Adaptive geometric multigrid for the mixed finite cell formulation of Stokes and Navier–Stokes equations

S. Saberi1 | G. Meschke2 | A. Vogel1

1High Performance Computing, Ruhr University Bochum, Bochum, Germany
2Institute for Structural Mechanics, Ruhr University Bochum, Bochum, Germany

Correspondence
S. Saberi, High Performance Computing, Ruhr University Bochum, Universitätsstr. 150, 44801 Bochum, Germany.
Email: seyed.saberi@rub.de

Funding information
Deutsche Forschungsgemeinschaft; Interaction Modeling in Mechanized Tunneling, Grant/Award Number: 77309832

Abstract
The unfitted finite element methods have emerged as a popular alternative to classical finite element methods for the solution of partial differential equations and allow modeling arbitrary geometries without the need for a boundary-conforming mesh. On the other hand, the efficient solution of the resultant system is a challenging task because of the numerical ill-conditioning that typically entails from the formulation of such methods. We use an adaptive geometric multigrid solver for the solution of the mixed finite cell formulation of saddle-point problems and investigate its convergence in the context of the Stokes and Navier–Stokes equations. We present two smoothers for the treatment of cut cells in the finite cell method and analyze their effectiveness for the model problems using a numerical benchmark. Results indicate that the presented multigrid method is capable of solving the model problems independently of the problem size and is robust with respect to the depth of the grid hierarchy.

Keywords
domain decomposition, finite cell method, geometric multigrid, saddle-point problems, unfitted finite element method

1 INTRODUCTION

The finite element method is a powerful tool for the numerical approximation of partial differential equations and has been successfully applied to a wide range of problems in the past decades. The generation of an appropriate tessellation of the computational domain is often a daunting task in boundary-conforming finite element methods and has motivated the development of a number of techniques, such as the extended finite element method (XFEM),1 cutFEM2 and the finite cell method3-5 under the umbrella of unfitted finite element methods. The mutual goal of such methods is to separate the physical domain from the computational domain so that an arbitrary geometry can be resolved by a trivially constructed computational mesh, circumventing the need for the generation of a boundary-conforming tessellation of the domain. The finite cell method is a fictitious domain method and employs an integration technique such as Gaussian over-integration and uniform or adaptive subcell integration for the resolution of the physical domain and a method for the weak imposition of essential boundary conditions such as Lagrange multipliers,6-9 the penalty method10-12 and the Nitsche’s method.13-17 The finite cell method has been applied to different classes of problems, including structural mechanics3,4 and fluid dynamics.18 A review of the finite cell method was published in the context of structural mechanics.5

This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

© 2023 The Authors. International Journal for Numerical Methods in Fluids published by John Wiley & Sons Ltd.

Int J Numer Meth Fluids. 2023;95:1035–1053. wileyonlinelibrary.com/journal/fld 1035
A wide variety of problems in fluid dynamics, structural mechanics, finance and so forth, are mathematically described as saddle-point problems. Several methods, including Uzawa methods, multigrid methods and Krylov subspace solvers are typically employed for the iterative solution of saddle-point systems. Effective preconditioning is essential for Krylov subspace solvers to achieve scalable convergence. Multigrid methods are often used as preconditioners in Krylov subspace solvers. Other preconditioners include Schur complement methods, such as pressure correction schemes, approximate commutator schemes and domain decomposition methods, such as overlapping Schwarz methods. Geometric multigrid methods are among the most efficient algorithms for the solution of saddle-point problems. A number of multigrid methods have been proposed in the context of flow problems, including a multigrid method based on incomplete LU factorization, pressure correction methods used as smoothers in the multigrid method, the Braess-Sarazin smoother, which is based on pressure correction schemes and the Vanka smoother, which is based on the solution of local saddle-point problems.

The iterative solution of saddle-point problems is a challenging task which is further complicated by the fact that the weak formulation of the finite cell method typically leads to matrices that are ill-conditioned. The conditioning of the system matrix depends on how the physical domain intersects the computational mesh and deteriorates in the presence of small cut fractions. A conjugate gradient method for the finite cell formulation of the elasticity problem with an AMG preconditioner and an additive Schwarz preconditioner were studied. A preconditioner for the conjugate gradient and GMRES methods was studied for the solution of a variety of finite cell problems. Geometric multigrid with Schwarz-type smoothers was recently studied for the finite cell formulation of elliptic problems.

We use the mixed finite cell formulation of the incompressible Stokes and Navier–Stokes equations with Nitsche’s boundary conditions. We employ space trees for the spatial discretization of the computational domain and use adaptive integration in addition to adaptive mesh refinement. The contributions of this work can be summarized as follows:

- We formulate an adaptive geometric multigrid solver for the solution of the mixed finite cell formulation of saddle-point problems in the context of the Stokes and Navier–Stokes equations and present two smoothers for the treatment of the resultant systems.
- We investigate the convergence of the geometric multigrid method in different scenarios both as a solver and as a preconditioner in Krylov subspace solvers using numerical benchmarks for the model problems.
- We discuss our observations surrounding the performance and scalability of the solver with regards to factors such as the choice of the smoother operator, the stabilization method, the number of smoothing steps and so forth.

The remainder of this article is organized as follows. The mixed finite cell formulation of the incompressible Stokes and Navier–Stokes equations is derived in Section 2. The necessary components of the geometric multigrid solver, including the generation of the grid hierarchy and smoothers are discussed in Section 3. The performance of the developed multigrid method is analyzed in Section 4. Finally, some concluding remarks are given in Section 5.

## 2 | MIXED FINITE CELL FORMULATION OF THE MODEL PROBLEMS

Given a physical domain $\Omega$ with boundary $\partial\Omega$, we consider a regular embedding domain $\Omega_e$ such that $\Omega \subset \Omega_e$, see Figure 1A. We start by deriving the mixed finite cell formulation of the incompressible Stokes equations. The strong form of the Stokes equations can be written as

\[
-\eta \nabla^2 \mathbf{u} + \nabla p = \mathbf{f} \quad \text{in} \ \Omega, \\
\nabla \cdot \mathbf{u} = 0 \quad \text{in} \ \Omega,
\]

where $\mathbf{u}$ is the fluid velocity vector, $p$ is the fluid pressure, $\mathbf{f}$ is the body force exerted on the fluid and $\eta$ is the kinematic viscosity.

We denote by $(\cdot, \cdot)_\Omega$ and $(\cdot, \cdot)_{\partial \Omega}$ the $L^2$ scalar product on $\Omega$ and $\partial \Omega$, respectively. The weak form of the Stokes equations is obtained as follows by multiplying Equation (1) by test functions from the left, integrating over the computational domain and transferring the derivatives using integration by parts where appropriate:

\[
(\eta \nabla v, \nabla \mathbf{u})_\Omega - (\mathbf{v}, \eta \nabla \mathbf{u})_{\partial \Omega} - (\nabla \cdot \mathbf{v}, p)_\Omega + (\mathbf{v}, n p)_{\partial \Omega} - (q, \nabla \cdot \mathbf{u})_\Omega = (\mathbf{v}, \mathbf{f})_\Omega.
\]

(2)
where \((v, q)\) are the infinite-dimensional vector-valued velocity and scalar pressure test functions, respectively. The Stokes boundary value problem is formed by requiring the solution to Equation (1) to satisfy the following boundary conditions

\[
\begin{align*}
\mathbf{u} &= \mathbf{w} \text{ on } \Gamma_D \subset \partial \Omega, \\
\eta \frac{\partial \mathbf{u}}{\partial n} - np &= h \text{ on } \Gamma_N := \partial \Omega \setminus \Gamma_D, 
\end{align*}
\]

where \(\partial \Omega\) is the boundary of the computational domain, \(\Gamma_D\) and \(\Gamma_N\) denote the Dirichlet and Neumann parts of the boundary, respectively, such that \(\partial \Omega = \Gamma_D \cup \Gamma_N\) and \(\Gamma_D \cap \Gamma_N = \emptyset\) and \(n\) is the outward unit normal vector to the boundary with unit length.

