# A SCALABLE AUXILIARY SPACE PRECONDITIONER FOR HIGH-ORDER FINITE ELEMENT METHODS

YOUNG-JU LEE, WEI LENG, AND CHEN-SONG ZHANG

ABSTRACT. In this paper, we revisit an auxiliary space preconditioning method proposed by Xu [Computing 56, 1996], in which low-order finite element spaces are employed as auxiliary spaces for solving linear algebraic systems arising from high-order finite element discretizations. We provide a new convergence rate estimate and parallel implementation of the proposed algorithm. We show that this method is user-friendly and can play an important role in a variety of Poisson-based solvers for more challenging problems such as the Navier–Stokes equation. We investigate the performance of the proposed algorithm using the Poisson equation and the Stokes equation on 3D unstructured grids. Numerical results demonstrate the advantages of the proposed algorithm in terms of efficiency, robustness, and parallel scalability.

## 1. INTRODUCTION

Iterative methods have been successfully applied to large-scale sparse linear systems arising from discretizations of partial differential equations (PDEs). Many linear systems of equations can be handled by preconditioned Krylov subspace methods [4, 40]. In fact, preconditioners play a crucial role in the convergence of iterative methods. The construction of an "ideal" preconditioner depends on following basic, sometimes contradictory, guidelines: (1) Optimality and Robustness: The convergence rate of an appropriate iterative methods on the preconditioned system is uniform or nearly uniform, independent of mesh size or physical parameters; (2) Cost-effectiveness and Scalability: The computational costs and memory requirements of constructing and applying the preconditioning action are low and have good parallel scalability; and (3) User-friendliness: Users require little information and implementation is not difficult.

The Poisson equation $-\Delta u = f$ and its variants arise in many applications. The geometric multigrid (GMG) method is one of the most efficient iterative methods for solving discrete Poisson or Poisson-like equations. A vast number of works have explored multigrid methods; references include the monographs and the survey papers [11, 25, 9, 48]. Though the classical multigrid algorithm based on a geometric hierarchy can be an effective solver for a well-structured, it is usually very difficult to obtain such a hierarchy in practice. The algebraic multigrid (AMG) method [15,

12, 46, 48, 13, 19], on the other hand, requires minimal geometric information about the underlying problem and can sometimes be employed as a "black-box" iterative solver or preconditioner for other iterative methods. The version known as the Classical AMG [15, 37] is used frequently and has been shown to be effective for a range of problems in practice. In an effort to render AMG methodologies more broadly applicable and to improve robustness, various versions have been developed; for example, see [49, 55, 35, 16].

AMG methods are readily applicable and potentially scalable for large 3D problems. Recently, parallel versions of multigrid methods have attracted a lot of attention (and will continue to do so) because of their fundamental role in modern computational mathematics and engineering; see [24, 20, 44, 5] and references therein for details. In this paper, we will not discuss parallelization and implementation of AMG. Throughout this paper, we employ the Parallel Modified Independent Set (PMIS) coarsening strategy [45] and the Extend+i+cc interpolation [44] in Boomer-AMG of *hypre* package [1], which has been numerically proven to be efficient and scalable [5].

Although AMG methods have been proven effective for many problems, it is important to note that generally the performance of the Classical AMG method deteriorates for high-order finite elements (see Shu, Sun and Xu [43] for the 2D test examples). In Table 1, we show a simple numerical experiment. It is easy to see that for about the same degree of freedom ($10^7$) the convergence rate of the AMG method (PMIS and Extend+i+cc) deteriorates. Furthermore, the performance of AMG is very sensitive to the strength threshold $\theta$ in the coarsening procedure (see §6 for details). On the other hand, it is clear that AMG can be also very effective for the discrete Poisson equations in relatively low order finite element spaces.

TABLE 1. Number of iterations for the AMG preconditioned GMRES method. We solve the 3D Poisson equation with 64 processing cores (piecewise continuous Lagrangian finite element discretizations are applied, the stopping criterion is when the relative residual is less than $10^{-6}$, and DOF is the total degree of freedom.)

| Element Type | DOF | $\theta = 0.25$ | $\theta = 0.5$ | $\theta = 0.7$ | $\theta = 0.8$ | $\theta = 0.9$ |
|:---:|:---:|:---:|:---:|:---:|:---:|:---:|
| $P^{1,0}$ | 13M | 5 | 5 | 8 | 10 | 13 |
| $P^{2,0}$ | 13M | 6 | 7 | 10 | 13 | 17 |
| $P^{3,0}$ | 13M | 8 | 10 | 12 | 15 | 18 |
| $P^{4,0}$ | 12M | >500 | >500 | 17 | 18 | 22 |

Studies have proposed using a two-level approach to handle Poisson equations on the higher-order finite element spaces. Such an approach would consist of (1) a smoother for the Poisson equation on the higher order finite element spaces, (2) transfer operators between the higher-order finite element spaces and the lower-order finite element spaces, and (3) an AMG method applied to the Poisson equations defined on the lower-order finite element space. Shu, Sun and Xu [43] has designed an algebraic multigrid method by constructing lower order finite element coefficient matrices algebraically with the help of characteristics of Lagrangian finite

element spaces. Their study is restricted to the quadratic and cubic Lagrangian finite element discretizations in 2D. Another attempt to use low-order finite element space for preconditioning can be found in Heys, et al. [26].

