction of r_{k-1} so that $r_k \approx \rho^k r_0$, where ρ is the convergence rate.¹² Our primary performance metric is then defined by

$$\eta := \frac{\log 0.1}{\log \rho}. \quad (14)$$

Here, η represents the expected number of iterations needed to reduce the residual by a factor of 10; e.g., $k\eta$ represents the number of iterations to reduce it by 10^k . We prefer to use η instead of ρ as the former is more intuitively useful [2]; e.g., a two-fold reduction in η coincides with a two-fold reduction in iteration count, and so, all else being equal, two times faster.

For each prototype Stokes problem, we consider polynomial degrees $\varphi \in \{1, 2, 3\}$ and test on grid sizes ranging from 4×4 up to 1024×1024 in 2D, and $4 \times 4 \times 4$ up to $256 \times 256 \times 256$ in 3D. We also test against three types of boundary conditions: (i) periodic, in which both $\mathbf{u}$ and p are periodic on $\partial\Omega$; (ii) Dirichlet, in which $\mathbf{u}$ is prescribed on $\partial\Omega$ and p is free; and (iii) stress, in which $\mathbf{u}$ and p are coupled via the specification of $\sigma \cdot \mathbf{n}$ on $\partial\Omega$. We test all three kinds as they can stress the numerics in rather different ways: (i) periodic boundary conditions are arguably the easiest to handle, and focus on the bulk-domain multigrid smoothing behaviour; (ii) velocity Dirichlet boundary conditions are enforced only weakly, via penalty operators that have very different spectral and smoothing characteristics compared to the bulk domain; and (iii) stress boundary conditions, which are not penalised at all, yet in some sense are a blend of Dirichlet boundary conditions on pressure and Neumann boundary conditions on the velocity field.

Our numerical results are presented through graphs of the form shown in Fig. 4. Each column corresponds to a specific spatial dimension and polynomial degree and contains three sequences of markers, one for each boundary condition type: symbols Δ , $\bullet$, and $\circ$ denote periodic, Dirichlet, and stress boundary conditions, respectively. From left-to-right, the markers in each sequence plot the experimentally-determined multigrid convergence speed η as measured on a grid of size $n \times n (\times n)$, where $n = 4, 8, 16, 32, \dots$, in that order.

4.1. Single-phase, steady-state, standard-form

We begin with perhaps the simplest case, that of a single-phase steady-state standard-form Stokes problem on the unit rectangular domain $\Omega = (0, 1)^d$ with $\mu \equiv 1$, using uniform Cartesian grids. Fig. 4 compiles the corresponding numerical results. For each combination of spatial dimension, polynomial degree, and boundary condition, we see that η essentially plateaus, i.e., the multigrid iteration count required to reduce the residual by a fixed factor remains bounded as the grid is refined, as should be the case for an effective multigrid method. In particular, for polynomial degrees $\varphi = 2$ and $\varphi = 3$, a 10^{10} -fold reduction in the residual requires 5 to 9 multigrid iterations—these speeds match those of classical geometric multigrid methods applied to Poisson problems. On the other hand, the case of $\varphi = 1$ yields noticeably larger iteration counts, e.g., with η approaching 2.5 in 3D. This outcome is observed in most of our numerical results: for the same residual reduction threshold, the case of $\varphi = 1$ requires 2–3 times as many outer solver iterations compared to the cases of $\varphi \in \{2, 3\}$. In this regard, $\varphi = 1$ is a kind of outlier; on the other hand, the corresponding discrete solution is low-order (indeed, it only yields first-order pressure fields) and thus may not require many digits of accuracy, thereby allowing for a looser convergence criterion. Note also that in the context of high-order LDG methods, $\varphi = 1$ is perhaps only of peripheral interest; further comments are given in the concluding remarks.

4.2. Single-phase, steady-state, stress-form

We next test multigrid performance when considering steady-state Stokes equations in stress form, i.e., when $\gamma = 1$. We consider again a single-phase problem on $\Omega = (0, 1)^d$ with $\mu \equiv 1$ and uniform Cartesian grids. Fig. 5 compiles the corresponding numerical results. Compared to the standard-form case, we observe essentially the same multigrid efficiency, though with a mild speedup in some configurations; e.g., for $\varphi = 3$ in 2D, η plateaus to around 0.7 (resp., 0.9) for the stress-form (resp., standard-form) cases, see the second-from-right column of Fig. 5 (resp., Fig. 4).

¹² The components of b are i.i.d. randomly drawn from the uniform distribution on $[-1, 1]$ so that the test problem is likely to contain modes for which convergence is the slowest. The residual norm is the same as that used by left-preconditioned GMRES, i.e., $r_k := \|V A x_k - V b\|_2$. On a log-linear plot, the data points (k, r_k) are well approximated by a straight line of negative slope; simple linear regression is used to compute the corresponding convergence rate ρ . Our numerical tests typically have 5–10 such data points, except when convergence is so fast that just a few data points are collected before reaching the limits of double-precision arithmetic. Transient behaviour is sometimes observed over the first 1–2 iterations, after which the preconditioned residual norm closely tracks that of the true residual, each with steady geometric decay.

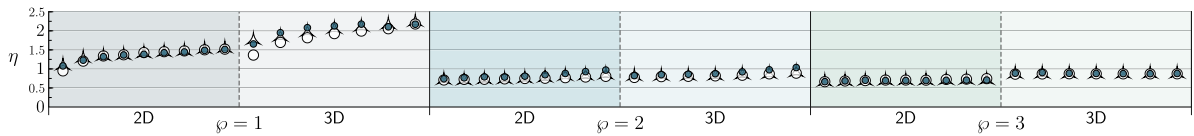


Fig. 5. Multigrid solver performance for the single-phase steady-state stress-form Stokes problem considered in Section 4.2. For each configuration (2D or 3D, polynomial degree φ , and boundary condition type), the corresponding sequence of symbols plot the numerically determined speed η on grid sizes $n \times n (\times n)$ where $n = 4, 8, 16, 32, \dots$, from left-to-right. Periodic, Dirichlet, and stress boundary conditions are denoted by $\triangle$, $\star$, and $\circ$, respectively.

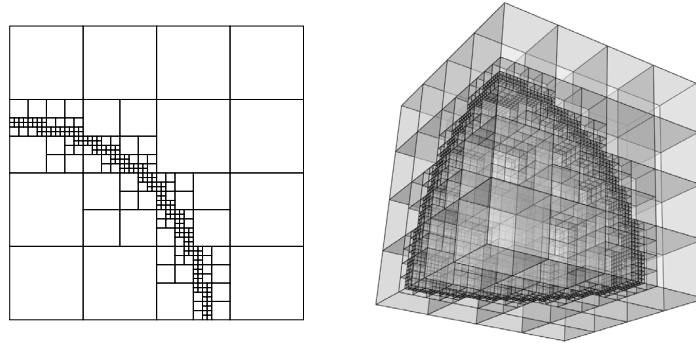


Fig. 6. Examples of the nonuniform quadtree (left) and octree (right) used in the adaptive mesh refinement problem in Section 4.3. Each example corresponds to a base level grid size of $n = 4$, refined up to four levels around a circular arc or spherical shell of the same radius.

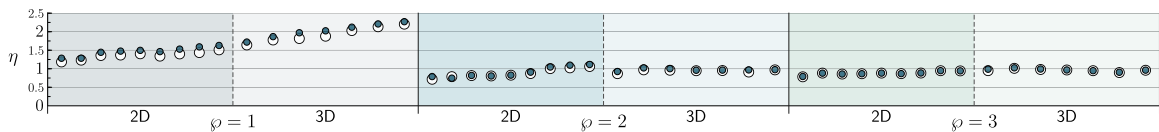


Fig. 7. Multigrid solver performance for the quadtree/octree nonuniform mesh problem considered in Section 4.3. For each configuration (2D or 3D, polynomial degree φ , and boundary condition type), the corresponding sequence of symbols plot the numerically determined speed η on meshes with a base level grid size of $n \times n (\times n)$ where $n = 4, 8, 16, 32, \dots$, from left-to-right. Dirichlet and stress boundary conditions are denoted by $\bullet$ and $\star$, respectively.

4.3. Adaptive mesh refinement

Our third test problem is similar to the first (in Section 4.1), except that we now apply it to a nonuniform adaptively refined mesh. The mesh has up to four levels of nested refinement around a circular arc (in 2D) or spherical shell of the same radius (in 3D), as illustrated in Fig. 6. Owing to the rapid rate of refinement, this problem has considerably different stencil and sparsity structure. Compiled in Fig. 7 are the corresponding experimentally-determined multigrid solver speeds.¹³ Examining Fig. 7, we observe similar multigrid speeds as to prior test problems, though with a modest $\approx 25\%$ increase in η that can be attributed to the more complex mesh geometry, stencils, and sparsity structure.