The spaces of test and trial functions in the weak form of the Stokes equations in boundary-conforming finite element methods are chosen as

\[
\begin{align*}
V_w &:= \{ \mathbf{u} \in H^1(\Omega)^d \mid \mathbf{u} = \mathbf{w} \text{ on } \Gamma_D \}, \\
V_0 &:= \{ v \in H^1(\Omega)^d \mid v = 0 \text{ on } \Gamma_D \}, \\
Q &:= \{ q \in L^2(\Omega) \}.
\end{align*}
\]

(4)

where \(d\) is the dimension of the domain. The weak form in Equation (2) can be written as follows by combining the boundary terms and separating the boundary into its Dirichlet and Neumann parts:

Find \((\mathbf{u}, p) \in (V_w, Q)\) such that

\[
(\eta \nabla \mathbf{v}, \nabla \mathbf{u})_{\Omega_2} - (\mathbf{v} \cdot \mathbf{n}, p)_{\Omega_2} - (q, \nabla \cdot \mathbf{u})_{\Omega_2} = (\mathbf{v}, f)_{\Omega_2} + (\mathbf{v}, h)_{\Gamma_N}, \quad \forall (\mathbf{v}, q) \in (V_0, Q).
\]

(5)

The imposition of essential boundary conditions in the form above is referred to as the strong imposition of boundary conditions. In unfitted finite element methods, however, the strong imposition of essential boundary conditions is in general not possible as the physical domain does not necessarily coincide with the computational domain. Therefore, essential boundary conditions are imposed weakly using methods such as Lagrange multipliers,\(^6-9\) the penalty method\(^10-12\) and Nitsche’s method.\(^13-17\) The trial and test functions are now chosen from the same spaces \((V, Q)\), given by

\[
\begin{align*}
V &:= \{ \mathbf{u} \in H^1(\Omega)^d \}, \\
Q &:= \{ q \in L^2(\Omega) \}.
\end{align*}
\]

(6)
Therefore, the boundary terms in Equation (2) do not vanish over the Dirichlet boundary. We employ Nitsche’s method for the imposition of essential boundary conditions in this work. The weak form of the Stokes equations with Nitsche’s boundary terms is written as

\[
(\eta \nabla v, \nabla u)_\Omega - (v, n \cdot \eta \nabla u)_{\Gamma_D} - (n \cdot \eta \nabla v, u)_{\Gamma_D} + (v, \lambda (u - w))_{\Gamma_D} \\
- (\nabla \cdot v, p)_\Omega + (v, np)_{\Gamma_D} + (nq, u)_{\Gamma_D} - (q, \nabla \cdot u)_\Omega = (v, f)_{\Omega} + (v, h)_{\Gamma_N}.
\]  

(7)

**Remark 1.** The third and seventh terms on the l.h.s of Equation (7) are the velocity and pressure symmetric consistency terms, respectively. These terms ensure that the modified weak form retains the symmetry of the original weak form. The fourth term on the l.h.s of Equation (7) is a stabilization term with a positive stabilization parameter $\lambda$ that ensures the stability of the solution for a large-enough $\lambda$ and that the solution satisfies the essential boundary conditions. These terms also retain the variational consistency in Nitsche’s method (see Remark 2).

The boundary terms in Equation (7) can be separated to form the following weak problem:

Find $(u, p) \in (V, Q)$ such that

\[
(\eta \nabla v, \nabla u)_\Omega - (v, n \cdot \eta \nabla u)_{\Gamma_D} - (n \cdot \eta \nabla v, u)_{\Gamma_D} + (v, \lambda u)_{\Gamma_D} \\
- (\nabla \cdot v, p)_\Omega + (v, np)_{\Gamma_D} + (nq, u)_{\Gamma_D} - (q, \nabla \cdot u)_\Omega \\
= (v, f)_{\Omega} + (v, h)_{\Gamma_N} - (n \cdot \eta \nabla v, w)_{\Gamma_D} + (v, \lambda w)_{\Gamma_D} + (nq, w)_{\Gamma_D},
\]

\forall (v, q) \in (V, Q).

(8)

**Remark 2.** The weak form in Equation (8) is consistent in the sense that it can be shown that the solution to the original problem in Equation (1) with the boundary conditions in Equation (3) also satisfies the weak form in Equation (8).

**Remark 3.** Compared to the Lagrange multiplier method and the penalty method, the Nitsche’s method has the advantage of keeping the problem size the same as the original problem, retaining the symmetry of the original weak form and being variationally consistent.

We now derive the finite cell weak form of the Stokes equations with Nitsche’s boundary conditions by extending the weak form in Equation (8) to an embedding domain as follows:

Find $(u, p) \in (V_e, Q_e)$ such that

\[
(\eta \nabla v, a \nabla u)_{\Omega_e} - (v, n \cdot \eta \nabla u)_{\Gamma_D} - (n \cdot \eta \nabla v, u)_{\Gamma_D} + (v, \lambda u)_{\Gamma_D} \\
- (\nabla \cdot v, ap)_{\Omega_e} + (v, np)_{\Gamma_D} + (nq, u)_{\Gamma_D} - (q, a \nabla \cdot u)_{\Omega_e} \\
= (v, af)_{\Omega_e} + (v, h)_{\Gamma_N} - (n \cdot \eta \nabla v, w)_{\Gamma_D} + (v, \lambda w)_{\Gamma_D} + (nq, w)_{\Gamma_D},
\]

\forall (v, q) \in (V_e, Q_e).

(9)

where $\Omega_e$ is the embedding domain, $(V_e, Q_e)$ are the spaces in Equation (6) defined over $\Omega_e$ and $a$ is a penalization parameter chosen as

\[
\begin{cases} 
\alpha = 1 & \text{in } \Omega, \\
\alpha = 0 & \text{in } \Omega_e \setminus \Omega. 
\end{cases}
\]

(10)

**Remark 4.** It can be seen, that with this choice of the penalization parameter, the weak form in Equation (9) recovers the weak form in Equation (8).

Finally, introducing discrete spaces $(V_{eh}, Q_{eh}) \subset (V_e, Q_e)$, the discrete bilinear and linear forms of the mixed finite cell formulation of the Stokes equations can be written as

\[
a(v_h, u_h) + b(v_h, p_h) = f(v_h), \\
b(q_h, u_h) + c(q_h, p_h) = g(q_h),
\]

with

\[
a(v_h, u_h) := (\eta \nabla v_h, a \nabla u_h)_{\Omega_{eh}} - (v_h, n \cdot \eta \nabla u_h)_{\Gamma_{eh}} - (n \cdot \eta \nabla v_h, u_h)_{\Gamma_{eh}} \\
+ (v_h, \lambda u_h)_{\Gamma_{eh}},
\]

(11)
where \( \Omega_{e,h} \) and \( \Gamma_{e,h} \) are appropriate discretizations of the embedding domain and the physical boundary, respectively with \( T_{e,h} := \{K_i\}_{i=1}^{n_k} \) and \( K_i \cap K_j = \emptyset \) for \( i \neq j \) being a tessellation of \( \Omega_e \) into a set of \( n_k \) compact, connected, Lipschitz sets \( K_i \) with nonempty interior, such that by \( \Omega_{e,h} := \bigcup_{i=1}^{n_k} K_i \) an approximation of \( \Omega_e \) is defined by the discrete embedding domain \( \Omega_{e,h} \), In the terminology of the finite cell method, \( K_i \) are often referred to as cells. The additional form \( c \) in Equation (11) corresponds to the stabilization term discussed below.

Remark 5. The penalization parameter \( \alpha \) in the discrete bilinear and linear forms in Equation (11) is taken as a small value \( (\alpha \ll 1) \) instead of absolute zero outside of the physical domain in order to mitigate the numerical ill-conditioning of the bilinear form.