The solution technique to discrete Poisson equations is itself of great interest. However, even more compelling are Poisson-based solution techniques that can be applied to constructing an efficient solver for more complicated problems [53]. Over the last few decades, intensive research has been devoted to developing efficient linear solvers for almost all kinds of sparse linear systems in scientific and engineering computing. The main idea of efficient preconditioning is to transform a seemingly intractable problem to a (sequence of) problem(s) that can be approximated rapidly. One such mathematical technique is a general framework called Auxiliary Space Preconditioning or ASP [53, 54]. This method represents a large class of preconditioners that (1) by using auxiliary spaces transform a complicated system into a sequence of simpler systems, and (2) construct efficient preconditioners with efficient solvers for these simpler systems. Based on fast Poisson solvers and analytic insight into PDEs or PDE systems, efficient solvers can be developed using the auxiliary space preconditioning framework for various cases that arise in practical computations. Successful examples include simple and complex fluid problems, linear elasticity, and $H(grad)$, $H(div)$, and $H(curl)$ systems with applications to the Maxwell equations [27, 54, 33].

In this paper, we revisit the algorithm proposed in [53] for solving a large-scale discrete second-order elliptic equations by high-order finite element methods. Moreover, using easily available mesh information, we provide a parallel implementation of this auxiliary space preconditioner and analyze its performance for problems with about half a billion unknowns in terms of the robustness, efficiency, and scalability. This paper makes an additional contribution by providing an alternative proof for the convergence rate of the proposed algorithms. Lastly, the proposed method will be applied to solving the 3D Stokes equation on unstructured meshes. It is noteworthy that this proposed preconditioner is user-friendly and can improve the robustness, efficiency, and scalability of the solution to the Stokes equation compared with pure AMG methods. This indicates that the proposed method can also make a useful building block for other Poisson-based solvers.

Throughout this paper, we will use the following notation. The symbol $L_0^2$ denotes the space of all square integrable functions, $L^2$, whose entries have zero mean values. Let $H^k$ be the standard Sobolev space of the scalar function whose weak derivatives up to order $k$ are square integrable, and, let $\|\cdot\|_k$ and $|\cdot|_k$ denote the standard Sobolev norm and its corresponding seminorm on $H^k$, respectively. Furthermore, $\|\cdot\|_{k,\omega}$ and $|\cdot|_{k,\omega}$ denote the norm $\|\cdot\|_k$ and the semi-norm $|\cdot|_k$ restricted to the domain $\omega \subset \Omega$, respectively. We use the notation $X \lesssim (\gtrsim)Y$ to denote the existence of a generic constant $C$, which depends only on $\Omega$, such that $X \leq (\geq)CY$.

The rest of the paper is organized as follows. In §2, using the auxiliary space preconditioning framework, we present the construction of the geometric hierarchy and a two-level method for the Poisson equation from high-order finite element discretizations. In §3 and §4, we analyze the convergence of the proposed two-level algorithm by casting it into the augmented matrix formulation by Griebel [23]. In §5, we discuss the preconditioning techniques for saddle point problems from the

mixed finite element for the Stokes equation. In §6, a number of numerical experiments are reported and summarized to demonstrate the efficiency and robustness of our parallel implementation.

## 2. A geometric–algebraic multigrid algorithm

This section is devoted to present the algebraic multigrid methods for the Poisson equations discretized by the higher order finite element methods with geometric hierarchy between higher order finite elements and the lower order finite element spaces consisting of piecewise linear elements.

Let $\Omega \subset \mathbb{R}^d$ be a bounded polyhedral domain. We consider the Poisson equation

$$(1) \qquad\qquad -\Delta u = f \qquad \text{in } \Omega,$$

subject to zero Dirichlet boundary condition $u = 0$ on $\partial\Omega$. We then consider the following weak formulation of (1): Find $u \in V = H_0^1(\Omega)$, such that

$$(2) \qquad\qquad a(u, v) = \langle f, v \rangle \qquad \forall\, v \in V,$$

where

$$(3) \qquad a(u, v) := \int_\Omega \nabla u \cdot \nabla v \, dx \quad \text{and} \quad \langle f, v \rangle := \int_\Omega f \, v \, dx \qquad \forall\, u, v \in V.$$

We now discretize the equation (2) using the finite element method. To introduce the finite element spaces. Assume that $\mathcal{T}_h$ is a shape-regular triangular (tetrahedral) mesh of $\Omega$. For any $T \in \mathcal{T}_h$, let $P^k(T)$ be the set of polynomials on $T$ of degree less than or equal to $k$. We denote the piecewise continuous $P^k$ Lagrangian finite element space as $V_h := P^{k,0}$. In this paper, $V_h$ denotes a finite element space consisting of the $k$-th ($k \geq 2$) order piecewise continuous polynomials, such as quadratic, cubic or quartic polynomials. That is to say

$$(4) \qquad V_h := \{v \in C(\Omega) : v|_T \in P^k(T), \ \forall\, T \in \mathcal{T}_h\} = \text{span}\{\phi_1, \dots, \phi_{n_h}\},$$

where $n_h$ is the total number of degrees of freedom and $\{\phi_i\}_{i=1,\dots,n_h}$ are the standard $k$-th order Lagrange basis functions. The discrete weak formulation of (2) can be written as

$$(5) \qquad\qquad a(u_h, v_h) = \langle f, v_h \rangle \qquad \forall\, v_h \in V_h.$$

We introduce an auxiliary space, the continuous piecewise linear polynomial space,

$$(6) \qquad V_H := \{v \in C(\Omega) : v|_T \in P^1(T), \ \forall\, T \in \mathcal{T}_h\} = \text{span}\{\psi_1, \dots, \psi_{n_H}\},$$

where $\{\psi_j\}_{j=1,\dots,n_H}$ are the canonical basis functions or the hat functions. We denote, by $\{x_i^h\}_{i=1,\dots,n_h}$ and $\{x_i^H\}_{i=1,\dots,n_H}$, the set of evenly-spaced nodes where the degree of freedom (DOF) for the Lagrange finite spaces $V_h$ and $V_H$ are defined, respectively. Figure 1 shows the local ordering of $x_i^h$ on a single simplex in 3D.