4.4. Multiphase, steady-state, stress-form

We now consider a more challenging case of a high-contrast multiphase Stokes problem, wherein the viscosity μ exhibits a jump several orders in magnitude across an embedded interface. Let $\Omega = (0, 1)^d$ be divided into an interior phase, $\Omega_0 = (\frac{1}{4}, \frac{3}{4})^d$, and an exterior phase, $\Omega_1 = \Omega \setminus \overline{\Omega_0}$, each with viscosity μ_i . Note that in this multiphase example, there is an implicit specification of interfacial jump conditions, on both velocity and stress, on $\Gamma = \partial\Omega_0 \cap \partial\Omega_1$. Two representative viscosity ratios are examined, $\mu_0/\mu_1 = 10^4$ and $\mu_0/\mu_1 = 10^{-4}$; Fig. 8 compiles the corresponding numerical results. Compared to the prior test problems, we see a greater dependence on the applied domain boundary conditions. For example, when the interior phase is a factor 10^4 more viscous than the exterior, we see from the top row of Fig. 8 that Dirichlet boundary conditions generally lead to larger η compared to the

¹³ The largest such problem corresponds to the 3D case wherein the base level $256 \times 256 \times 256$ mesh is refined by four more levels, to a total of ≈ 88 million elements; for $\varphi = 3$, this corresponds to a velocity/pressure solution vector of ≈ 19 billion degrees of freedom.

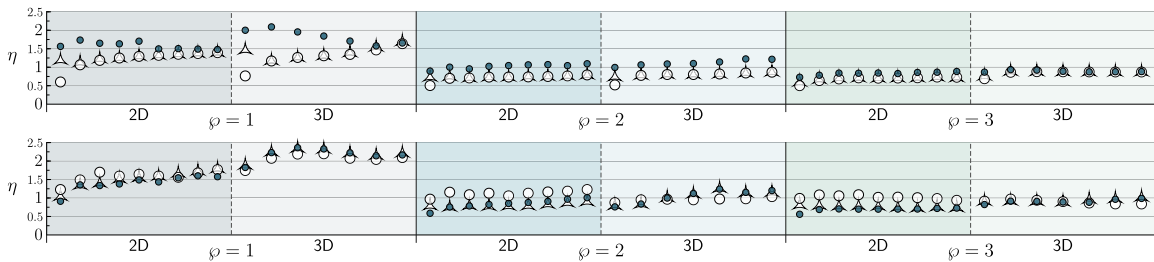


Fig. 8. Multigrid solver performance for the multiphase steady-state stress-form Stokes problems considered in Section 4.4; the top (resp., bottom) row corresponds to an interior phase with viscosity a factor 10^4 (resp., 10^{-4}) that of the exterior phase. For each configuration (2D or 3D, polynomial degree φ , and boundary condition type), the corresponding sequence of symbols plot the numerically determined speed η on grid sizes $n \times n (\times n)$ where $n = 4, 8, 16, 32, \dots$, from left-to-right. Periodic, Dirichlet, and stress boundary conditions are denoted by $\triangle$, $\bullet$, and $\circ$, respectively.

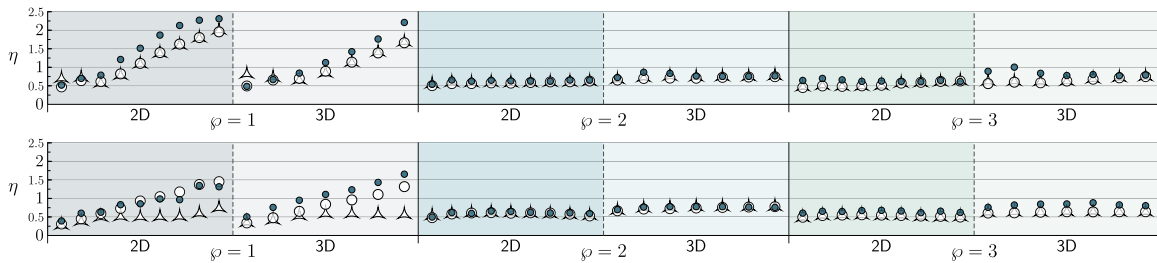


Fig. 9. Multigrid solver performance for the single-phase unsteady standard-form Stokes problems considered in Section 4.5; the top (resp., bottom) row corresponds to an effective Reynolds number of 10^2 (resp., 10^4). For each configuration (2D or 3D, polynomial degree φ , and boundary condition type), the corresponding sequence of symbols plot the numerically determined speed η on grid sizes $n \times n (\times n)$ where $n = 4, 8, 16, 32, \dots$, from left-to-right. Periodic, Dirichlet, and stress boundary conditions are denoted by $\triangle$, $\bullet$, and $\circ$, respectively.

other boundary conditions; this is most pronounced with $\varphi = 1$ and to a lesser extent on $\varphi = 2$, though with negligible difference for $\varphi = 3$. Conversely, when the exterior phase is a factor 10^4 more viscous than the interior phase, we see from the bottom row of Fig. 8 that sometimes stress boundary conditions lead to mildly slower convergence. Notwithstanding these effects, we observe fast multigrid convergence that remains stable under grid refinement; this favourable outcome is attributed in part to the use of viscosity-upwinded numerical fluxes, which bias the interfacial LDG discretisation to the mutual benefits of multigrid speed and high-order accuracy [9] as well as to the balancing process described in Section 3.2. Viscosity ratios of $10^{\pm 2}$, $10^{\pm 6}$, and $10^{\pm 8}$ have also been tested, with essentially identical results to those shown in Fig. 8.

4.5. Single-phase, unsteady, standard-form

So far we have focused on steady-state Stokes problems; our last set of test cases examine unsteady Stokes problems. As discussed in Section 3.3, the amended system includes a $\frac{\rho}{\delta}$ -weighted mass matrix whose strength, relative to the viscous operator, determines the nature of the unsteady Stokes problem: in one extreme, the viscous operator dominates and we expect to observe multigrid behaviour akin to the steady-state case; in the other extreme, the viscous operator is negligible so that we are in effect solving a flux-form Poisson problem with an effective ellipticity coefficient equal to $\frac{\delta}{\rho}$. In between, the smoother parameters $(\zeta_u, \omega_u, \omega_p)$ are blended; recall that the blending amount may actually change across the domain and across the multigrid hierarchy, since it depends on the local grid-dependent relative strengths of the ρ -weighted and μ -weighted operators.

In the specific examples of this subsection, we consider two single-phase, unsteady, standard-form Stokes problems on $\Omega = (0, 1)^d$: one with $\mu = 10^{-2}$ and the other with $\mu = 10^{-4}$. In both cases, we set $\rho = 1$ and define $\delta = 0.1 h$ where h is the element size of the primary-level uniform Cartesian grid; this choice of δ represents a typical scenario of applying the unsteady Stokes equations in a temporal integration method having unit-order CFL number. These two problems correspond to an effective Reynolds number $\text{Re} = \rho UL / \mu$ of $\text{Re} = 10^2$ and $\text{Re} = 10^4$, respectively: the former (with $\mu = 10^{-2}$) represents a viscous-dominated case, but where the time-derivative term nevertheless influences solver characteristics; the latter (with $\mu = 10^{-4}$) represents a case wherein the time-derivative term dominates, though with progressively weaker influence as the primary mesh is refined. With this setup, Fig. 9 compiles the numerical results. For polynomial degrees $\varphi \in \{2, 3\}$, we observe relatively stable and fast multigrid solver performance for all kinds of boundary condition; in particular, η is often close to 0.5, i.e., just 5 multigrid GMRES iterations are needed to reduce the residual by ten orders of magnitude. Meanwhile, the case of $\varphi = 1$ exhibits more nuanced behaviour: for the high-Reynolds case with periodic boundary conditions, the multigrid solver performs well, reaching speeds of $\eta \approx 0.5$; however, for all other cases we observe an upwards trend in η . This is largely attributed to the fact that, as the grid is refined, eventually we obtain a perturbed unsteady Stokes problem, for which we have seen in prior examples (see Figs. 4, 5, 7 and 8) that $\eta \approx 2$ is typical.

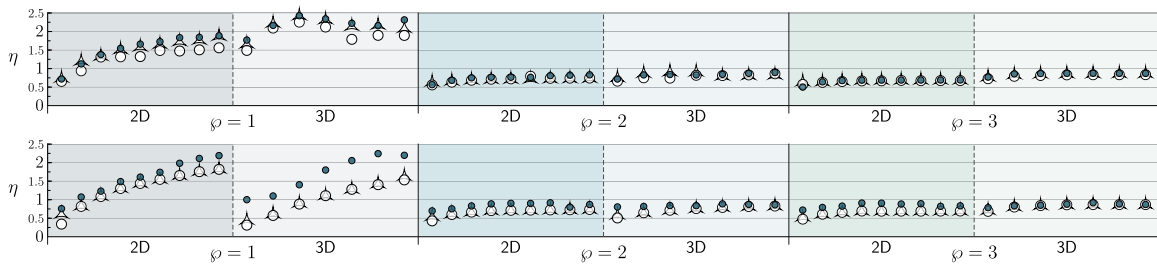


Fig. 10. Multigrid solver performance for the multiphase unsteady stress-form Stokes problems considered in Section 4.6; the top (resp., bottom) row corresponds to a gas droplet surrounded by liquid (resp., a liquid droplet surrounded by gas). For each configuration (2D or 3D, polynomial degree φ , and boundary condition type), the corresponding sequence of symbols plot the numerically determined speed η on grid sizes $n \times n$ ($n = 4, 8, 16, 32, \dots$), from left-to-right. Periodic, Dirichlet, and stress boundary conditions are denoted by Δ , $\bullet$, and $\circ$, respectively.