The discrete form in Equation (11) leads to a linear system of equations of the form

\[
\begin{bmatrix}
A & B
\end{bmatrix}
\begin{bmatrix}
u
\end{bmatrix}
= -\begin{bmatrix}
f
\end{bmatrix},
\]

where the matrices \( A \in \mathbb{R}^{n_u \times n_u} \) and \( B \in \mathbb{R}^{n_u \times n_p} \) and the vectors \( f \in \mathbb{R}^{n_u} \) and \( g \in \mathbb{R}^{n_p} \) are defined according to Equation (12). \( u \in \mathbb{R}^{n_u} \) and \( p \in \mathbb{R}^{n_p} \) are the coefficients of expansion corresponding to the vector-valued velocity and scalar pressure functions, respectively. The matrix \( C \in \mathbb{R}^{n_p \times n_p} \) corresponds to the stabilization form \( c \) in Equation (11), which is either zero in the case of conforming spaces that satisfy the inf-sup condition or the stabilization block otherwise. \( n_u \) and \( n_p \) are the number of velocity and pressure coefficients of expansion, respectively.

It is well known that equal-order \( Q_p - Q_p \) and \( P_p - P_p \) elements do not satisfy the inf-sup condition of the Stokes equations, and therefore require stabilization. On the other hand, most conforming pairs often involve the need for data structures and computations that can be considered inconvenient from an implementation perspective compared to their equal-order counterparts. Therefore, stabilized equal-order elements, especially of low orders are extremely attractive from a computational perspective and are widely used in the engineering practice. Given that the embedding domain \( \Omega_e \) in the finite cell discretization is commonly chosen as a structured grid, and considering the optimization opportunities such discretizations provide, the choice of equal-order elements can be even more emphatically motivated. Many stable formulations, for example, based on a Petrov-Galerkin formulation for the \( C^0 \) interpolation of both velocity and pressure, the SUPG method, bubble-type stabilization methods, polynomial projection methods etc. have been studied in the literature. We employ a \( Q_1 - Q_1 \) discretization in this work and consider two stabilization methods. The first stabilization method is based on the discrete Laplacian, where the stabilization term \( c \) in Equation (11) is given by

\[
c(q_h, p_h) := -\beta \sum_{K_i \in T_h} h_{K_i}^2 (\nabla q_h, \nabla p_h)_{\Omega_{K_i}},
\]

where \( \Omega_{K_i} \) is the domain associated with \( K_i \), \( h_{K_i} \) is the diameter of \( K_i \) and \( \beta \) is a sufficiently large positive constant.

The second stabilization method is based on a local \( L^2 \) projection of the pressure. The general form of the stabilization term is given by

\[
c(q_h, p_h) := -\frac{1}{\eta} (q_h - \Pi q_h, p_h - \Pi p_h)_{\Omega_{K_i}},
\]

where \( \Pi \) is an appropriate projection operator. It can be shown that a local \( L^2 \) projection operator with a piecewise constant range \( \Pi_0 := L^2(\Omega_0) \rightarrow R_0 \) is an appropriate choice for the lowest equal order \( C^0 \) pairs where the projection of a given function \( q \) must satisfy

\[
\int_{\Omega_{K_i}} (\Pi_0 q - q) d\Omega_{K_i} = 0, \quad \forall K_i \in T_h.
\]
We note that both stabilization methods are local, involving independent computations that are restricted to the domain of $K_i$. Such properties are computationally attractive since the need for specialized data structures capable of handling neighborhood and/or edge computations are avoided on the one hand, and since the local computations can be carried out concurrently on the other hand. From an implementation perspective, the integration of the stabilization terms in both methods can easily be incorporated in the same units in charge of computing the linear and bilinear forms. Furthermore, while the first stabilization method involves the positive constant $\beta$ and the mesh-dependent variable $h_K$, the projection method is unconditionally stable, that is, does not involve mesh-dependent parameters.

We now focus on the mixed finite cell formulation of the incompressible Navier–Stokes equations. Navier–Stokes equations are nonlinear equations that can describe flow regimes outside of the scope of Stokes equations, for example, at high Reynolds numbers. The strong form of the Navier–Stokes equations is obtained by adding a nonlinear convection term to the Stokes equations (see Equation (1)) and is given by

\[ -\eta \nabla^2 \mathbf{u} + \mathbf{u} \cdot \nabla \mathbf{u} + \nabla p = f \quad \text{in } \Omega, \]
\[ \nabla \cdot \mathbf{u} = 0 \quad \text{in } \Omega. \]  

Equation (17) along with the boundary conditions in Equation (3) form the Navier–Stokes boundary value problem. The mixed finite cell weak form of the Navier–Stokes equations is obtained following the same procedure described above for the Stokes equations:

Find $(\mathbf{u}, p) \in (V_c, Q_c)$ such that

\[
F(\mathbf{u}, p) := (\eta \nabla \mathbf{v}, a \nabla \mathbf{u})_{\Omega} - (\mathbf{v}, \mathbf{n} \cdot \eta \nabla \mathbf{u})_{\Gamma D} - (\mathbf{n} \cdot \eta \nabla \mathbf{v}, \mathbf{u})_{\Gamma D} + (\mathbf{v}, \lambda \mathbf{u})_{\Gamma D} \\
+ (\mathbf{v}, \mathbf{u} \cdot \nabla \mathbf{u})_{\Omega} - (\mathbf{v} \cdot a \mathbf{p})_{\Omega} + (\mathbf{v}, q)_{\Omega} - (q, a \nabla \cdot \mathbf{u})_{\Omega} \\
+ (\mathbf{v}, a \alpha(\mathbf{h} - \mathbf{u}))_{\Omega} + (\mathbf{n} \cdot \eta \nabla \mathbf{v}, \mathbf{w})_{\Gamma D} - (\mathbf{v}, \lambda \mathbf{w})_{\Gamma D} - (\mathbf{n} \cdot \eta \nabla \mathbf{w})_{\Gamma D} = 0, \\
\forall (\mathbf{v}, q) \in (V_c, Q_c). \]

The weak form of the Navier–Stokes equations is solved using nonlinear iterative methods, where the iterations comprise a series of linearized problems. We discuss two linearization methods, namely the Newton’s method and the Picard’s method. Let $G(\mathbf{u}, p) : (V_c, Q_c) \rightarrow (V_c, Q_c)$ be a function such that $G(\mathbf{u}^*, p^*) = (\mathbf{u}^*, p^*)$ if $F(\mathbf{u}^*, p^*) = 0$. The iterations $(\mathbf{u}^{k+1}, p^{k+1}) = G(\mathbf{u}^k, p^k)$ can then be shown to converge to $(\mathbf{u}^*, p^*)$ if $G$ is a contraction. \(^{55}\)

The Newton iteration can then be formulated as $G_{\text{Newton}} := (\mathbf{u}^*, p^*) + \frac{R(\delta \mathbf{u}^k, \delta p^k)}{F'(\delta \mathbf{u}^k, \delta p^k)}$, where $R$ is the residual and $F'$ is the Fréchet derivative of the weak form in Equation (18). The Newton iteration is commonly formulated as a function of $(\delta \mathbf{u}^k, \delta p^k) := (\mathbf{u}^{k+1} - \mathbf{u}^k, p^{k+1} - p^k)$ as

\[
F'(\mathbf{u}^k, p^k) (\delta \mathbf{u}^k, \delta p^k) = R(\mathbf{u}^k, p^k),
\]

with

\[
F'(\mathbf{u}^k, p^k) (\delta \mathbf{u}^k, \delta p^k) := (\eta \nabla \mathbf{v}, a \nabla \delta \mathbf{u}^k)_{\Omega} - (\mathbf{v}, \mathbf{n} \cdot \eta \nabla \delta \mathbf{u}^k)_{\Gamma D} - (\mathbf{n} \cdot \eta \nabla \mathbf{v}, \delta \mathbf{u}^k)_{\Gamma D} \\
+ (\mathbf{v}, \lambda \delta \mathbf{u}^k)_{\Gamma D} + (\mathbf{v}, \mathbf{u}^k \cdot \nabla \delta \mathbf{u}^k)_{\Omega} + (\mathbf{v}, \delta \mathbf{u}^k \cdot \nabla \mathbf{u}^k)_{\Omega} - (\mathbf{v} \cdot a \delta p^k)_{\Omega} + (\mathbf{v}, n \delta p^k)_{\Gamma D} \\
+ (\mathbf{n} \cdot \eta \nabla \mathbf{v}, \mathbf{w})_{\Gamma D} - (\mathbf{v}, \lambda \mathbf{w})_{\Gamma D} - (\mathbf{n} \cdot \eta \nabla \mathbf{w})_{\Gamma D} = 0, \\
\forall (\mathbf{v}, q) \in (V_c, Q_c). \]