Throughout this paper, we use the following convention for the vector representation of a function in $V_h$: For any $v_h = \sum_{i=1}^{n_h} v_i \phi_i \in V_h$ and $w_H = \sum_{i=1}^{n_H} w_i \psi_i \in V_H$, we denote by $\mathbf{v}_h$ and $\mathbf{w}_H$, the vector representation of $v_h$ and $w_h$, respectively. Namely, $\mathbf{v}_h = (v_1, \dots, v_{n_h})^T$ and $\mathbf{w}_H = (w_1, \dots, w_{n_H})^T$. Similarly, the symbol $\mathbf{f}_h$ denotes $(f_1, \dots, f_{n_h})^T$ with $f_i = \langle f, \phi_i \rangle$ for all $i = 1, \dots, n_h$. The equation (5) can then be cast into the following equivalent finite-dimensional linear system of equations: Find $\mathbf{u}_h$ such that
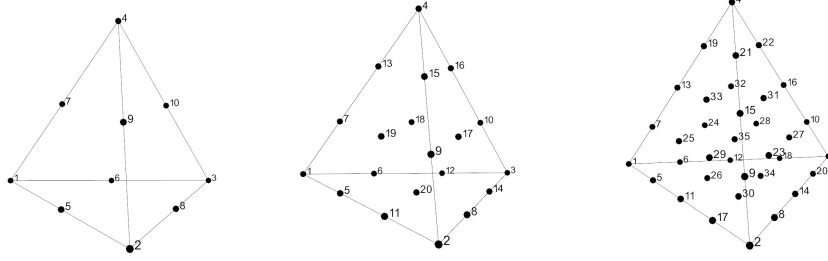
$$(7) \qquad\qquad \mathsf{A}_h \mathbf{u}_h = \mathbf{f}_h,$$

FIGURE 1. Local numbering of nodes $x_i^h$'s in a single tetrahedron used for $P2$, $P3$ and $P4$ FEMs

where $\mathsf{A}_h$ is a symmetric positive definite matrix with $(\mathsf{A}_h)_{ij} = a(\phi_j, \phi_i)$.

We are now at the position to present a geometry-based algebraic multigrid (GAMG) method by using the auxiliary linear finite element space. For this purpose, we start by introducing the transfer operators between $V_h$ and $V_H$. Since $k \geq 2$, any basis function $\psi_j \in V_H$ can be represented by the basis functions $\{\phi_i\}_{1 \leq i \leq n_h}$ of the space $V_h$. Namely, for any $j = 1, \ldots n_H$, there exists $\mathbf{c}_j = (c_j^1, \ldots, c_j^{n_h})^T \in \mathbb{R}^{n_h}$ such that

$$(8) \qquad \psi_j = \sum_{i=1}^{n_h} c_j^i \phi_i.$$

These coefficients $c_j^i$'s can be easily obtained by $c_j^i = \psi_j(x_i^h)$ for any $i = 1, \ldots, n_h$ and $j = 1, \ldots, n_H$. Hence, we have that

$$(9) \qquad \psi_j = \sum_{i \in N_h^H(j)} \psi_j(x_i^h)\phi_i,$$

where $N_h^H(j) = \{i \in \{1, \ldots, n_h\} : \mathrm{supp}(\phi_i) \cap \mathrm{supp}(\psi_j) \neq \emptyset\}$. We define the transfer operators

$$(10) \qquad \mathsf{I}_\mathsf{P} = (\mathbf{c}_1, \ldots, \mathbf{c}_{n_H}) \in \mathbb{R}^{n_h \times n_H} \quad \text{and} \quad \mathsf{I}_\mathsf{R} = \mathsf{I}_\mathsf{P}^T.$$

We note that the transfer operators $\mathsf{I}_\mathsf{R}$ and $\mathsf{I}_\mathsf{P}$ can be generated easily with the mesh $\mathcal{T}_h$ available. An alternate (algebraic) approach for generating them has been introduced by Shu, Sun, and Xu [43].

Now, let $\mathsf{A}_h$ and $\mathsf{A}_H$ be the stiffness matrices defined by (5) on the finite element spaces $V_h$ and $V_H$, respectively. Then we have the Galerkin relation $\mathsf{A}_H = \mathsf{I}_\mathsf{R}\mathsf{A}_h\mathsf{I}_\mathsf{P}$. Let $\mathsf{G}_h$ be a smoother and we can state the following two-level algorithm:

**Algorithm 2.1** (A Two-level Method). *Given an initial iterate, $\mathbf{u}_h^0$ on the fine grid, we perform the following steps until convergence for $\ell = 0, 1, \ldots$*
   **Step 1.** *Solve the coarse grid equation*

$$\mathsf{A}_H \mathbf{w}_H = \mathsf{I}_\mathsf{R}(\mathbf{f}_h - \mathsf{A}_h \mathbf{u}_h^\ell)$$

   **Step 2.** *Correction*

$$\widetilde{\mathbf{u}}_h = \mathbf{u}_h^\ell + \mathsf{I}_\mathsf{P} \mathbf{w}_H$$

**Step 3.** *Postsmoothing*

$$\mathbf{u}_h^{\ell+1} = \widetilde{\mathbf{u}}_h + \mathsf{G}_h(\mathbf{f}_h - \mathsf{A}_h\widetilde{\mathbf{u}}_h)$$

**Remark 2.1** (The Two-Level Method and GAMG). In Step 1 of Algorithm 2.1, we can employ an AMG method or a few AMG cycles to solve the coarse-level problem approximately. We can also add a presmoothing step in front of Step 1 to make the method symmetric. This particular version of Algorithm 2.1 will then be referred to as the geometric-algebraic multigrid (GAMG) method as easily available geometric information (mesh) is used in our implementation of Step 3.