4.6. Multiphase, unsteady, stress-form

Our last two examples are in some regards the most challenging, as they involve density and viscosity coefficients that jump by 3 to 4 orders of magnitude across an embedded interface, thereby examining the ability to handle high-contrast coefficients in addition to the grid-dependent relative strengths of the time derivative and viscous operator terms. We consider two scenarios: a liquid bubble surrounded by gas, and a gas bubble surrounded by liquid, with $\rho_{\text{liquid}} = 1$, $\mu_{\text{liquid}} = 1$, $\rho_{\text{gas}} = 0.001$, and $\mu_{\text{gas}} = 0.0002$ (approximately accurate values in CGS units for ambient air and 100 cSt silicone oil). Here, the “bubble” refers to an interior rectangular phase, $\Omega_0 = (\frac{1}{4}, \frac{3}{4})^d$, while the exterior phase is $\Omega_1 = \Omega \setminus \overline{\Omega_0}$, with $\Omega = (0, 1)^d$. Note this problem also examines the multigrid solver’s ability to handle velocity and stress interfacial jump conditions across the embedded liquid–gas interface. As before, we set $\delta = 0.1h$ to represent, say, solving the incompressible Navier–Stokes equations via an explicit treatment of the advection term and implicit treatment of the viscous term. With this setup, Fig. 10 compiles the corresponding numerical results. As in other test problems, the low-order case of $\varphi = 1$ is a kind of outlier; excluding it, we observe multigrid speeds that attain those of classical geometric multigrid methods: essentially grid-independent convergence of around 5 to 9 solver iterations to reduce the residual by ten orders of magnitude.

5. Concluding remarks

The two main LDG approaches for Stokes problems are that of equal-degree (whereby $\mathbf{u}_h \in V_{\varphi}^d$ and $p_h \in V_{\varphi}$) and mixed-degree (whereby $\mathbf{u}_h \in V_{\varphi}^d$ and $p_h \in V_{\varphi-1}$). Both yield optimal order accuracy for the velocity field, i.e., order $\varphi + 1$ in the maximum norm; and, except for idealised cases, both yield order φ accuracy in the maximum norm for the pressure field.¹⁴ However, the equal-degree approach often leads to significant pressure errors near nontrivial boundaries and interfaces, whereas the mixed-degree approach yields considerably more accurate and cleaner pressure fields, see Section 2.1. But, and perhaps superseding aspects of accuracy, is the complication of pressure penalty stabilisation. In general, pressure stabilisation is required when using the equal-degree approach, but there is no known efficient way to implement it for stress boundary conditions nor interfacial jump conditions, see Section 2.1. In contrast, the mixed-degree approach does not require pressure stabilisation, thereby eliminating many subtle implementation details.

Besides the development of a mixed-degree LDG framework for multiphase Stokes problems, in this paper we focused mainly on multigrid solvers. In particular, we developed a multigrid smoother that is identical to multicoloured element-wise block Gauss–Seidel, except the block diagonal inverses A_{ii}^{-1} are replaced by approximate inverses Q_i . In turn, this inverse is computed through a weighted least squares approximation of the form $AQ \approx \mathbb{I}$, controlled by four tunable parameters, $\zeta_{\sigma}, \zeta_u, \omega_u$, and ω_p . The ζ parameters control the weights in the least squares problem, while the ω parameters allow the velocity and pressure variables to be damped/overrelaxed by different amounts. Intuitively, their combined purpose is to simultaneously and maximally damp the high frequency error components of the velocity and pressure variables, in part by balancing the relative strengths of the stress divergence and velocity divergence operators. In addition, the smoother applies a balancing algorithm to treat two critical aspects of Stokes problems: (i) the mesh-dependent, unequal scaling of the viscous, pressure gradient, velocity divergence, and density-weighted temporal derivative operators; and (ii) for high-contrast multiphase problems, viscosity and/or density coefficients varying over multiple orders in magnitude. In effect, this balancing process augments the weighted least squares formulation so that the aforementioned smoother parameters $(\zeta_{\sigma}, \zeta_u, \omega_u, \omega_p)$ can be applied independent of physical parameters, element sizes, mesh connectivity, etc.

The smoother parameters can be tuned in various ways. Fortunately, ζ_{σ} can be quickly eliminated from consideration, essentially by choosing it arbitrarily large, see Section 3.5. For the remaining three parameters, we used an optimisation approach based on

¹⁴ It is possible for the equal-degree approach to yield order $\varphi + 1$ in the maximum norm, for both velocity and pressure, but only in some idealised cases such as periodic boundary conditions on uniform grids using one-sided fluxes.

Table 1

Relative cost of computing Q_i vs. A_{ii}^{-1} , for canonical Stokes problems on uniform d -dimensional Cartesian grids.

Stokes problem	d	ℓ	Relative cost mn^2/n^3		
			$\wp = 1$	$\wp = 2$	$\wp = 3$
Standard form	2D	4	1.89	1.36	1.20
	3D	6	1.72	1.20	1.08
Stress form	2D	6	2.33	1.55	1.29
	3D	12	2.44	1.40	1.16

two-grid local Fourier analysis (LFA). The outcome is an optimised set of parameters that depend solely on the spatial dimension d , the polynomial degree $\wp$, and the form of the Stokes problem (whether it is steady-state standard-form, steady-state stress-form, or unsteady vanishing-viscosity); these parameters are listed in Figs. 2 and 3. Extensive numerical tests—over a variety of steady and unsteady Stokes problems, in 2D and 3D, across multiple grid sizes, on polynomial degree $\wp \in \{1, 2, 3\}$, and with different boundary conditions—demonstrated that the resulting multigrid solver matches the speed of Poisson-style geometric multigrid methods, e.g., 5 to 10 iterations to reduce the residual by a factor of 10^{10} (putting aside the low-order outlier of $\wp = 1$).

Regarding the smoother's computational efficiency, this was designed from the outset to be the same or comparable to the equivalent block Gauss–Seidel method. In particular, a single iteration is directly comparable to standard block Gauss–Seidel in per-iteration cost, memory footprint, and scalability. Meanwhile, the dominant component of building Q_i is a least squares problem of size $m \times n$, where $n = d(\wp + 1)^d + \wp^d$ is the elemental block size and $m = n + d\ell$, where ℓ equals the number of nonzero off-diagonal blocks (e.g., a standard-form Stokes problem in 3D has a 7-element stencil, so $\ell = 6$). To leading order, the corresponding solution requires $\mathcal{O}(mn^2)$ floating point operations, whereas computing A_{ii}^{-1} requires $\mathcal{O}(n^3)$ operations; Table 1 provides a rough indication of their relative cost by tabulating the ratio $(mn^2)/n^3$ for canonical uniform-grid Stokes problems. In practice, the cost of precomputing Q is almost negligible—e.g., to reduce the residual by a factor of 10^{10} , say, the time spent precomputing Q is no more than 10% of the overall multigrid solver time. Additionally, note that each block of Q can be precomputed independently of every other block, making its construction trivially parallelisable and scalable.

As observed in our numerical results, the case of $\wp = 1$ is an outlier in the sense it requires 2–3 times more iterations compared to the multigrid solvers for $\wp > 1$. However, this comparison is perhaps incomplete as it overlooks some application details; e.g., low-order solutions are unlikely to require many digits of accuracy, so the same solver tolerances need not apply. Some numerical experiments (not shown here) suggest that the $\wp = 1$ smoother parameters could be tweaked for an additional 10%–20% speedup; in other words, the predictions of two-grid LFA appear to be somewhat suboptimal for $\wp = 1$. To that end, a three-grid LFA method may have better predictive power in determining the absolute-best smoother parameters. Alternatively, generalising the approach used here to an overlapping-block smoother method, similar to Vanka-type smoothers, offers a more promising avenue for building effective multigrid solvers in the $\wp = 1$ case; we leave this exploration open for future study.