The Picard iteration is defined as $G_{\text{Picard}} := \frac{1}{p^k} \mathcal{K}$ and is commonly formulated as a direct iteration for $(\mathbf{u}^{k+1}, p^{k+1})$ as

\[
P(\mathbf{u}^{k+1}, p^{k+1}) = \mathcal{K}
\]
with

\[ \mathcal{P}(u^{k+1}, p^{k+1}) := (\eta \nabla v, a \nabla u^{k+1})_{\Omega} - (v, n \cdot \eta \nabla u^{k+1})_{\Gamma_0} - (n \cdot \eta \nabla u^{k+1})_{\Gamma_0} \\
+ (v, \lambda u^{k+1})_{\Gamma_0} + (v, u^k \cdot \nabla u^{k+1})_{\Omega} - (\nabla \cdot v, a p^{k+1})_{\Omega} + (v, n p^{k+1})_{\Gamma_0} \\
+ (n q, u^{k+1})_{\Gamma_0} - (q, a \nabla \cdot u^{k+1})_{\Omega_e} , \]  

(23)

and

\[ \mathcal{K} := -(v, a f)_{\Omega} - (v, h)_{\Gamma_N} + (n \cdot \eta \nabla v, w)_{\Gamma_0} - (v, \lambda w)_{\Gamma_0} - (n q, w)_{\Gamma_0} . \]  

(24)

\[ \text{Remark 6.} \text{ The Picard’s method linearizes Equation (18) by fixing the convection coefficient at the current velocity and can also be derived from the Newton’s method by dropping the term} (v, \delta u \cdot \nabla u^k)_{\Omega} \text{ in Equation (20).} \]

\[ \text{Remark 7.} \text{ The Newton’s method can be shown to achieve quadratic convergence when the current iteration becomes sufficiently close to the solution, while the Picard’s method achieves linear convergence in the general case. On the other hand, the sphere of convergence of the Picard’s method is significantly larger compared to the Newton’s method, that is, the Picard’s method is more likely to converge to the solution for an initial guess that is far from the solution.} \]

Note that analogous to the Stokes equations, the \( Q_1 - Q_1 \) discretization of the Navier–Stokes equations requires stabilization as discussed above, see Equations (14) and (15).

3 ADAPTIVE GEOMETRIC MULTIGRID

Multigrid methods are typically employed in one of two fashions for the solution of saddle-point problems: in the first approach, multigrid is used to treat the entire system, and the coupling between velocity and pressure is therefore preserved. The second approach is to use a variant of Schur complement schemes and use multigrid to solve the resultant systems associated with the velocity block and the Schur complement system. In general, the coupling between velocity and pressure is at least partially lost in the latter approach. In this work, we focus on the former approach and develop a monolithic adaptive geometric multigrid method for the mixed finite cell formulation of saddle-point problems in the context of Stokes and Navier–Stokes systems.

We consider a linear system of the form

\[ Lx = b, \]  

(25)

where \( L := \begin{pmatrix} A & B \\ B^T & C \end{pmatrix} \) is defined according to the bilinear form of the discretized weak problem, \( x := \begin{pmatrix} u \\ p \end{pmatrix} \) is the solution vector and \( b := \begin{pmatrix} f \\ 0 \end{pmatrix} \) is defined according to the linear form of the discretized weak problem. Multigrid methods try to find the solution to the system in Equation (25) from the solutions to a series of smaller problems. The main assumptions are that high-frequency errors can be effectively eliminated on finer discretizations and smooth errors can be adequately represented on coarser discretizations. In geometric multigrid methods, the problem hierarchy is generated as a set of \( n_g \) progressively coarser discretizations, \( \Omega_{e,h}^1 \ldots \Omega_{e,h}^{n_g} \), where \( \Omega_{e,h}^1 \) corresponds to the finest problem and \( \Omega_{e,h}^{n_g} \) is referred to as the base grid and corresponds to the coarsest problem. We discuss the main components of the geometric multigrid method below, which in addition to the grid hierarchy include transfer operators, smoothers and the base solver.

We make use of space tree data structures for the spatial discretization of the computational domain. Mesh refinement is performed through bisection in the physical space and corresponds to the division of a coarse cell into its \( 2^d \) children, where \( d \) is the dimension. We impose the usual 2:1 balance such that neighboring cells are apart by at most one level of refinement. As a consequence of AMR, hanging nodes are introduced in the discretized system. We handle hanging nodes as constraints and remove them from the global system of equations. We generate the grid hierarchy for geometric multigrid top-down. Given the fine grid \( \Omega_{e,h}^1 \), the immediate coarse grid \( \Omega_{e,h}^{2-1} \) is obtained according to Algorithm 1. This process is repeated \( n \) times to generate a set of \( n \) nested spaces. The multigrid V cycle with adaptive refinement on a sample domain is illustrated in Figure 2. Transfer operators are responsible for the flow of information between grid levels, that is, restriction from grid level \( l \) to \( l-1 \) and prolongation from grid level \( l-1 \) to \( l \) and can be represented as
Algorithm 1. Generation of the coarse grid $\Omega_{e,h}^{l-1}$ from the fine grid $\Omega_{e,h}^l$. The coarsen operation replaces $K$ and all of its siblings with their parent. The balance operation is the standard 2:1 balancing between adjacent tree leaves.

input : $\Omega_{e,h}^l$
output: $\Omega_{e,h}^{l-1}$

$\Omega_{e,h}^{l-1} \leftarrow \Omega_{e,h}^l$

// $r_K$ is the refinement level of $K$
$r_{max} \leftarrow \max(r_K \forall K \in \Omega_{e,h}^{l-1})$

for $K \in \Omega_{e,h}^{l-1}$ do
  if $r_K == r_{max}$ then
    coarsen $K$
  end
end

Balance $\Omega_{e,h}^{l-1}$;

![Figure 2](image-url)  
A four-level multigrid V cycle along with the adaptive refinement of a sample domain around an arc. [Colour figure can be viewed at wileyonlinelibrary.com]

$$w^{l-1} = R'w',$$
$$w^l = P^{l-1}w^{l-1},$$

(26)

where $R$ and $P$ are the restriction and prolongation operators, respectively, and $w$ is a vector.\textsuperscript{22,56}

In order to simplify the notation, we consider a given grid and drop the grid level $l$ from the notation in the following. Given the system in Equation (25), a series of iterative corrections can be formed as follows

$$x^{k+1} = x^k + S(b - Lx^k),$$

(27)

where $S$ is the smoother operator and $k$ is the iteration number. The success of the multigrid method heavily depends on the smoother operator. The Vanka smoother\textsuperscript{41} has been shown to be an effective choice for the finite difference and
finite element discretizations of flow problems. In unfitted finite element methods, it is known that small cut configurations, where the physical boundary intersects a small fraction of a cell lead to severe ill-conditioning of the system matrix. Therefore, the treatment of cut cells in the smoother is an essential aspect of multigrid methods for the finite cell formulation of saddle-point problems. We present two multiplicative Vanka-type smoothers for the model problems in the context of Schwarz domain decomposition methods. Given a set of $n_S$ subdomains $S := \{ S_1, \ldots, S_{n_S} \}$, where subdomain $S_i$ consists of a subset of the degrees of freedom in the general system $L$, the general framework of the Schwarz method is to obtain local corrections to the global system based on the solution of the local systems of the subdomains.