## 3. Augmented matrix formulation of the two-level method

In this section, we introduce an equivalent form of Algorithm 2.1 in terms of the augmented algebraic systems based on so-called the redundant representation of the solution space. This observation was originally made by Griebel [23], however, the convergence rate estimate in this framework has not been seen in literature. The main idea lies in the following redundant representation of the functions in the space $V_h$: For any $v_h \in V_h$, we have the representation

$$(11) \qquad v_h = \sum_{i=1}^{n_H} \bar{v}_i \psi_i + \sum_{i=1}^{n_h} \widehat{v}_i \phi_i.$$

We notice that the above representation is not unique since the basis functions used in the representation are not independent, which is why (11) is also called the redundant representation of functions in $V_h$. Based on (11), we can consider the discrete weak formulation: Let $\widetilde{V}_h = \mathrm{span}\{\psi_1, \ldots, \psi_{n_H}, \phi_1, \ldots, \phi_{n_h}\}$ and find $v_h = \sum_{i=1}^{n_h} \widehat{v}_i \phi_i + \sum_{i=1}^{n_H} \bar{v}_i \psi_i \in \widetilde{V}_h$, such that

$$(12) \qquad a(v_h, w_h) = \langle f, w_h \rangle \qquad \forall\, w_h \in \widetilde{V}_h.$$

Let $\widehat{v} = (\widehat{v}_1, \ldots, \widehat{v}_{n_h})^T$ and $\overline{v} = (\bar{v}_1, \ldots, \bar{v}_{n_H})^T$. It is then easy to establish that the resulting system of equations from the aforementioned weak formulation (12) leads to the following augmented matrix systems [23]:

**Lemma 3.1.** *Let $\mathsf{I_P}$ and $\mathsf{I_R}$ be the prolongation and restriction given in (10), respectively. Then the problem (5) can be written as the following matrix equation*

$$(13) \qquad \mathcal{A}\begin{pmatrix} \overline{v} \\ \widehat{v} \end{pmatrix} = \widetilde{f} \qquad i.e. \quad \begin{pmatrix} \mathsf{I_R}\mathsf{A}_h\mathsf{I_P} & \mathsf{I_R}\mathsf{A}_h \\ \mathsf{A}_h\mathsf{I_P} & \mathsf{A}_h \end{pmatrix}\begin{pmatrix} \overline{v} \\ \widehat{v} \end{pmatrix} = \begin{pmatrix} \mathsf{I_R}\,\mathbf{f}_h \\ \mathbf{f}_h \end{pmatrix}.$$

*Proof.* From the following relation that

$$(14) \qquad \psi_i = \sum_{j=1}^{n_h} c_i^j \phi_j, \qquad i = 1, \ldots, n_H,$$

we can deduce that

$$(15) \qquad a(\psi_i, \psi_j) = (\mathsf{I_R}\mathsf{A}_h\mathsf{I_P})_{ij}$$

and

$$(16) \qquad a(\psi_i, \phi_j) = a\Big(\sum_{k=1}^{n_h} c_i^k \phi_k, \phi_j\Big) = \sum_{k=1}^{n_h} c_i^k a(\phi_k, \phi_j) = (\mathsf{I_R}\mathsf{A}_h)_{ij}.$$

We also note that

$$\langle f, \psi_i \rangle = \Big\langle f, \sum_{k=1}^{n_h} c_j^k \phi_k \Big\rangle = \sum_{k=1}^{n_h} c_j^k \langle f, \phi_k \rangle = \mathsf{I_R f}_h.$$

This completes the proof. $\qquad\square$

There are a few interesting properties of the augmented matrix $\mathcal{A}$ given in (13) which will be useful in later sections. First of all, the matrix $\mathcal{A}$ is singular with positive diagonal entries. Moreover, the range and the null space of $\mathcal{A}$ denoted by $\mathcal{R}(\mathcal{A}) = \mathcal{N}^\perp$ and $\mathcal{N} = \mathcal{N}(\mathcal{A})$, respectively, can be characterized as follows:

$$\mathcal{R}(\mathcal{A}) = \left\{ \begin{pmatrix} \mathsf{I_R v}_h \\ \mathbf{v}_h \end{pmatrix} : v_h \in V_h \right\} \quad \text{and} \quad \mathcal{N}(\mathcal{A}) = \left\{ \begin{pmatrix} \mathbf{c}_H \\ -\mathsf{I_P c}_H \end{pmatrix} : c_H \in V_H \right\}.$$

Let $\widetilde{\mathbf{e}}_k$ for $k = 1, \ldots, n_H + n_h$ be the canonical basis for $\mathbb{R}^{n_H + n_h}$. We denote the solution space of the augmented matrix system (13) as

$$(17) \qquad \mathcal{V} := \left\{ \widetilde{v} = \begin{pmatrix} \overline{v} \\ \widehat{v} \end{pmatrix} : v_h = \sum_{i=1}^{n_H} \overline{v}^i \psi_i + \sum_{i=1}^{n_h} \widehat{v}^i \phi_i, \quad \forall\, v_h \in \widetilde{V}_h \right\} \subseteq \mathbb{R}^{n_H + n_h}.$$

It is worthy to note that Algorithm 2.1[1] can analyzed in the framework of Successive Subspace Corrections (SSC) [52] with the subspace decomposition

$$\mathcal{V} = \mathcal{V}_0 + \mathcal{V}_1 + \cdots + \mathcal{V}_{n_h},$$

where $\mathcal{V}_0 = \mathrm{span}\,\{\widetilde{\mathbf{e}}_1, \ldots, \widetilde{\mathbf{e}}_{n_H}\}$ and $\mathcal{V}_j = \mathrm{span}\{\widetilde{\mathbf{e}}_{n_H + j}\}$ for $j = 1, \ldots, n_h$. In this setting, the SSC method can be written as follows:

**Algorithm 3.1** (Successive Subspace Correction Method). *Let $\widetilde{u}^0 \in \mathcal{V}$ be given.*
$\quad$ **for** $\ell = 1, 2, \ldots$
$\qquad \widetilde{u}_0^{\ell-1} = \widetilde{u}^{\ell-1}$
$\qquad$ **for** $k = 0, 1, 2, \ldots, n_h$
$\qquad\qquad$ *Find* $\widetilde{w}_k \in \mathcal{V}_k : \quad (\mathcal{A}\widetilde{w}_k, \widetilde{v}_k) = \left( \widetilde{f}, \widetilde{v}_k \right) - \left( \mathcal{A}\widetilde{u}_{k-1}^{\ell-1}, \widetilde{v}_k \right) \quad \forall \widetilde{v}_k \in \mathcal{V}_k$
$\qquad\qquad \widetilde{u}_k^{\ell-1} = \widetilde{u}_{k-1}^{\ell-1} + \widetilde{w}_k$
$\qquad$ **endfor**
$\qquad \widetilde{u}^\ell = \widetilde{u}_{n_h}^{\ell-1}$
$\quad$ **endfor**

The error transfer operator of the above algorithm can be identified as

$$(18) \qquad\qquad \mathcal{E} = (\mathcal{I} - \mathcal{P}_{n_h})(\mathcal{I} - \mathcal{P}_{n_h - 1}) \cdots (\mathcal{I} - \mathcal{P}_0),$$

where $\mathcal{I} : \mathcal{V} \mapsto \mathcal{V}$ is the identity matrix and $\mathcal{P}_j : \mathcal{V} \mapsto \mathcal{V}_j$ for $j = 0, \ldots, n_h$ is the $\mathcal{A}$-projection onto the space $\mathcal{V}_j$. More precisely, $\mathcal{P}_0 : \mathcal{V} \mapsto \mathcal{V}_H$ can be defined as

$$\mathcal{P}_0 \widetilde{v} = \begin{pmatrix} \overline{v} + (\mathsf{I_R A}_h \mathsf{I_P})^{-1} \mathsf{I_R A}_h \widehat{v} \\ 0 \end{pmatrix} = \begin{pmatrix} \overline{v} + \mathsf{P}_0 \widehat{v} \\ 0 \end{pmatrix} \qquad \forall \widetilde{v} = \begin{pmatrix} \overline{v} \\ \widehat{v} \end{pmatrix} \in \mathcal{V},$$

where $\mathsf{P}_0 = (\mathsf{I_R A}_h \mathsf{I_P})^{-1} \mathsf{I_R A}_h$. For $j = 1, \ldots, n_h$, we define the projections

$$\mathcal{P}_j \widetilde{v} := \frac{(\mathcal{A}\widetilde{\mathbf{e}}_{n_H + j}, \widetilde{v})}{(\mathcal{A}\widetilde{\mathbf{e}}_{n_H + j}, \widetilde{\mathbf{e}}_{n_H + j})} \widetilde{\mathbf{e}}_j = \frac{(\mathsf{A}_h \mathbf{e}_j, (\mathsf{I_P}\overline{v} + \widehat{v}))}{(\mathsf{A}_h \mathbf{e}_j, \mathbf{e}_j)} \widetilde{\mathbf{e}}_{n_H + j} = (\mathsf{P}_j(\mathsf{I_P}\overline{v} + \widehat{v}), \mathbf{e}_j) \widetilde{\mathbf{e}}_{n_H + j},$$

---

[1]We can also analyze the Algorithm 2.1 with presmoothing by modifying the space decomposition slightly to make it symmetric.

where $\mathbf{e}_j$ for $j = 1, \ldots, n_h$ is the canonical basis for $\mathbb{R}^{n_h}$ and $\mathsf{P}_j$ is $\mathsf{A}_h$-projection onto the space $\mathrm{span}\{\mathbf{e}_j\}$ given by

$$\mathsf{P}_j \widehat{v} = \frac{(\mathsf{A}_h \mathbf{e}_j, \widehat{v})}{(\mathsf{A}_h \mathbf{e}_j, \mathbf{e}_j)} \mathbf{e}_j \qquad \forall \widehat{v} \in \mathbb{R}^{n_h}.$$

**Remark 3.1** (Algorithm 2.1 and Block Gauss-Seidel Method)**.** We apply the space decomposition

$$(19) \qquad \qquad \mathcal{V} = \mathcal{V}_H + \mathcal{V}_h,$$

where

$$\mathcal{V}_H = \mathrm{span}\{\widetilde{\mathbf{e}}_1, \ldots, \widetilde{\mathbf{e}}_{n_H}\} \quad \text{and} \quad \mathcal{V}_h = \mathrm{span}\{\widetilde{\mathbf{e}}_{n_H+1}, \ldots, \widetilde{\mathbf{e}}_{n_H+n_h}\}.$$

We decompose of the matrix $\mathsf{A}_h$ in the following form

$$(20) \qquad \qquad \mathsf{A}_h = \mathsf{D} - \mathsf{L} - \mathsf{L}^T,$$

where $\mathsf{D} = (a_{ii})_{i=1,\ldots,n_h}$ is the diagonal part of $\mathsf{A}_h$ and $\mathsf{L} = (\ell_{ij})_{i,j=1,\ldots,n_h}$ with $\ell_{ij} = 0$ for $i > j$ and $\ell_{ij} = -a(\phi_j, \phi_i)$ for $i < j$, i.e., the strictly lower triangular part of $\mathsf{A}_h$. Similarly we can decompose $\mathcal{A}$ as follows

$$(21) \qquad \qquad \mathcal{A} = \mathcal{D} - \mathcal{L} - \mathcal{L}^T,$$

where

$$\mathcal{D} := \begin{pmatrix} \mathsf{I}_\mathsf{R} \mathsf{A}_h \mathsf{I}_\mathsf{P} & 0 \\ 0 & \mathsf{D} \end{pmatrix} \quad \text{and} \quad \mathcal{L} := \begin{pmatrix} 0 & 0 \\ -\mathsf{A}_h \mathsf{I}_\mathsf{P} & \mathsf{L} \end{pmatrix}.$$