Finally, we mention some extensions and future research possibilities. We considered here an LDG modal polynomial basis. This choice mainly impacted the design and construction of the smoother, where it was used to easily select a few rows per off-diagonal block of $A_{:,i}$ (i.e., the rows corresponding to the constant-mode lowest-degree modal coefficient of the stress divergence operator). If a nodal basis was preferred, one could make a temporary nodal-to-modal change of basis, build the block-diagonal smoother Q , and then map it back to the nodal basis. An alternative approach might be to preserve the nodal basis, but select some other subset of rows from the off-diagonal blocks—e.g., rows corresponding to collocated nodes on neighbouring elements—though this approach would likely require a new set of optimised smoother parameters. On a similar note, we expect these ideas could be adapted to other LDG settings, e.g., triangular/tetrahedral meshes, total-degree polynomial bases, etc., and perhaps other kinds of DG discretisations besides LDG. Meanwhile, we focused here on V-cycles with 3 pre-smoothing and 3 post-smoothing steps. Other possibilities include V- and W-cycles with differing pre- and post-smoother counts, F-cycles, etc. As suggested by Brown et al. [16], even faster multigrid solvers could be developed by applying one set of smoother parameters in the pre-smoother and a different set of parameters in the post-smoother.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research's Applied Mathematics Competitive Portfolios program under Contract No. DE-AC02-05CH11231, and by a U.S. Department of Energy, Office of Science, Early Career Research Program award. Some computations used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy, Office of Science User Facility operated at Lawrence Berkeley National Laboratory.

Appendix A. Local discontinuous Galerkin framework

We derive here a mixed-degree LDG framework for multiphase variable-viscosity Stokes problems. The construction closely follows that of prior work [1], except herein the pressure field has polynomial degree one less than the velocity field. At the same time, the framework is partly based on the LDG schemes initially developed by Cockburn et al. [7], extending them to multiphase variable-viscosity problems, incorporating interfacial jump conditions, viscosity-upwinded fluxes, and additional types of domain boundary conditions.

For simplicity of presentation, we consider meshes $\mathcal{E}$ arising from Cartesian grids, quadrees, or octrees. To establish some notation, *intrapphase* mesh faces are those shared by two elements in the same phase; *interphase* mesh faces are those shared by two elements in differing phases (and thus located on Γ_{ij} for some i, j); and *boundary* mesh faces are those situated on $\partial\Omega$. Each face has an associated unit normal vector: interphase faces adopt the same unit normal as their corresponding interface Γ_{ij} ; boundary faces adopt the natural outwards-pointing domain boundary unit normal; while intrapphase faces are all positively oriented in the same way, e.g., $\mathbf{n} = \hat{\mathbf{x}}$ on every vertical intrapphase face. The notation $\llbracket \cdot \rrbracket$ denotes the jump of a quantity across an interface and/or mesh face: $\llbracket u \rrbracket := u^- - u^+$ where $u^\pm(x) = \lim_{\epsilon \rightarrow 0^\pm} u(x \pm \epsilon \mathbf{n})$ evaluates the trace of u on either side. In addition, for an element $E \in \mathcal{E}$, let $\mathbf{n}_E$ denote the outwards pointing normal on the boundary of the element; define $\chi(E)$ to be the phase of E (so that $E \subseteq \Omega_{\chi(E)}$); and for any discontinuous piecewise polynomial function u , let $u|_E$ denote the restriction of u to E (so that the quantity $u|_E$ is well defined on ∂E).

In the first step of the LDG construction, a discrete approximation to $\nabla \mathbf{u}$ is defined. Given $\mathbf{u} \in V_{\mathcal{E}}^d$, we define $\boldsymbol{\tau} \in V_{\mathcal{E}}^{d \times d}$ to be the discrete gradient of $\mathbf{u}$ via a “strong-weak” form, so that

$$\int_E \boldsymbol{\tau} : \boldsymbol{\omega} = \int_E \boldsymbol{\omega} : \nabla \mathbf{u} + \int_{\partial E} (\hat{\mathbf{u}}_{\chi(E)} - \mathbf{u}|_E) \cdot \boldsymbol{\tau}|_E \cdot \mathbf{n}_E \quad (15)$$

holds for every element E and every test function $\boldsymbol{\omega} \in V_{\mathcal{E}}^{d \times d}$. Here, the numerical flux $\hat{\mathbf{u}}_\chi$ is defined by

$$\hat{\mathbf{u}}_\chi := \begin{cases} \mathbf{u}^- & \text{on any intrapphase face,} \\ \lambda \mathbf{u}^- + (1 - \lambda)(\mathbf{u}^+ + \mathbf{g}_{\chi j}) & \text{on } \Gamma_{\chi j} \text{ if } \chi < j, \\ \lambda(\mathbf{u}^- - \mathbf{g}_{i\chi}) + (1 - \lambda)\mathbf{u}^+ & \text{on } \Gamma_{i\chi} \text{ if } i < \chi, \\ \mathbf{u}^- & \text{on } \Gamma_N, \\ \mathbf{g}_\partial & \text{on } \Gamma_D. \end{cases} \quad (16)$$

On intrapphase faces, we have chosen a minimal dissipation LDG scheme [7], whereby the numerical flux is purely one-sided, being sourced from the “left” element, i.e., $\hat{\mathbf{u}} = \mathbf{u}^-$. On interphase faces, the numerical flux is multivalued so as to implement the interfacial jump condition $\llbracket \mathbf{u} \rrbracket = \mathbf{g}_{ij}$; whenever the weak formulation of Eq. (15) “reaches across” an interface Γ_{ij} to evaluate the trace of $\mathbf{u}$ on the other side, the result is offset by the jump data $\mathbf{g}_{ij}$ so as to appropriately account for the intended discontinuity in the solution. The blending of the trace values on either side of the interface is controlled by the convex weighting parameter $\lambda \in [0, 1]$; its specification allows for the implementation of viscosity-upwinded fluxes [9].

A routine calculation (see [1] for an example) shows that, upon summing Eq. (15) over every mesh element, the variational form is equivalent to the statement that

$$\tau_{ij} = G_j u_i + J_{\mathbf{g}, ij},$$

where $G = (G_1, \dots, G_d) : V_{\mathcal{E}} \rightarrow V_{\mathcal{E}}^d$ is a discrete gradient operator implementing the numerical fluxes of Eq. (16), $\tau_{ij} \in V_{\mathcal{E}}$ denotes the (i, j) th component of $\boldsymbol{\tau}$, u_i denotes the i th component of $\mathbf{u}$, and $J_{\mathbf{g}} \in V_{\mathcal{E}}^{d \times d}$ implements the lifting of any Dirichlet-like source data $\mathbf{g}_\partial$, $\mathbf{g}_{ij}$. (The precise definitions of G and $J_{\mathbf{g}}$ are summarised in Appendix A.1.)

In the second step of the LDG construction, a discrete approximation of $\boldsymbol{\sigma} = \mu(\boldsymbol{\tau} + \gamma \boldsymbol{\tau}^\top) - p \mathbb{I}$ is defined. Two aspects need attention: (i) the viscosity μ may vary spatially (and may possibly be a tensor, though for simplicity of presentation we do not consider that here); and (ii) the pressure field p is of polynomial degree one less than the velocity field. Both aspects can be concisely treated by performing an L^2 projection of $\mu(\boldsymbol{\tau} + \gamma \boldsymbol{\tau}^\top) - p \mathbb{I}$ onto $V_{\mathcal{E}}^{d \times d}$, resulting in the definition of $\boldsymbol{\sigma}$ as

$$\sigma_{ij} := M^{-1} M_\mu (\tau_{ij} + \gamma \tau_{ji}) - \delta_{ij} \mathbb{P}_{V_{\mathcal{E}}}(p) = M^{-1} M_\mu (G_j u_i + \gamma G_i u_j + J_{\mathbf{g}, ij} + \gamma J_{\mathbf{g}, ji}) - \delta_{ij} \mathbb{P}_{V_{\mathcal{E}}}(p). \quad (17)$$

Here, M_μ is a μ -weighted mass matrix relative to $V_{\mathcal{E}}$, while $\mathbb{P}_{V_{\mathcal{E}}}(p)$ is the projection of $p \in V_{\mathcal{E}-1}$ onto $V_{\mathcal{E}}$ and simply corresponds to injecting p into the higher-degree polynomial space.