Given the associated subdomain restriction operators $R_i$, the local system of subdomain $S_i$ is given by $L_i := R_i L R_i^T$. The restriction operator $R_i$ extracts the set of DoFs in $S_i$ and its transpose $R_i^T$ injects a vector from the local subspace of the subdomain to the global space. We define the first variant, denoted as the cell-based smoother, as

$$S_{\text{cell}} = \left[ I - \prod_{i=1}^{n_K} \left( I - R_i^T \omega_i L_i^{-1} R_i L_i \right) \right] L_i^{-1}, \quad \text{(28)}$$

where $n_K$ is the number of cells in $\Omega_{e,h}$, and $\omega_i$ is in general a diagonal weighting matrix. The subdomain $S_{\text{cell}}$, contains all the pressure degrees of freedom that appear on $K_i$ and all the velocity degrees of freedom that are connected to them. The connection, that is, coupling, between pressure DoF $p_i$ and the velocity DoFs in the discretized bilinear form algebraically corresponds to the nonzero entries in the $j$th row of $B^T$ (see Equation (13)). The cell-based subdomains are illustrated in Figure 3A for the $Q_1 - Q_1$ discretization.

We define a variant of the cell-based smoother next. In this variant, a cell-based subdomain is generated for every cutcell as explained above. For all pressure degrees of freedom $p_i$ that do not appear in any of the cell-based subdomains, a node-based subdomain is generated that contains $p_i$ along with all the velocity DoFs connected to it, see Figure 3B. We denote this variant as the cutcell-based smoother, whose operator can be written as

$$S_{\text{cutcell}} = \left[ I - \prod_{i=1}^{n_S} \left( I - R_i^T \omega_i L_i^{-1} R_i L_i \right) \right] L_i^{-1}, \quad \text{(29)}$$

where the number of subdomains $n_S$ is dependent on the cut configuration of the domain. Both the cell-based and cutcell-based smoothers can be interpreted as multiplicative Schwarz methods, and we note that, conceptionally, the above ideas can be applied to other saddle-point problems.

Remark 8. The majority of subdomains in the cutcell-based smoother have a smaller dimension than their counterparts in the cell-based smoother since the physical domain typically intersects only a small fraction of all the cells in the computational domain. Therefore, the computational cost of $S_{\text{cutcell}}$ is expected to be noticeably lower than $S_{\text{cell}}$. On the other hand, larger subdomains usually increase the effectiveness of the smoother. The performance and computational cost of both smoothers are analyzed in Section 4.

4 NUMERICAL BENCHMARKS

We consider a number of numerical studies to investigate the performance of the developed geometric multigrid method using two well-known benchmarks commonly used for the verification of flow applications, namely the driven cavity benchmark and the cylinder flow benchmark. The range of solution strategies, including the geometric multigrid settings, the choice of the smoother operator, whether geometric multigrid is used as a standalone solver or used inside a Krylov accelerator, the stabilization method, the linearization method for the Navier–Stokes equations, etc. gives rise to a large number of possible configurations. Nevertheless, we try to discuss the most important aspects of such configurations in the following.

The kinematic viscosity of the fluid is $\eta = \frac{1}{1000}$ m$^2$/s, the finite cell penalization factor is $\alpha = 10^{-10}$ and the stabilization parameter is $\beta = 10$ for the cases where the stabilization method in Equation (14) is used. We use geometric multigrid both as a solver and as a preconditioner in Krylov subspace solvers. Specifically, we wrap GMG inside the BiCGSTAB method.

We note that Krylov accelerators are expected to improve the performance of geometric multigrid methods in general; however, in order to better analyze the performance of the developed multigrid method for the systems of equations that arise from the model problems, we use multigrid also as a standalone fixed-point iterator to exclude the added effect of
FIGURE 3  (A) Subdomains in the cell-based smoother. The designated degrees of freedom belong to the subdomain of the shaded cell. A subdomain is generated for every cell in the domain in the same manner. And (B) subdomains in the cutcell-based smoother. A cell-based subdomain is generated for cells cut by the physical boundary (shaded cells in (B)) according to (A), while a node-based subdomain is generated for all pressure nodes that do not appear in any of the cell-based subdomains. A node-based subdomain is illustrated for the designated pressure node in (B). The arrows demonstrate how hanging nodes are implicitly included in the subdomains through their nonhanging counterparts. [Colour figure can be viewed at wileyonlinelibrary.com]

Krylov accelerators. We consider the two smoother variants introduced in Section 3 within a V-cycle in order to study the performance of the solver with respect to the choice of smoothers and the required computational cost. A damping factor of $2/3$ is used, which is observed to be the minimum amount of damping necessary to obtain convergence. The smoothers differ solely in the Schwarz subdomains. A subdomain is generated for each cell in the first variant. On the other hand, cell-based subdomains are only generated for pressure nodes that appear in a cutcell and node-based subdomains are employed for pressure nodes that do not appear in any cutcell in the second variant. In both cases, the subdomains follow the Morton order (Z-order) of the space tree data structure. The local system of the subdomains are solved using a direct solver down to machine accuracy. We note that for constant viscosity, the local system of subdomains of the same size remain the same; however, such optimizations are intentionally not considered in the reported runtimes in order for the results to be relevant also to the more general case. The restriction and prolongation operators are based on the standard finite element embedding. All examples are executed on an Intel Xeon Skylake Gold 6148 CPU with 192 GB of main memory using an in-house C++ implementation with p4est\textsuperscript{47} and PETSc\textsuperscript{57} for mesh manipulation and some linear algebra components, respectively, and Paraview\textsuperscript{58} for visualization and postprocessing.

The convergence of the solver is analyzed through a mesh study using a progressively finer set of problems. A combination of uniform and adaptive refinement steps is employed to produce the computational meshes $\{\Omega_{e,h}^i\}_{i=0}^n$, where $n$ is the number of grids in the convergence study. $\Omega_{c,h}^1$ is used as the coarse grid for all problems, that is, finer problems
TABLE 1  The grid hierarchy used in the convergence study of the driven cavity benchmark.

| Grid   | $n_c$ | $n_{DoF}$ | $n^i_{DoF}/n^{i-1}_{DoF}$ |
|--------|-------|-----------|-----------------------------|
| $\Omega_{e,h}^1$ | 256   | 867       | --                          |
| $\Omega_{e,h}^2$ | 832   | 2,643     | 2.97                        |
| $\Omega_{e,h}^3$ | 2,704 | 8,331     | 3.15                        |
| $\Omega_{e,h}^4$ | 9,184 | 27,891    | 3.34                        |
| $\Omega_{e,h}^5$ | 31,828| 96,027    | 3.44                        |
| $\Omega_{e,h}^6$ | 110,824| 333,363 | 3.47                        |
| $\Omega_{e,h}^7$ | 378,088| 1,135,731 | 3.40                       |
| $\Omega_{e,h}^8$ | 1,039,516| 3,109,779 | 2.74                       |

Note: $n_c$ and $n_{DoF}$ are the number of cells and the number of degrees of freedom, respectively.

employ a larger number of grid hierarchies. $M_S$ designates a problem with $\Omega_{e,h}^1$ as the fine grid and consequently $i$ levels of grid hierarchy. $S$ denotes the smoother operator and is either cell for the cell-based smoother or cutcell for the cutcell-based smoother. The coarse grid $\Omega_{e,h}^1$ is solved using a direct solver down to machine accuracy. We note that the computational cost of the coarse grid solver is comparatively insignificant, especially for larger problems, given that $\Omega_{e,h}^1$ remains relatively small and constant in size, see Tables 1 and 4. Adaptive refinement is performed towards the boundaries of the physical domain, see Figures 4 and 6, which is generally observed to improve the solution quality of finite cell applications. In addition to adaptive refinement, eight steps of adaptive integration are used for cutcells.