In fact, we can easily show that the two-level method Algorithm 2.1 is equivalent to the Gauss-Siedel method for the augmented system of equations (13):

$$(22) \qquad \qquad \widetilde{v}^{\ell+1} = \widetilde{v}^\ell + (\mathcal{D} - \mathcal{L})^{-1} (\widetilde{f} - \mathcal{A}\widetilde{v}^\ell) \qquad \ell = 0, 1, \ldots$$

## 4. Convergence rate estimate for the two-level method

In this section, we establish a convergence rate estimate for Algorithm 2.1 using the formulation introduced in the previous section. We denote a semi inner product $(\cdot, \cdot)_\mathcal{A} : \mathcal{V} \times \mathcal{V} \mapsto \mathbb{R}$ and the induced semi-norm by $|\cdot|_\mathcal{A} = (\cdot, \cdot)_\mathcal{A}^{1/2}$. We now establish a convergence rate identity for the error transfer operator (18). Note that relevant estimates have been reported in [30, 32], but we provide a proof for completeness.

**Theorem 4.1** (Convergence Rate Identity)**.** *The convergence rate for the iterative method (22) can be given by*

$$(23) \qquad \qquad |\mathcal{E}|_\mathcal{A}^2 = 1 - \frac{1}{K},$$

*where*

$$K = 1 + \sup_{\widetilde{u} \in \mathcal{N}^\perp} \inf_{\widetilde{c} \in \mathcal{N}} \frac{(\mathcal{S}(\widetilde{u} + \widetilde{c}), (\widetilde{u} + \widetilde{c}))}{(\widetilde{u}, \widetilde{u})_\mathcal{A}} = \sup_{\widetilde{u} \in \mathcal{N}^\perp} \inf_{\widetilde{c} \in \mathcal{N}} \frac{\sum_{i=0}^{n_h} \left| \mathcal{P}_i \left( \sum_{j=i}^{n_h} \widetilde{u}_j \right) \right|_\mathcal{A}^2}{(\widetilde{u}, \widetilde{u})_\mathcal{A}},$$

*where $\mathcal{S} = \mathcal{L}\mathcal{D}^{-1}\mathcal{L}^T$ and $\widetilde{u}_j \in \mathcal{V}_j$ for $j = 0, \ldots, n_h$.*

*Proof.* Let $\mathcal{B} := (\mathcal{D} - \mathcal{L})^{-1}$. From Remark 3.1, we have $\mathcal{E} = \mathcal{I} - \mathcal{B}\mathcal{A}$. By the definition of the semi-norm $|\mathcal{E}|_{\mathcal{A}}$, we have the following identity

$$
\begin{aligned}
|\mathcal{E}|_{\mathcal{A}}^2 &= \sup_{\widetilde{u} \in \mathcal{N}^{\perp}} \frac{(\mathcal{E}\widetilde{u}, \mathcal{E}\widetilde{u})_{\mathcal{A}}}{(\widetilde{u}, \widetilde{u})_{\mathcal{A}}} = \sup_{\widetilde{u} \in \mathcal{N}^{\perp}} \frac{((\mathcal{I} - \mathcal{B}\mathcal{A})\widetilde{u}, (\mathcal{I} - \mathcal{B}\mathcal{A})\widetilde{u})_{\mathcal{A}}}{(\widetilde{u}, \widetilde{u})_{\mathcal{A}}} \\
&= \sup_{\widetilde{u} \in \mathcal{N}^{\perp}} \frac{((\mathcal{I} - \mathcal{B}\mathcal{A})^*(\mathcal{I} - \mathcal{B}\mathcal{A})\widetilde{u}, \widetilde{u})_{\mathcal{A}}}{(\widetilde{u}, \widetilde{u})_{\mathcal{A}}},
\end{aligned}
$$

where $(\mathcal{I} - \mathcal{B}\mathcal{A})^*$ is the adjoint operator of $\mathcal{I} - \mathcal{B}\mathcal{A}$ with respect to the semi inner product $(\cdot, \cdot)_{\mathcal{A}}$.

We notice that $(\mathcal{I} - \mathcal{B}\mathcal{A})^* = \mathcal{I} - \mathcal{B}^T\mathcal{A}$, where $\mathcal{B}^T$ is the adjoint of $\mathcal{B}$ with respect to the usual $\ell^2$ inner product. Furthermore, since

$$
\begin{aligned}
(\mathcal{I} - \mathcal{B}\mathcal{A})^*(\mathcal{I} - \mathcal{B}\mathcal{A}) &= \mathcal{I} - \mathcal{B}^T(\mathcal{B}^{-T} + \mathcal{B}^{-1} - \mathcal{A})\mathcal{B}\mathcal{A} \\
&= \mathcal{I} - (\mathcal{B}^{-1}\mathcal{D}^{-1}\mathcal{B}^{-T})^{-1}\mathcal{A} = \mathcal{I} - (\mathcal{A} + \mathcal{S})^{-1}\mathcal{A},
\end{aligned}
$$

we then have that

$$
|\mathcal{E}|_{\mathcal{A}}^2 = 1 - \inf_{\widetilde{u} \in \mathcal{N}^{\perp}} \frac{((\mathcal{A} + \mathcal{S})^{-1}\mathcal{A}\widetilde{u}, \widetilde{u})_{\mathcal{A}}}{(\widetilde{u}, \widetilde{u})_{\mathcal{A}}}.
$$