In the third step, we consider the weak formulation for computing the discrete divergence of $\boldsymbol{\sigma}$. This proceeds similarly to the first step, except numerical fluxes are taken in the opposite direction. (For simplicity of presentation, the following numerical flux for $\boldsymbol{\sigma}$ is matrix-valued, however only the normal component is used.) Define $\boldsymbol{\xi} \in V_{\mathcal{E}}^d$ as the discrete divergence of $\boldsymbol{\sigma}$ so that

$$\int_E \boldsymbol{\xi} \cdot \boldsymbol{\omega} = - \int_E \boldsymbol{\sigma} : \nabla \boldsymbol{\omega} + \int_{\partial E} \boldsymbol{\omega}|_E \cdot \hat{\boldsymbol{\sigma}}_{\chi(E)} \cdot \mathbf{n}_E \quad (18)$$

holds for every element E and every test function $\omega \in V_{\varphi}^d$. Here, the numerical flux $\hat{\sigma}$ is defined by

$$\hat{\sigma}_{\chi} := \begin{cases} \sigma^+ & \text{on any intraphase face,} \\ (1 - \lambda) \sigma^- + \lambda (\sigma^+ + \mathbf{h}_{\chi j} \otimes \mathbf{n}) & \text{on } \Gamma_{\chi j} \text{ if } \chi < j, \\ (1 - \lambda) (\sigma^- - \mathbf{h}_{i\chi} \otimes \mathbf{n}) + \lambda \sigma^+ & \text{on } \Gamma_{i\chi} \text{ if } i < \chi, \\ \mathbf{h}_{\partial} \otimes \mathbf{n} & \text{on } \Gamma_N, \\ \sigma^- & \text{on } \Gamma_D. \end{cases} \quad (19)$$

Analogous to the numerical flux $\hat{\mathbf{u}}$, the interfacial jump condition $\llbracket \sigma \cdot \mathbf{n} \rrbracket = \mathbf{h}_{ij}$ is taken into account in Eq. (19) such that, whenever an element reaches across the interface, the neighbouring element's trace is offset by $\mathbf{h}_{ij}$ so as to appropriately account for the intended discontinuity in $\sigma \cdot \mathbf{n}$. Summing Eq. (18) over every mesh element, a routine calculation (see [1] for an example) shows that the variational form leads to

$$\xi_i = - \sum_{j=1}^d M^{-1} G_j^T M \sigma_{ij} + J_{\mathbf{h},i}, \quad (20)$$

where $\xi_i \in V_{\varphi}$ is the i th component of ξ and $J_{\mathbf{h}} \in V_{\varphi}^d$ (defined in Appendix A.1) implements the lifting of any Neumann-like source data $\mathbf{h}_{\partial}$, $\mathbf{h}_{ij}$. Note that ξ_i computes the discrete divergence of σ via the adjoint of the discrete gradient operator G ; the adjoint arises because the direction and weighting of the numerical fluxes for $\hat{\sigma}$ in Eq. (19) are opposite to those of $\hat{\mathbf{u}}$ in Eq. (16). To complete this step, we (temporarily) set the negative divergence of σ equal to the L^2 projection of the Stokes momentum equation source data $\mathbf{f}$; combining Eqs. (17) and (20) we therefore obtain, for every component i ,

$$\sum_{j=1}^d M^{-1} G_j^T M_{\mu} (G_j u_i + \gamma G_i u_j + J_{\mathbf{g},ij} + \gamma J_{\mathbf{g},ji}) - M^{-1} G_i^T M \mathbb{P}_{V_{\varphi}}(p) - J_{\mathbf{h},i} = \mathbb{P}_{V_{\varphi}}(\mathbf{f}_i). \quad (21)$$

One may identify the first term as a discrete approximation to $-\nabla \cdot (\mu(\nabla \mathbf{u} + \gamma \nabla \mathbf{u}^T))$ and the second as a mixed-degree discrete gradient of pressure.

We now turn to the LDG formulation for the Stokes divergence constraint. Given $\mathbf{u} \in V_{\varphi}^d$, we define $\varphi \in V_{\varphi-1}$ as a discrete divergence of $\mathbf{u}$ via a strong-weak form, so that

$$\int_E \varphi \omega = \int_E \omega \nabla \cdot \mathbf{u} + \int_{\partial E} \omega (\hat{\mathbf{u}}_{\chi(E)} - \mathbf{u}|_E) \cdot \mathbf{n}_E \quad (22)$$

holds for every element E and every test function $\omega \in V_{\varphi-1}$. Here, the same numerical flux $\hat{\mathbf{u}}$ is used, i.e., Eq. (16), but only the normal component is seen by the weak formulation in Eq. (22). Summing Eq. (22) over the entire mesh, it follows that

$$\varphi = \sum_{i=1}^d \tilde{G}_i u_i + J_{\mathbf{g},\mathbf{n}},$$

where $\tilde{G} = (\tilde{G}_1, \dots, \tilde{G}_d) : V_{\varphi} \rightarrow V_{\varphi-1}^d$ is a mixed-degree discrete gradient operator and $J_{\mathbf{g},\mathbf{n}} \in V_{\varphi-1}$ implements the lifting of the normal component of any Dirichlet-like source data $\mathbf{g}_{\partial}$, $\mathbf{g}_{ij}$. Note that $\tilde{G}$ maps the full-degree space to the lower-degree space of the pressure field; in fact, $\tilde{G}$ is identical to G except for an L^2 projection, i.e., $\tilde{G}_i = \mathbb{P}_{V_{\varphi-1}} G_i$. To complete this step, the negative discrete divergence of $\mathbf{u}$ is set equal to the L^2 projection of the Stokes divergence equation source data f , thereby obtaining

$$- \sum_{i=1}^d \tilde{G}_i u_i - J_{\mathbf{g},\mathbf{n}} = \mathbb{P}_{V_{\varphi-1}}(f). \quad (23)$$

Finally, in the last step of the LDG construction, penalty stabilisation terms are included so as to ensure the well-posedness of the discrete problem. Solely targeting the velocity field, these terms weakly enforce Dirichlet boundary conditions and interfacial jump conditions; they may also be used to weakly enforce continuity of the velocity field between neighbouring elements of the same phase. Let $\tau_{\mathbf{u},\partial}$, $\tau_{\mathbf{u},ij}$, and $\tau_{\mathbf{u},\circ}$ denote the corresponding penalty parameters for the boundary, interphase, and intraphase faces, respectively. Let $E_{\mathbf{u},\mathbf{g}} : V_{\varphi}^d \rightarrow V_{\varphi}^d$ be the affine operator such that

$$\int_{\Omega} E_{\mathbf{u},\mathbf{g}}(\mathbf{u}) \cdot \omega = \int_{\Gamma_D} \tau_{\mathbf{u},\partial} (\mathbf{u}^- - \mathbf{g}_{\partial}) \cdot \omega^- + \sum_{i < j} \int_{\Gamma_{ij}} \tau_{\mathbf{u},ij} (\llbracket \mathbf{u} \rrbracket - \mathbf{g}_{ij}) \cdot \llbracket \omega \rrbracket + \int_{\Gamma_{\circ}} \tau_{\mathbf{u},\circ} \llbracket \mathbf{u} \rrbracket \cdot \llbracket \omega \rrbracket$$

holds for every test function $\omega \in V_{\varphi}^d$. (Here, $\Gamma_{\circ}$ denotes the union of all intraphase faces.) Subject to a suitable specification of penalty parameter values, $E_{\mathbf{u},\mathbf{g}}$ is added to the left-hand side of Eq. (21). Multiplying the resulting equations by M , multiplying Eq. (23) by $\tilde{M}$, and then rearranging the resulting system of equations so that the entire influence of the source data is moved to the right-hand side, we then obtain the governing equations Eq. (4) as described in the main article.

A.1. Operator definitions

For completeness, we define here the various operators and source terms omitted from the above derivation.

- The mass matrix M relative to the full-degree space V_φ is defined such that $u^\top M v = \int_\Omega u v$ for every $u, v \in V_\varphi$; similarly for the mass matrix $\tilde{M}$ relative to the space $V_{\varphi-1}$. Meanwhile, the μ -weighted mass matrix M_μ is defined such that $u^\top M_\mu v = \sum_i \int_{\Omega_i} u \mu_i v$, for all $u, v \in V_\varphi$.
- The discrete gradient operator $G : V_\varphi \rightarrow V_\varphi^d$ is defined by $G := \hat{G} + L$, where the broken gradient operator $\hat{G} : V_\varphi \rightarrow V_\varphi^d$ and lifting operator $L : V_\varphi \rightarrow V_\varphi^d$ are defined such that

$$\int_\Omega (\hat{G}u) \cdot \omega = \sum_{E \in \mathcal{E}} \int_E (\nabla u) \cdot \omega, \quad \text{and}$$

$$\int_\Omega (Lu) \cdot \omega = \int_{\Gamma_D} (u^+ - u^-) \omega^+ \cdot \mathbf{n} - \int_{\Gamma_D} u^- \omega^- \cdot \mathbf{n} + \sum_{i < j} \int_{\Gamma_{ij}} (1 - \lambda) (u^+ - u^-) \omega^- \cdot \mathbf{n} + \lambda (u^+ - u^-) \omega^+ \cdot \mathbf{n}$$

hold for every $u \in V_\varphi$ and $\omega \in V_\varphi^d$.