The driven cavity benchmark describes the flow of a fluid inside an enclosed cavity, where a constant horizontal velocity drives the fluid on the top. We take the unit square as the computational domain and use $u_c = 1$ m/s on the top edge and no-slip wall conditions elsewhere. The grid hierarchy in Table 1 is produced by adaptive refinement towards the boundaries of the domain, see Figure 4. We study the convergence of the solver with different configurations for the Stokes equations. Three steps of pre and postsmoothing are used. The iteration count of the solver in each case is shown in Table 2. The stop criterion is to reduce the relative residual by $10^{-9}$. It can be seen that the iteration count of GMG both as a solver and used within the BiCGSTAB method remains bounded. Furthermore, it can be observed that the use of the cell-based smoother, compared to the cutcell-based smoother, and the addition of the Krylov accelerator in general improve the convergence of the solver. Similarly, the $Q_1 - Q_1$ element with the $L^2$ projection stabilization requires fewer iterations in general. Although the difference between the two stabilization methods is negligible for deeper GMG hierarchies, we consider the $Q_1 - Q_1$ element with the $L^2$ projection stabilization in the remainder of the benchmarks.

Given the different computational cost associated with each iteration of the solver depending on the particular configuration and each smoothing step depending on the smoother operator, we solve the Problem $M^7$ from Table 2 using different numbers of smoothing steps. The iteration count of the solver as well as its normalized relative runtime is reported in Table 3. It can be seen that the fastest time to solution is achieved using one pre and postsmoothing step of the cutcell-based smoother and using GMG within the BiCGSTAB method. Moreover, additional smoothing steps of the cutcell-based smoother lead to a steep increase in the runtime of the solver in spite of the reduced iteration count because of the high computational complexity of the cutcell-based smoother. These observations shall be further consolidated when we discuss the performance of the solver in the cylinder flow benchmark below.

We consider the cylinder flow benchmark next, where a channel with dimensions $2.2 \times 0.41$ m with a cylinder with a diameter of $0.1$ m near the inflow is studied. The geometry of the channel is shown in Figure 5. The parabolic inflow boundary condition is described as

$$u_c(0, y) = \frac{4\bar{u}y(H - y)}{H^2}, \quad y \in [0, H],$$
$$u_c(0, y) = 0, \quad y \in [0, H],$$

where $\bar{u} = 0.3$ m/s and $H$ is the width of the channel. The wall boundary condition on the top and bottom sides is nonslip, see Figure 5. The computational meshes are produced using adaptive refinement towards the channel and around the cylinder as shown in Figure 6A. We note that despite the relatively simple geometry of the benchmark, the curved surface
FIGURE 4  (A) The computational mesh for the driven cavity benchmark, where the domain is refined towards the boundaries of the cavity. The color map indicates the refinement level of the cell. (B) The streamlines of the solution. [Colour figure can be viewed at wileyonlinelibrary.com]

TABLE 2  The convergence of different solver configurations for the driven cavity benchmark with the Stokes equations.

| Problem | GMG | GMG + BiCGSTAB |
|---------|-----|----------------|
|         | n^k_{cutcell} | n^k_{cell} | n^{k,2}_{cutcell} | n^{k,2}_{cell} | n^k_{cutcell} | n^k_{cell} | n^{k,2}_{cutcell} | n^{k,2}_{cell} |
| M^2     | 14  | 3  | 8  | 3  | 4  | 2  | 3  | 1  |
| M^3     | 10  | 3  | 8  | 2  | 4  | 2  | 3  | 1  |
| M^4     | 9   | 3  | 7  | 3  | 4  | 2  | 3  | 2  |
| M^5     | 8   | 3  | 6  | 3  | 3  | 2  | 3  | 2  |
| M^6     | 6   | 4  | 5  | 4  | 3  | 2  | 2  | 2  |
| M^7     | 6   | 4  | 5  | 3  | 3  | 2  | 2  | 2  |
| M^8     | 4   | 3  | 4  | 3  | 2  | 2  | 2  | 2  |

Note: n^k is the iteration count of the solver, where the stop criterion is the reduction of the relative residual by 10^{-9}. \Delta and L^2 indicate the stabilization methods in Equations (14) and (15), respectively. cutcell and cell indicate the cutcell-based smoother and the cell-based smoother, respectively. BiCGSTAB indicates the cases where geometric multigrid is used as a preconditioner. Otherwise, geometric multigrid is used as a solver. The problems M^i are defined according to the grid hierarchy in Table 1.
TABLE 3 The iteration count and normalized relative runtime of different solver configurations for Problem $M^7$ in the driven cavity benchmark with the Stokes equations, see Table 1.

| # pre + postsmoothing | GMG \( (n_{it}, t_{rel})^{\text{cutcell}} \) | GMG + BiCGSTAB \( (n_{it}, t_{rel})^{\text{cutcell}} \) | GMG + BiCGSTAB \( (n_{it}, t_{rel})^{\text{cell}} \) |
|-----------------------|---------------------------------------------|---------------------------------------------|---------------------------------------------|
| 1 + 1                 | 11, 0.35                                    | 4, 0.25                                     | 2, 0.26                                    |
| 2 + 2                 | 6, 0.37                                     | 3, 0.37                                     | 2, 0.52                                    |
| 3 + 3                 | 5, 0.45                                     | 2, 0.37                                     | 2, 0.77                                    |
| 4 + 4                 | 4, 0.48                                     | 2, 0.49                                     | 2, 1.00                                    |

Note: $n_{it}$ is the iteration count of the solver, where the stop criterion is the reduction of the relative residual by $10^{-9}$. $t_{rel}$ is the normalized runtime of the solver. cutcell and cell indicate the cutcell-based smoother and the cell-based smoother, respectively. BiCGSTAB indicates the cases where geometric multigrid is used as a preconditioner. Otherwise, geometric multigrid is used as a solver.

FIGURE 5 The geometry and boundary conditions of the cylinder flow benchmark. Note that the outlet boundary condition can equivalently be described using a homogeneous Neumann boundary condition. The inlet boundary condition is defined in Equation (30). The wall boundary condition is no-slip.
FIGURE 6 The cylinder flow benchmark: (A) adaptive refinement of the computational mesh towards the boundaries of the channel and the cylinder, where the red and blue spectra indicate lower and higher refinement levels, respectively, (B) the velocity magnitude and streamlines for the Stokes problem and (C) the velocity magnitude and streamlines for the Navier–Stokes problem. Note that only the left part of the domain is shown in (B,C). [Colour figure can be viewed at wileyonlinelibrary.com]

TABLE 4 The grid hierarchy used in the convergence study of the cylinder flow benchmark.

| Grid   | \( n_c \) | \( n_{\text{DoF}} \) | \( n^i_{\text{DoF}} / n^{i-1}_{\text{DoF}} \) |
|--------|----------|----------------|---------------------------------|
| \( \Omega^1_{e,h} \) | 1,024    | 3,315          | –                               |
| \( \Omega^2_{e,h} \) | 4,096    | 12,771         | 2.42                            |
| \( \Omega^3_{e,h} \) | 11,497   | 35,076         | 2.75                            |
| \( \Omega^4_{e,h} \) | 26,164   | 78,906         | 2.25                            |
| \( \Omega^5_{e,h} \) | 55,081   | 165,294        | 2.09                            |
| \( \Omega^6_{e,h} \) | 112,795  | 337,740        | 2.04                            |
| \( \Omega^7_{e,h} \) | 215,539  | 644,616        | 1.91                            |
| \( \Omega^8_{e,h} \) | 419,371  | 1,253,445      | 1.94                            |
| \( \Omega^9_{e,h} \) | 809,956  | 2,419,908      | 1.93                            |

Note: \( n_c \) and \( n_{\text{DoF}} \) are the number of cells and the number of degrees of freedom, respectively.
FIGURE 7  The convergence of the geometric multigrid solver for the Stokes flow in the cylinder flow benchmark problem. Geometric multigrid is used as a solver. The grid hierarchy given in Table 4 is used for the mesh study. [Colour figure can be viewed at wileyonlinelibrary.com]