We shall now define $\mathcal{M} = \mathcal{A}^{1/2}(\mathcal{A} + \mathcal{S})^{-1}\mathcal{A}\mathcal{A}^{1/2}$ and obtain the following identity :

$$
\begin{aligned}
K &= \sup_{\widetilde{u} \in \mathcal{N}^{\perp}} \frac{(\widetilde{u}, \widetilde{u})_{\mathcal{A}}}{((\mathcal{A} + \mathcal{S})^{-1}\mathcal{A}\widetilde{u}, \widetilde{u})_{\mathcal{A}}} = \sup_{\widetilde{u} \in \mathcal{N}^{\perp}} \frac{(\widetilde{u}, \widetilde{u})_{\mathcal{A}}}{(\mathcal{A}^{-1/2}\mathcal{M}\mathcal{A}^{-1/2}\widetilde{u}, \widetilde{u})_{\mathcal{A}}} \\
&= \sup_{\widetilde{u} \in \mathcal{N}^{\perp}} \frac{(\mathcal{A}^{-1/2}\mathcal{M}\mathcal{A}^{-1/2}\widetilde{u}, \widetilde{u})_{\mathcal{A}}}{(\mathcal{A}^{-1/2}\mathcal{M}\mathcal{A}^{-1/2}\widetilde{u}, \mathcal{A}^{-1/2}\mathcal{M}\mathcal{A}^{-1/2}\widetilde{u})_{\mathcal{A}}}.
\end{aligned}
$$

The last equality is from the fact that $\mathcal{M} : \mathcal{N}^{\perp} \mapsto \mathcal{N}^{\perp}$ is symmetric and positive definite matrix and by replacing $\widetilde{u}$ by $\mathcal{M}^{1/2}\widetilde{u}$. Let $Q : \mathcal{V} \mapsto \mathcal{N}^{\perp}$ be the $\ell^2$-orthogonal projection, for which

$$
\mathcal{A}Q\widetilde{v} = \mathcal{A}\widetilde{v} \qquad \forall \widetilde{v} \in \mathcal{V}. \tag{24}
$$

We now write $(\mathcal{A} + \mathcal{S})^{-1}\mathcal{A}\widetilde{u} = \widetilde{w} + \widetilde{c}(\widetilde{w})$, where $\widetilde{w} := Q(\mathcal{A} + \mathcal{S})^{-1}\mathcal{A}\widetilde{u} \in \mathcal{N}^{\perp}$ and $\widetilde{c}(\widetilde{w}) \in \mathcal{N}$. Note that $\widetilde{c}(\widetilde{w})$ is uniquely determined by $\widetilde{w}$. Therefore, due to the fact that

$$
\mathcal{A}^{-1/2}\mathcal{M}\mathcal{A}^{-1/2}\widetilde{u} = (\mathcal{A} + \mathcal{S})^{-1}\mathcal{A}\widetilde{u} = \widetilde{w} + \widetilde{c}(\widetilde{w}) \quad \text{and} \quad (\mathcal{A} + \mathcal{S})(\widetilde{w} + \widetilde{c}(\widetilde{w})) = \mathcal{A}\widetilde{u},
$$

we immediately obtain that

$$
K = \sup_{\widetilde{w} \in \mathcal{N}^{\perp}} \frac{((\widetilde{w} + \widetilde{c}(\widetilde{w})), (\mathcal{A} + \mathcal{S})(\widetilde{w} + \widetilde{c}(\widetilde{w})))}{(\widetilde{w}, \widetilde{w})_{\mathcal{A}}} = \sup_{\widetilde{w} \in \mathcal{N}^{\perp}} \inf_{\widetilde{c} \in \mathcal{N}} \frac{((\mathcal{A} + \mathcal{S})(\widetilde{w} + \widetilde{c}), (\widetilde{w} + \widetilde{c}))}{(\widetilde{w}, \widetilde{w})_{\mathcal{A}}}.
$$

The last equality is obtained by the following reasoning: For a fixed $\widetilde{w} \in \mathcal{N}^{\perp}$, we assume that

$$
\widetilde{\xi} = \arg \inf_{\widetilde{c} \in \mathcal{N}} \frac{((\mathcal{A} + \mathcal{S})(\widetilde{w} + \widetilde{c}), (\widetilde{w} + \widetilde{c}))}{(\widetilde{w}, \widetilde{w})_{\mathcal{A}}}. \tag{25}
$$

Then the first order optimality condition implies that $\xi$ satisfies

$$
\left((\mathcal{A} + \mathcal{S})(\widetilde{w} + \widetilde{\xi}), \widetilde{c}\right) = 0 \qquad \forall \widetilde{c} \in \mathcal{N}. \tag{26}
$$

This in turn implies that $(\mathcal{A} + \mathcal{S})(\widetilde{w} + \widetilde{\xi}) \in \mathcal{N}^{\perp}$ and $\widetilde{\xi} = \widetilde{c}(\widetilde{w})$. This completes the proof. $\qquad\square$

To establish a uniform convergence rate of the proposed two-level method (Algorithm 2.1), we show that $K$ in Theorem 4.1 can be bounded by a generic constant independent of mesh size. The standard prolongation $I_h^H : V_H \mapsto V_h$ (the inclusion operator) and the restriction operator $I_H^h : V_h \mapsto V_H$ can be written as

$$(27) \qquad I_h^H w_H = (\phi_1, \ldots, \phi_{n_h})\mathsf{I}_\mathsf{P}\mathbf{w}_H \quad \text{and} \quad I_H^h v_h = (\psi_1, \ldots, \psi_{n_H})\mathsf{I}_\mathsf{R}\mathbf{v}_h.$$