- The mixed-degree discrete gradient operator $\tilde{G} : V_\varphi \rightarrow V_{\varphi-1}^d$ is defined as the L^2 projection of G onto $V_{\varphi-1}^d$, i.e., so that

$$\int_\Omega (\tilde{G}u) \cdot \omega = \int_\Omega (Gu) \cdot \omega$$

holds for every $u \in V_\varphi$ and $\omega \in V_{\varphi-1}^d$. Written differently, note that $u^\top \tilde{G}_i^\top \tilde{M} \omega = u^\top G_i^\top M \mathbb{P}_{V_\varphi}(\omega)$ for each component i and every $u \in V_\varphi$ and $\omega \in V_{\varphi-1}^d$. The latter form shows that the pressure gradient term in Eq. (21) can also be rewritten as $M^{-1} \tilde{G}_i^\top \tilde{M} p$.

- The lifted Dirichlet-like source data $J_g \in V_{\varphi}^{d \times d}$ is defined such that

$$\int_\Omega J_g : \omega = \int_{\Gamma_D} \mathbf{g}_\partial \cdot \omega^- \cdot \mathbf{n} + \sum_{i < j} \int_{\Gamma_{ij}} (1 - \lambda) \mathbf{g}_{ij} \cdot \omega^- \cdot \mathbf{n} + \lambda \mathbf{g}_{ij} \cdot \omega^+ \cdot \mathbf{n}$$

holds for every $\omega \in V_{\varphi}^{d \times d}$, while $J_{g-n} \in V_{\varphi-1}$ is defined such that

$$\int_\Omega J_{g-n} \omega = \int_{\Gamma_D} \omega^- \mathbf{g}_\partial \cdot \mathbf{n} + \sum_{i < j} \int_{\Gamma_{ij}} ((1 - \lambda) \omega^- + \lambda \omega^+) \mathbf{g}_{ij} \cdot \mathbf{n}$$

holds for every $\omega \in V_{\varphi-1}$. Meanwhile, the lifted Neumann-like source data $J_h \in V_{\varphi}^d$ is defined such that

$$\int_\Omega J_h \cdot \omega = \int_{\Gamma_N} \mathbf{h}_\partial \cdot \omega^- + \sum_{i < j} \int_{\Gamma_{ij}} \lambda \mathbf{h}_{ij} \cdot \omega^- + (1 - \lambda) \mathbf{h}_{ij} \cdot \omega^+$$

holds for every $\omega \in V_{\varphi}^d$.

A.2. Viscosity-upwinded fluxes

For multiphase Stokes problems, we apply viscosity-upwinded numerical fluxes, developed for elliptic interface problems in [9] and extended to multiphase Stokes problems in [1]. Roughly speaking, in the limit of arbitrarily-large viscosity ratios, this biasing corresponds to the multiphase Stokes problem separating into two distinct problems: one for the highly viscous phase (which effectively “sees” a stress-like boundary condition at the interface), and one for the other phase (which effectively “sees” a Dirichlet-like boundary condition). As explored in [9], biasing the numerical fluxes in the same way is a key step in achieving efficient multigrid solvers as well as high-order accuracy. The viscosity-upwinded strategy corresponds to defining λ in the numerical flux functions $\hat{\mathbf{u}}$ and $\hat{\sigma}$ in Eqs. (16) and (19) as follows:

$$\lambda = \begin{cases} 0 & \text{if } \mu^- < \mu^+, \\ 0.5 & \text{if } \mu^- = \mu^+, \\ 1 & \text{if } \mu^- > \mu^+. \end{cases} \quad (24)$$

Accordingly, $\hat{\mathbf{u}}$ is biased to the more viscous phase and $\hat{\sigma}$ to the less viscous phase, with interfacial jump data $\mathbf{g}_{ij}$ and $\mathbf{h}_{ij}$ incorporated appropriately.

A.3. Well-posedness

Assuming a suitable specification of penalty parameters (see below), the mixed-degree LDG framework is observed to be well-posed, inf-sup stable, and optimal-order accurate. The formulation may be viewed as a generalisation of the LDG schemes developed by Cockburn et al. [7], for which optimal-order accuracy and inf-sup stability have been proven for single-phase, constant-coefficient model problems; see also [17,18]. Although we have not formally proven these properties for the generalised setting, extensive numerical experiments demonstrate that they hold, including for smoothly varying viscosity coefficients as well as high-contrast multiphase problems. We leave a formal proof for future work.

A.4. Penalty stabilisation parameters

In general, strictly positive penalty parameters $\tau_{u,\partial}$, $\tau_{u,ij}$, and $\tau_{u,o}$ are sufficient to ensure the well-posedness of the final discretised Stokes problem, e.g., to ensure it has the expected kernel of the continuum Stokes operator. However, strict positivity is not a necessary condition; e.g., on regular Cartesian grids, without any embedded interfaces, it is possible to entirely disable intraphase penalisation and still maintain stability. Further discussion on particular parameter choices and appropriate scaling with respect to viscosity, mesh size, polynomial degree, etc., are given in [1]; here, we specify the values used in the numerical experiments of this work:

- On Dirichlet boundary faces, $\tau_{u,\partial} = C_\varphi \mu^- h^{-1}$, where C_φ is a constant depending on the polynomial degree, μ^- is the local viscosity, and h the local element size. For the numerical experiments in this work, $C_1 = 1$ while $C_2 = C_3 = 16$; the case of $\varphi = 1$ benefits from somewhat weaker Dirichlet boundary penalisation.
- The AMR test problems require some amount of intraphase stabilisation, where it is set to $\tau_{u,o} = C_\varphi h^{-1}$. In all other test problems, intraphase penalisation is unnecessary and is disabled, $\tau_{u,o} = 0$.
- For the multiphase test problems, the interphase penalty parameter is set to $\tau_{u,ij} = 16 \min(\mu^-, \mu^+) h^{-1}$, where $\mu^\pm$ is the local viscosity of the two elements on either side of the interfacial mesh face. We note that it is important to choose the minimum of the two viscosities, as discussed further in [9].

These values have been chosen based on the dual goals of obtaining high-order accuracy as well as good multigrid performance, and follow typical values employed in previous work [1,2,9].

Appendix B. Optimising smoother parameters

We describe here an algorithm to optimise for the multigrid smoother parameters $(\zeta_u, \omega_u, \omega_p)$. The approach has two main components: (i) a search algorithm that adaptively explores the parameter space; and (ii) a method to evaluate the corresponding multigrid convergence rate ρ via two-grid local Fourier analysis.

Algorithm 3 Exploring the $(\zeta_u, \omega_u, \omega_p)$ parameter space.

```

1: Define initial seed point,  $\mathcal{Z} := \{(1, 1, 1)\}$ .
2: Define initial step size,  $\delta := (0.25, 0.25, 0.25)$ .
3: Let  $\varsigma$  track the minimum observed value of  $\rho$  so that at any instant,  $\varsigma \equiv \min_{z \in \mathcal{Z}} \rho(z)$ .
4: while exploring do
5:    $\rho_\star := \begin{cases} \varsigma^\alpha & \text{if } \varsigma < 1, \\ +\infty & \text{otherwise.} \end{cases}$ 
6:    $\mathcal{Z}_\star := \{z \in \mathcal{Z} \mid \rho(z) \leq \rho_\star\}$ .
7:    $n_i := \frac{1}{\delta_i} \left( \max_{z \in \mathcal{Z}_\star} z_i - \min_{z \in \mathcal{Z}_\star} z_i \right), \forall i \in \{1, 2, 3\}$ .
8:   if  $\min_i n_i \geq n_{\text{target}}$  then
9:     End search and return  $\mathcal{Z}$ .
10:  Refine step size:  $\delta_j := \frac{1}{2} \delta_j$ , where  $j = \operatorname{argmin}_i n_i$ .
11:  Mark every point in  $\mathcal{Z}$  as unvisited.
12:  while  $\mathcal{Z}$  has unvisited points do
13:     $z := \operatorname{argmin}_{\text{unvisited } z \in \mathcal{Z}} \rho(z)$ .
14:    Mark  $z$  as visited.
15:    if  $\varsigma < 1$  and  $\rho(z) > \varsigma^\alpha$  then
16:      break
17:    for  $y \in \prod_{i=1,2,3} \{z_i - \delta_i, z_i, z_i + \delta_i\}$  do
18:      if  $y \notin \mathcal{Z}$  then
19:        Insert  $y$  into  $\mathcal{Z}$  (marked unvisited).
```

Suppose for the moment that we have implemented a function $\rho : \mathbb{R}^3 \rightarrow \mathbb{R}$ that evaluates the multigrid convergence rate $\rho(z)$ for the parameters $z = (\zeta_u, \omega_u, \omega_p)$. Treating ρ as a black-box function, Algorithm 3 outlines the search algorithm that iteratively and adaptively evaluates ρ on a point cloud in parameter space. Given an (initial or interim) point cloud $\mathcal{Z}$ arising from a step size δ , the method selects from $\mathcal{Z}$ the set of near-optimal points, $\mathcal{Z}_\star$, i.e., those for which the corresponding convergence rate ρ is close to the minimum thus far. Next, the bounding box of $\mathcal{Z}_\star$ is computed so as to determine the effective sampling resolution relative to the step size δ . If the sampling resolution across all three parameters is sufficiently high, the search process terminates. Otherwise, the coordinate axis with the lowest resolution is chosen to undergo refinement, so that δ is halved in that direction. Then, for each point in $\mathcal{Z}$ —visited in order of increasing ρ value—a patch of neighbouring points is constructed via a $3 \times 3 \times 3$ Cartesian grid of cell size δ , and these neighbouring points are added to the search space. The sampling procedure prioritises areas where ρ is