FIGURE 8  The performance of the two smoother variants within the geometric multigrid solver for the Stokes flow in the cylinder flow benchmark problem: (A) the average reduction rate of the solver, (B) the normalized time per geometric multigrid iteration, and (C) the normalized total runtime of the solver. The grid hierarchy given in Table 4 is used. [Colour figure can be viewed at wileyonlinelibrary.com]
Table 5: The convergence of different solver configurations for the Navier–Stokes equations in the cylinder flow benchmark.

| Solver           | Cutcell | Cell  | $(n_{n}^{\text{nonlinear}}, n_{n}^{\text{linear}})$ |
|------------------|---------|-------|-----------------------------------------------------|
| Newton GMG       | cutcell | (6, 15) | $(M^6, M^7)$ |
|                  | cell    | (6, 12) | $(5, 12)$ |
| Newton GMG + BiCGSTAB | cutcell | (6, 9) | $(M^6)$ |
|                  | cell    | (6, 6) | $(5, 5)$ |
| Picard GMG       | cutcell | (12, 29) | $(M^6)$ |
|                  | cell    | (12, 24) | $(M^7)$ |
| Picard GMG + BiCGSTAB | cutcell | (12, 18) | $(M^6)$ |
|                  | cell    | (12, 12) | $(M^7)$ |

Note: $n_{n}^{\text{nonlinear}}$ and $n_{n}^{\text{linear}}$ are the iteration count of the nonlinear solver and the total number of linear iterations, respectively. The stop criteria are the reduction of the relative residual by $10^{-9}$ for the nonlinear solver and $10^{-2}$ at each linearization point for the linear solver. cutcell and cell indicate the cutcell-based smoother and the cell-based smoother, respectively. BiCGSTAB indicates the cases where geometric multigrid is used as a preconditioner. Otherwise, geometric multigrid is used as a solver. Because of the large number of configurations, the four finest problems, defined according to the grid hierarchy is Table 4 are presented.

Each linearization point, the linear iteration is stopped when the relative residual is reduced by $10^{-2}$. The stop criterion for the nonlinear solver is the reduction of the relative residual by $10^{-9}$. The initial guess is the zero vector. The velocity profile of the fluid near the cylinder is shown in Figure 6C. The iteration count of the nonlinear solver as well as the total number of linear iterations are given in Table 5. It can be seen that the Newton linearization requires fewer iterations in general. The linear solver demonstrates a similar behavior compared to the linear case. More specifically, the iteration count remains bounded and the addition of the BiCGSTAB method noticeably reduces the number of linear iterations, and in the majority of cases, only one or two iterations are sufficient to achieve the required relative reduction of the residual.

Remark 9. We note that the reduction of the relative residual by a larger factor, that is, more linear iterations at the linearization points does not significantly affect the overall convergence of the nonlinear iteration for the presented benchmark. Therefore, in favor of minimizing computational cost, we reduce the relative residual by $10^{-2}$.

Remark 10. We note that the initial guess for the solution of the Navier–Stokes system becomes increasingly important for higher Reynolds numbers. In such cases, one standard strategy is to solve a series of problems with increasing Reynolds numbers and use the solution to the Stokes system as the initial guess for the first problem.

As expected, the overall convergence of the nonlinear iteration is not significantly affected by the choice of smoother in the geometric multigrid method, and the solver demonstrates excellent convergence independently of the problem size and hierarchy depth for the presented benchmark. We note that minor differences can occur since the linearized systems are not solved to the same exact residual as the relative residual is reduced by $10^{-2}$.

5 | Conclusions

The unfitted finite element methods circumvent the need for the generation of a boundary-conforming mesh, which is a bottleneck in classical finite element methods. On the other hand, the discrete systems arising from such methods, including the finite cell method, are typically ill-conditioned which in turn renders the efficient solution of such problems a challenging task. We employed an adaptive geometric multigrid method for the solution of the mixed finite cell formulation of saddle-point problems in the context of the Stokes and Navier–Stokes equations. We presented two smoothers tailored to the finite cell formulation of saddle-point problems and studied the convergence of the multigrid method both as a standalone solver and as a preconditioner in Krylov accelerators through a number of numerical benchmarks. The influence of various factors such as the stabilization method and the number of smoothing steps on the performance of the solver as well as the scalability of the solver in terms of iteration count and runtime with respect to the problem size and the depth of the multigrid hierarchy were studied. We note that further case studies, for example, for 3D problems...
would be an interesting extension to this work. It is seen that both the cell-based smoother and the cutcell-based smoother are effective for the solution of the model problems. While the cell-based smoother is slightly more effective per iteration, the cutcell-based smoother is more favorable in terms of computational cost and achieves a faster total solution time in the presented benchmarks. Moreover, the addition of the BiCGSTAB method is observed to reduce the iteration count of the solver. Results indicate that the multigrid method both as a solver and as a preconditioner in a Krylov subspace solver can effectively solve the linear(ized) systems independently of the problem size. Furthermore, it is seen that the multigrid method is robust with respect to the grid hierarchy, allowing the solution to large problems to be obtained efficiently using a small coarse grid.

ACKNOWLEDGMENTS

Financial support was provided by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) in the framework of subproject C4 of the Collaborative Research Center SFB 837 Interaction Modeling in Mechanized Tunneling, grant number 77309832. This support is gratefully acknowledged. We also gratefully acknowledge the computing time on the computing cluster of the SFB 837 and the Department of Civil and Environmental Engineering at Ruhr University Bochum, which was used for the presented numerical studies. Open Access funding enabled and organized by Projekt DEAL. [Correction added on 18 May 2023, after first online publication: Projekt DEAL funding statement has been added.]

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

S. Saberi https://orcid.org/0000-0001-9962-9244

REFERENCES

1. Belytschko T, Moës N, Usui S, Parimi C. Arbitrary discontinuities in finite elements. Int J Numer Methods Eng. 2001;50(4):993-1013.
2. Burman E, Claus S, Hansbo P, Larson MG, Massing A. Cutfem: Discretizing geometry and partial differential equations. Int J Numer Methods Eng. 2015;104(7):472-501.
3. Parvizian J, Düster A, Rank E. Finite cell method. Comput Mech. 2007;41(1):121-133.
4. Düster A, Parvizian J, Yang Z, Rank E. The finite cell method for three-dimensional problems of solid mechanics. Comput Methods Appl Mech Eng. 2008;197(45-48):3768-3782.
5. Schillinger D, Ruess M. The finite cell method: A review in the context of higher-order structural analysis of cad and image-based geometric models. Arch Comput Methods Eng. 2015;22(3):391-455.
6. Fernández-Méndez S, Huerta A. Imposing essential boundary conditions in mesh-free methods. Comput Methods Appl Mech Eng. 2004;193(12-14):1257-1275.
7. Flemisch B, Wohlmuth BI. Stable Lagrange multipliers for quadrilateral meshes of curved interfaces in 3d. Comput Methods Appl Mech Eng. 2007;196(8):1589-1602.
8. Glowinski R, Kuznetsov Y. Distributed Lagrange multipliers based on fictitious domain method for second order elliptic problems. Comput Methods Appl Mech Eng. 2007;196(8):1498-1506.
9. Burman E, Hansbo P. Fictitious domain finite element methods using cut elements: I. A stabilized Lagrange multiplier method. Comput Methods Appl Mech Eng. 2010;199(41-44):2680-2686.
10. Babuška I. The finite element method with penalty. Math Comput. 1973;27(122):221-228.
11. Zhu T, Atluri SN. A modified collocation method and a penalty formulation for enforcing the essential boundary conditions in the element free Galerkin method. Comput Meth. 1998;21(3):211-222.
12. Burman E. Ghost penalty. Comptes Rendus Math. 2010;348(21):1217-1220. doi:10.1016/j.crma.2010.10.006
13. Nitsche J. Über ein variationsprinzip zur lösung von dirichlet-problemen bei verwendung von teilkümen, die keinen randbedingungen unterworfen sind. Abhandlungen aus dem mathematischen Seminar der Universität Hamburg. Vol 36. Springer; 1971:9-15.
14. Hansbo A, Hansbo P. An unfitted finite element method, based on Nitsche's method, for elliptic interface problems. Comput Methods Appl Mech Eng. 2002;191(47-48):5537-5552.
15. Dolbow J, Harari I. An efficient finite element method for embedded interface problems. Int J Numer Methods Eng. 2009;78(2):229-252.
16. Embar A, Dolbow J, Harari I. Imposing dirichlet boundary conditions with Nitsche's method and spline-based finite elements. Int J Numer Methods Eng. 2010;83(7):877-898.
17. Burman E, Hansbo P. Fictitious domain finite element methods using cut elements: II. A stabilized Nitsche method. Appl Numer Math. 2012;62(4):328-341.
18. Hoang T, Verhoosel CV, Ferdinando Auricchio E, van Brummelen H, Reali A. Mixed isogeometric finite cell methods for the Stokes problem. *Comput Methods Appl Mech Eng*. 2017;316:400-423.
19. Elman HC, Golub GH. Inexact and preconditioned Uzawa algorithms for saddle point problems. *SIAM J Numer Anal*. 1994;31(6):1645-1661.
20. Bramble JH, Pasciak JE, Vassilev AT. Analysis of the inexact Uzawa algorithm for saddle point problems. *SIAM J Numer Anal*. 1997;34(3):1072-1092.
21. Verfürth R. A multilevel algorithm for mixed problems. *SIAM J Numer Anal*. 1984;21(2):264-271.
22. Hackbusch W. *Multi-grid Methods and Applications*. Springer Series in Computational Mathematics. Springer; 1985.
23. Wittum G. Multi-grid methods for stokes and navier-stokes equations. *Numer Math*. 1989;54(5):543-563.
24. Bramble JH, Pasciak JE. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Math Comput*. 1988;50(181):1-17.
25. Rusten T, Winther R. A preconditioned iterative method for saddlepoint problems. *SIAM J Matrix Anal Appl*. 1992;13(3):887-904.
26. Silvester D, Wathen A. Fast iterative solution of stabilised stokes systems part II: Using general block preconditioners. *SIAM J Numer Anal*. 1994;31(5):1352-1367.
27. Saad Y. *Iterative methods for sparse linear systems*. Vol 82. siam; 2003.
28. Elman HC. Multigrid and Krylov subspace methods for the discrete Stokes equations. *Int J Numer Methods Fluids*. 1996;22(8):755-770.
29. Peters J, Reichelt V, Reusken A. Fast iterative solvers for discrete Stokes equations. *SIAM J Sci Comput*. 2005;27(2):646-666.
30. Larin M, Reusken A. A comparative study of efficient iterative solvers for generalized Stokes equations. *Numer Linear Algebra Appl*. 2008;15(1):13-34.
31. Patankar SV, Spalding DB. A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows. *Numerical prediction of flow, heat transfer and combustion*. Elsevier; 1972:54-73.
32. Silvester D, Elman H, Kay D, Wathen A. Efficient preconditioning of the linearized navier–stokes equations for incompressible flow. *J Comput Appl Math*. 2001;128(1-2):261-279.
33. Kay D, Loghin D, Wathen A. A preconditioner for the steady-state Navier–Stokes equations. *SIAM J Sci Comput*. 2002;24(1):237-256.
34. Elman H, Howle VE, Shadid J, Shuttlesworth R, Tuminaro R. Block preconditioners based on approximate commutators. *SIAM J Sci Comput*. 2006;27(5):1651-1668.
35. Klawonn A, Pavarino LF. Overlapping Schwarz methods for mixed linear elasticity and Stokes problems. *Comput Methods Appl Mech Eng*. 1998;165(1-4):233-245.
36. Klawonn A, Pavarino LF. A comparison of overlapping Schwarz methods and block preconditioners for saddle point problems. *Numer Linear Algebra Appl*. 2000;7(1):1-25.
37. Pavarino LF. Indefinite overlapping Schwarz methods for time-dependent stokes problems. *Comput Methods Appl Mech Eng*. 2000;187(1-2):35-51.
38. Schäfer M, Turek S, Durst F, Krause E, Rannacher R. Benchmark computations of laminar flow around a cylinder. *Flow simulation with high-performance computers II*. Springer; 1996:547-566.
39. Gjesdal T, Lossius MEH. Comparison of pressure correction smoothers for multigrid solution of incompressible flow. *Int J Numer Methods Fluids*. 1997;25(4):393-405.
40. Braess D, Sarazin R. An efficient smoother for the Stokes problem. *Appl Numer Math*. 1997;23(1):3-19.
41. Vanka SP. Block-implicit multigrid solution of Navier-Stokes equations in primitive variables. *J Comput Phys*. 1986;65(1):138-158.
42. Jomo JN, Zander N, Elhaddad M, et al. Parallelization of the multi-level hp-adaptive finite cell method. *Comput Math Appl*. 2017;74(1):126-142.
43. Jomo JN, de Prenter F, Elhaddad M, et al. Robust and parallel scalable iterative solutions for large-scale finite cell analyses. *Finite Element Anal Des*. 2019;163:14-30.
44. de Prenter F, Verhoosel CV, van Brummelen EH. Preconditioning immersed isogeometric finite element methods with application to flow problems. *Comput Methods Appl Mech Eng*. 2019;348:604-631.
45. de Prenter F, Verhoosel CV, van Brummelen EH, et al. Multigrid solvers for immersed finite element methods and immersed isogeometric analysis. *Comput Mech*. 2019b;65:1-32.
46. Saberi S, Vogel A, Meschke G. Parallel finite cell method with adaptive geometric multigrid. *European Conference on Parallel Processing*. Springer; 2020:578-593.
47. Burstedde C, Wilcox LC, Ghattas O. p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM J Sci Comput*. 2011;33(3):1103-1133.
48. Hughes TJR, Franca LP, Balestra M. A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuška-Brezzi condition: A stable Petrov-Galerkin formulation of the stokes problem accommodating equal-order interpolations. *Comput Methods Appl Mech Eng*. 1986;59(1):85-99.
49. Brooks AN, Hughes TJR. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Comput Methods Appl Mech Eng*. 1982;32(1-3):199-259.
50. Leopoldo PF, Sérgio LF. Stabilized finite element methods: II. the incompressible Navier-Stokes equations. *Comput Methods Appl Mech Eng*. 1999 (2-3): 209–233, 1992.
51. Knobloch P, Tobiska L. Stabilization methods of bubble type for the $q_1/q_1$-element applied to the incompressible Navier-Stokes equations. *ESAIM Math Model Numer Anal*. 2000;34(1):85-107.
52. Dohrmann CR, Bochev PB. A stabilized finite element method for the Stokes problem based on polynomial pressure projections. *Int J Numer Methods Fluids*. 2004;46(2):183-201.

53. Bochev PB, Dohrmann CR, Gunzburger MD. Stabilization of low-order mixed finite elements for the Stokes equations. *SIAM J Numer Anal*. 2006;44(1):82-101.

54. Li J, He Y. A stabilized finite element method based on two local gauss integrations for the Stokes equations. *J Comput Appl Math*. 2008;214(1):58-65.

55. Kelley CT. *Iterative methods for linear and nonlinear equations*. SIAM; 1995.

56. Sampath RS, Biros G. A parallel geometric multigrid method for finite elements on octree meshes. *SIAM J Sci Comput*. 2010;32(3):1361-1392.

57. Balay S, Gropp WD, McInnes LC, Smith BF. Efficient management of parallelism in object oriented numerical software libraries. In: Arge E, Bruaset AM, Langtangen HP, eds. *Modern Software Tools in Scientific Computing*. Birkhäuser Press; 1997:163-202.

58. Ahrens J, Geveci B, Law C. Paraview: An end-user tool for large data visualization. *The Visualization Handbook*, Vol 717. Elsevier; 2005.

How to cite this article: Saberi S, Meschke G, Vogel A. Adaptive geometric multigrid for the mixed finite cell formulation of Stokes and Navier–Stokes equations. *Int J Numer Meth Fluids*. 2023;95(7):1035-1053. doi:10.1002/fld.5180