Algorithm 4 Evaluation of $\rho(\zeta_u, \omega_u, \omega_p)$.

```

1: Let  $\Theta_n$  denote the set  $\{\frac{2\pi}{n}(i - \frac{1}{2})\}_{i=1}^n$ .
2:  $\rho_{\text{coarse}} := \max_{\theta \in (\Theta_6)^d} \rho(\theta; \zeta_u, \omega_u, \omega_p)$ .
3: if  $\rho_{\text{coarse}} > 1$  then
4:   return  $\rho_{\text{coarse}}$ .
5: if  $\varsigma < 1$  and  $\rho_{\text{coarse}} > \varsigma^\alpha$  then
6:   return  $\rho_{\text{coarse}}$ .
7:  $\rho_{\text{fine}} := \max_{\theta \in (\Theta_{12})^d} \rho(\theta; \zeta_u, \omega_u, \omega_p)$ .
8: return  $\max(\rho_{\text{coarse}}, \rho_{\text{fine}})$ .

```

expected to be the smallest; if the creation of a neighbouring point leads to the reduction of $\min_{z \in \mathcal{Z}} \rho(z)$, that point will be visited next. Conversely, if upon visiting an existing point whose ρ value is now considered too large, all remaining unvisited points have the same property, so the inner loop (line 12) search can terminate. The outer loop (line 4) progressively applies finer resolutions; once the target resolution is reached, the overall search terminates, with the result being a point cloud $\mathcal{Z}$ for which the majority of points focus in on the region where ρ is close to the minimum-observed ρ .

To specify a few more details in this search process, a sample z is considered close to optimal if $\rho(z) \leq (\min_{z \in \mathcal{Z}} \rho(z))^\alpha$, where $\alpha < 1$ is a user-chosen parameter; for the results in this work, we have used $\alpha = 1/1.1$, corresponding to the statement that the smoother parameters at z yield at most 10% more multigrid iterations (for a fixed residual reduction threshold) compared to the best found so far. The sampling process is initialised at the point $(\zeta_u, \omega_u, \omega_p) = (1, 1, 1)$ with an initial step size of 0.25. (Sometimes the initial seed yields a ρ larger than unity; when this occurs, the near-optimal threshold $\rho_\star$ is set to $+\infty$ (line 5), thereby switching to an “explore everywhere” heuristic.) As the search proceeds, it may be the case that the typical range of ζ_u , say, ends up being larger than that of ω_u , say; the convergence criterion roughly requires that the bounding box of the set of near-optimal points has at least n_{target} points per dimension; we have used $n_{\text{target}} = 16$, sufficient to produce the figures of Figs. 2 and 3. Our implementation of Algorithm 3 memoises/caches the results of the (expensive) black-box evaluation of ρ in a simple dictionary/map data structure; meanwhile, the inner loop over unvisited points ordered by ρ can be efficiently implemented via a simple priority queue.

We now turn to the evaluation of $\rho(z)$ for a given $z = (\zeta_u, \omega_u, \omega_p)$. We define ρ via the results of two-grid local Fourier analysis (LFA) [14–16,19]. In essence, this procedure analyses the performance of a two-level multigrid method as applied to an infinite uniform Cartesian grid spanning all of $\mathbb{R}^d$, together with a function space spanned by functions of the form $\exp(i\theta \cdot \mathbf{x}/h)$, where $i^2 = -1$. LFA analyses often proceed by recognising that a constant coefficient PDE operator is diagonalised by the Fourier modes $\exp(i\theta \cdot \mathbf{x}/h)$, thereby yielding “symbols” of the PDE operator A as well as symbols of the multigrid pre-smoothing, post-smoothing, restriction, interpolation, and coarse grid operators. These symbols combine to form a symbol $E(\theta)$ for the error propagation operator for a two-level method—in the present setting, $E(\theta)$ is a complex-valued square matrix of size proportional to the #dof on each element—from which the convergence rate, for a specific frequency θ , is defined as the spectral radius of $E(\theta)$. The maximum spectral radius over an appropriate frequency space is then used to define the LFA-predicted convergence rate ρ for the overall multigrid method. For details, see, e.g., [14–16].

In this work, instead of symbolically building and manipulating the various symbols, we found it simpler to mimic two-grid LFA by invoking the multigrid DG implementation on a virtual infinite-extent Cartesian grid of cell size h . Given a frequency $\theta \in [-\pi, \pi]^d$, we define a virtual grid function $x \mapsto \sigma \exp(i\theta \cdot \mathbf{x}/h)$, where $\mathbf{x}$ denotes the lower-left coordinates of each grid cell. Here, σ denotes a single set of coefficients of size $d(\mathcal{G} + 1)^d + \mathcal{G}^d$ (one per velocity and pressure modal coefficient, respectively); the coefficients σ are the same for every grid cell, but otherwise remain undefined. For a given set of smoother parameters $(\zeta_u, \omega_u, \omega_p)$, the multigrid smoother is computed following the methods of Section 3. Then, a two-level multigrid method is applied to solve $A_h x_h = b_h$, as follows (for simplicity of presentation we consider the 2D case, the 3D case being an obvious extension):

- (i) Define an initial grid function x_h on the virtual infinite-extent fine grid, $x_h^{i,j} = \sigma \exp(i\theta \cdot (i, j))$ on grid cell $(i, j) \in \mathbb{Z}^d$. Define a right-hand side of $b_h \equiv 0$.
- (ii) Apply the presmoothers v_1 times, using red-black ordering. The result of smoothing is a new grid function such that $x_h^{i,j} = (X_{i,j} \sigma) \exp(i\theta \cdot (i, j))$, where $X_{i,j}$ is cell-dependent complex-valued matrix. Crucially, owing to the nature of red-black smoothing, the new grid function repeats every 2×2 patch of grid cells, in the sense that $X_{i,j} = X_{i \bmod 2, j \bmod 2}$ for every $(i, j) \in \mathbb{Z}^2$.
- (iii) Compute the residual on the fine grid, $r_h = b_h - A_h x_h$; the result is another grid function which repeats every 2×2 grid cells.
- (iv) Restrict the residual to the virtual infinite-extent coarse grid; crucially, the restriction process agglomerates every patch of 2×2 cells into a single element, which means the restricted residual b_{2h} is the same on every coarse grid cell, in the sense that $b_{2h}^{i,j} = (B\sigma) \exp(i\theta \cdot (i, j))$ where B is a cell-independent complex-valued matrix and $(i, j) \in (2\mathbb{Z})^2$.
- (v) Solve the coarse-grid problem, exactly: using the symbol of A_{2h} , compute x_{2h} such that $A_{2h} x_{2h} = b_{2h}$. The result is a grid function such that $x_{2h}^{i,j} = (C\sigma) \exp(i\theta \cdot (i, j))$ for every coarse grid cell $(i, j) \in (2\mathbb{Z})^2$; in fact, $C = A_{2h}(\theta)^{-1} B$, where $A_{2h}(\theta)$ is the symbol of the coarse grid operator and $A_{2h}(\theta)^{-1}$ can be computed, e.g., via complex-valued SVD.

Table 2

Multigrid solver iteration counts for the 2D model problem considered in [Appendix C](#); blank entries denote failure to converge within 50 iterations.

Multigrid smoother	Damping	Grid size n							
		4	8	16	32	64	128	256	512
Block Jacobi	none	–	–	–	–	–	–	–	–
	$\omega = 0.55$	13	16	18	21	24	32	40	47
Block Gauss–Seidel (forward/forward)	none	–	–	–	–	–	–	–	–
	$\omega = 0.7$	7	8	9	10	10	13	16	21
Block Gauss–Seidel (forward/reverse)	none	50	–	–	–	–	–	–	–
	$\omega = 0.7$	8	8	10	10	12	13	15	17
Least-squares balanced block-diagonal Q (this work)	n/a	5	5	5	5	6	6	6	6

- (vi) Interpolate the coarse-grid solution and apply it as a correction to the fine grid function x_h . Then apply the postsmoother v_2 times. The updated fine grid function is again of the form $x_h^{i,j} = (X_{i,j}\sigma)\exp(i\theta \cdot (i, j))$, where $X_{i,j}$ is a cell-dependent complex-valued matrix; as before, these matrices repeat: $X_{i,j} = X_{i \bmod 2, j \bmod 2}$ for every $(i, j) \in \mathbb{Z}^2$, so only four such matrices need computing.
- (vii) At this point, we have in fact computed the two-grid error propagator of the two-level multigrid method, represented as a linear function applied to the unknown coefficients σ . Finally, we determine the slowest convergence rate over all possible coefficients, i.e., compute the spectral radius of the error propagator. Using the matrices of the previous step, we therefore define the two-grid multigrid convergence rate, for the input frequency θ and smoother parameters $(\zeta_u, \omega_u, \omega_p)$, via

$$\rho(\theta; \zeta_u, \omega_u, \omega_p) := \max_{(i,j) \in \{0,1\}^2} \rho(X_{i,j}).$$

The above method evaluates the two-grid convergence factor ρ for a concrete, numerically-defined frequency θ . Its implementation is nearly identical to the regular multigrid method, except for three main aspects: (i) on each element, state variables are complex-valued matrices instead of the usual real-valued coefficient vector; (ii) stencil operations appropriately account for the grid function complex-exponential factors, e.g., the $\begin{bmatrix} -1 & 2 & -1 \end{bmatrix}$ stencil of a 1D finite difference Laplacian is replaced by $\begin{bmatrix} -e^{-i\theta} & 2 & -e^{i\theta} \end{bmatrix}$; and (iii) instead of applying the multigrid algorithm to every cell, because of the repeating nature of the computations, we need only apply it to a 2×2 primary patch of cells. Finally, we find the slowest convergence rate across all possible frequencies to define the final two-grid LFA convergence factor¹⁵:

$$\rho(\zeta_u, \omega_u, \omega_p) := \sup_{\theta \in [-\pi, \pi]^d} \rho(\theta; \zeta_u, \omega_u, \omega_p). \quad (25)$$

In this work, we have used a simple maximisation approach, outlined in Algorithm 4, that works in tandem with the parameter search of Algorithm 3, as follows. As a preliminary coarse approximation, the above two-grid LFA method is used to compute $\rho(\theta; \zeta_u, \omega_u, \omega_p)$ on a 6×6 ($\times 6$) grid in frequency space. If the maximum-found ρ is too large compared to the best smoother parameters found so far (located somewhere else), then it is in some sense futile to perform a more accurate calculation. If, on the other hand, the coarse approximation results in a sufficiently small ρ , then we apply a higher-resolution evaluation on a 12×12 ($\times 12$) grid in frequency space. The net effect is the evaluation of Eq. (25) on semi-uniform grid in frequency space having $6+12$ points per dimension; in this work it was found that the coarse resolution of 6 does well to quickly rule out bad smoother parameters, while the fine resolution of $6+12$ yields a predicted convergence factor ρ sufficiently close to the true maximum, at least for the overall purpose of predicting a good set of smoother parameters as discussed in Section 3.5.

Appendix C. A brief comparison with classical smoothers

We consider here an illustrative two-dimensional model problem to compare multigrid performance across a variety of classical smoothers. The model problem consists of a single-phase, constant-viscosity, steady-state, standard-form Stokes problem, with periodic boundary conditions, on an $n \times n$ uniform Cartesian grid, and $\varphi = 2$, i.e., biquadratic velocity field and bilinear pressure field. The solver consists of a multigrid V-cycle left-preconditioned GMRES method, with $v = 3$ pre- and post-smoothing steps. We consider three kinds of classical smoothers: (i) block Jacobi; (ii) multicoloured block Gauss–Seidel using identical pre- and post-smoothing colour sequences (forward/forward); and (iii) multicoloured block Gauss–Seidel using a reversed post-smoothing colour sequence (forward/reverse). For each of these, we consider undamped and optimally damped versions.¹⁶ Table 2 reports the number of outer GMRES iterations required to achieve a residual reduction factor of 10^8 (using a zero initial guess and a randomly generated right-hand side); blank entries denote failure to converge within 50 iterations. Overall, block Jacobi is ineffective, while

¹⁵ In typical LFA approaches, it is common to consider blocked forms of the error propagation symbol, blocked according to a primary low-frequency space $[-\frac{\pi}{2}, \frac{\pi}{2}]^d$, together with multiple π -shifted copies thereof, and then maximise over the low-frequency space [16]. For the alternative approach presented in this paper, it is appropriate to maximise over the entire frequency space.

¹⁶ A simple parameter sweep was used to select an approximately optimal damping factor ω that maximises the convergence rate.

multicoloured block Gauss–Seidel degrades as the mesh is refined. In contrast, the Q smoother developed in this work maintains fast and nearly grid-independent convergence. Other model problem configurations, such as three-dimensional cases, yield similar findings.

Data availability

No data was used for the research described in the article.

References

- [1] Saye RI. Fast multigrid solution of high-order accurate multiphase Stokes problems. *Commun Appl Math Comput Sci* 2020;15(2):147–96. <http://dx.doi.org/10.2140/camcos.2020.15.33>.
- [2] Saye RI. A comparative study of efficient multigrid solvers for high-order local discontinuous Galerkin methods: Poisson, elliptic interface, and multiphase Stokes problems. *Commun Appl Math Comput Sci* 2025;20(1):253–91. <http://dx.doi.org/10.2140/camcos.2025.20.253>.
- [3] Vanka SP. Block-implicit multigrid solution of Navier-Stokes equations in primitive variables. *J Comput Phys* 1986;65(1):138–58. [http://dx.doi.org/10.1016/0021-9991\(86\)90008-2](http://dx.doi.org/10.1016/0021-9991(86)90008-2).
- [4] Vanka SP. A calculation procedure for three-dimensional steady recirculating flows using multigrid methods. *Comput Methods Appl Mech Engrg* 1986;55(3):321–38. [http://dx.doi.org/10.1016/0045-7825\(86\)90058-7](http://dx.doi.org/10.1016/0045-7825(86)90058-7).
- [5] Farrell PE, He Y, MacLachlan SP. A local Fourier analysis of additive Vanka relaxation for the Stokes equations. *Numer Linear Algebra Appl* 2021;28(3):e2306. <http://dx.doi.org/10.1002/nla.2306>.
- [6] Brandt A, Livne OE. *Multigrid Techniques*. Society for Industrial and Applied Mathematics; 2011. <http://dx.doi.org/10.1137/1.9781611970753>.
- [7] Cockburn B, Kanschat G, Schötzau D, Schwab C. Local discontinuous Galerkin methods for the Stokes system. *SIAM J Numer Anal* 2002;40(1):319–43. <http://dx.doi.org/10.1137/S0036142900380121>.
- [8] Fortunato D, Rycroft CH, Saye RI. Efficient operator-coarsening multigrid schemes for local discontinuous Galerkin methods. *SIAM J Sci Comput* 2019;41(6):A3913–37. <http://dx.doi.org/10.1137/18M1206357>.
- [9] Saye RI. Efficient multigrid solution of elliptic interface problems using viscosity-upwinded local discontinuous Galerkin methods. *Commun Appl Math Comput Sci* 2019;14(2):247–83. <http://dx.doi.org/10.2140/camcos.2019.14.247>.
- [10] Demmel JW. *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics; 1997. <http://dx.doi.org/10.1137/1.9781611971446>.
- [11] Saad Y. *Iterative methods for sparse linear systems*. 2nd ed. Society for Industrial and Applied Mathematics; 2003. <http://dx.doi.org/10.1137/1.9780898718003>.
- [12] Chorin AJ. Numerical solution of the Navier-Stokes equations. *Math Comp* 1968;22(104):745–62. <http://dx.doi.org/10.1090/S0025-5718-1968-0242392-2>.
- [13] Lewis R. In: *A guide to graph colouring*, vol. 7, Springer; 2015. <http://dx.doi.org/10.1007/978-3-319-25730-3>.
- [14] Trottenberg U, Oosterlee C, Schüller A. *Multigrid*. Academic Press; 2001.
- [15] Wienands R, Joppich W. *Practical Fourier analysis for multigrid methods*. CRC Press; 2005.
- [16] Brown J, He Y, MacLachlan S, Menickelly M, Wild SM. Tuning multigrid methods with robust optimization and local Fourier analysis. *SIAM J Sci Comput* 2021;43(1):A109–38. <http://dx.doi.org/10.1137/19M1308669>.
- [17] Cockburn B, Kanschat G, Schötzau D. LDG methods for Stokes flow problems. In: *Numerical mathematics and advanced applications*. Springer; 2003, p. 755–64. http://dx.doi.org/10.1007/978-88-470-2089-4_68.
- [18] Cockburn B, Kanschat G, Schötzau D. A locally conservative LDG method for the incompressible Navier-Stokes equations. *Math Comp* 2005;74(251):1067–95. <http://dx.doi.org/10.1090/S0025-5718-04-01718-1>.
- [19] Brandt A. Multi-level adaptive solutions to boundary-value problems. *Math Comp* 1977;31:333–90.