We introduce two additional restriction operators: the usual $L^2$ projection $Q_H : V_h \mapsto V_H$ and the elliptic projection $P_H : V_h \mapsto V_H$ defined by

$$(Q_H v, w_H) = (v, w_H) \text{ and } a(P_H v, w_H) = a(v, w_H) \quad \forall v \in V_h,\ w_H \in V_H,$$

respectively. It is clear that $(\mathsf{A}_h \mathsf{I}_\mathsf{P}\mathbf{v}_H, \mathsf{I}_\mathsf{P}\mathbf{v}_H) = \|v_H\|_1^2$ and the vector representation of $P_H v$ is simply $\mathsf{A}_H^{-1}\mathsf{I}_\mathsf{R}\mathsf{A}_h\mathbf{v}$. Therefore, we have that, for any $v \in V_h$, the following inequality holds:

$$(28) \qquad (\mathsf{A}_h\mathsf{I}_\mathsf{P}\mathsf{A}_H^{-1}\mathsf{I}_\mathsf{R}\mathsf{A}_h\mathbf{v}, \mathsf{I}_\mathsf{P}\mathsf{A}_H^{-1}\mathsf{I}_\mathsf{R}\mathsf{A}_h\mathbf{v}) = \|P_H v\|_1^2 \leq \|v\|_1^2.$$

We are now ready to prove the main theoretical result in this paper. Note that there are abundant literatures regarding the uniform convergence of two grid methods. Unlike those reported in literatures such as [43], our convergence estimate uses a different and novel technique, which is based on a convergence rate estimate for a singular system from the multilevel and redundant decomposition of the solution space.

**Theorem 4.2** (Uniform Convergence). *The following estimate holds true*

$$
\begin{aligned}
K &= 1 + \sup_{\widetilde{u}\in\mathcal{N}^\perp} \inf_{\widetilde{c}\in\mathcal{N}} \frac{(\mathcal{S}(\widetilde{u}+\widetilde{c}),(\widetilde{u}+\widetilde{c}))}{(\widetilde{u},\widetilde{u})_\mathcal{A}} \\
&= \sup_{\widetilde{u}\in\mathcal{N}^\perp} \inf_{\widetilde{c}\in\mathcal{N}} \frac{\sum_{i=0}^{n_h}\left|\mathcal{P}_i\left(\sum_{j=i}^{n_h}\widetilde{u}_j\right)\right|_\mathcal{A}^2}{(\widetilde{u},\widetilde{u})_\mathcal{A}} \lesssim \sup_{v_h\in V_h} \frac{\|v_h - Q_H v_h\|_1}{\|v_h\|_1} + 1 \lesssim 1,
\end{aligned}
$$

*where $\widetilde{u}_j \in \mathcal{V}_j$ for $j = 0, 1, \ldots, n_h$ such that $\sum_{j=0}^{n_h} \widetilde{u}_j = \widetilde{u} + \widetilde{c}$.*

*Proof.* Let $\widetilde{u} \in \mathcal{N}^\perp$ and $\widetilde{c} \in \mathcal{N}$ be given by

$$(29) \qquad \widetilde{u} = \begin{pmatrix} \mathsf{I}_\mathsf{R}\mathbf{v}_h \\ \mathbf{v}_h \end{pmatrix} \text{ and } \widetilde{c} = \begin{pmatrix} \mathbf{v}_H \\ -\mathsf{I}_\mathsf{P}\mathbf{v}_H \end{pmatrix}, \quad \text{where } \mathbf{v}_h \in \mathbb{R}^{n_h},\ \mathbf{v}_H \in \mathbb{R}^{n_H}.$$

From Theorem 4.1, we have the identity

$$K = \sup_{\widetilde{u}\in\mathcal{N}^\perp} \inf_{\widetilde{c}\in\mathcal{N}} \frac{\sum_{i=0}^{n_h}\left|\mathcal{P}_i\left(\sum_{j=i}^{n_h}\widetilde{u}_j\right)\right|_\mathcal{A}^2}{(\widetilde{u},\widetilde{u})_\mathcal{A}}.$$

We note that the following identity holds true

$$(30) \qquad (\widetilde{u},\widetilde{u})_\mathcal{A} = \|(\mathsf{I}+\mathsf{I}_\mathsf{P}\mathsf{I}_\mathsf{R})\mathbf{v}_h\|_{\mathsf{A}_h}^2 = \|v_h + I_H^h v_h\|_1^2,$$

where $\mathbf{v}_h$ is the vector representation of $v_h \in V_h$ in terms of the basis functions $\{\phi_i\}_{i=1}^{n_h}$. Therefore, by choosing $\mathbf{v}_H$ as the vector representation of $v_H = Q_H v_h$, we obtain the following relation

$$\sum_{j=0}^{n_h} \widetilde{u}_j = \begin{pmatrix} \mathsf{I}_\mathsf{R}\mathbf{v}_h + \mathbf{v}_H \\ \mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H \end{pmatrix} = \begin{pmatrix} \mathsf{I}_\mathsf{R}\mathbf{v}_h + \mathbf{v}_H \\ \mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H \end{pmatrix}$$

and, in turn,

$$\sum_{i=0}^{n_h}\left|\mathcal{P}_i\left(\sum_{j=i}^{n_h}\widetilde{u}_j\right)\right|_{\mathcal{A}}^2 = \sum_{i=1}^{n_h}\left|\mathcal{P}_i\left(\sum_{j=i}^{n_h}\widetilde{u}_j\right)\right|_{\mathcal{A}}^2 + \left|\mathcal{P}_0\left(\begin{array}{c}\mathsf{I}_\mathsf{R}\mathbf{v}_h + \mathbf{v}_H \\ \mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H\end{array}\right)\right|_{\mathcal{A}}^2$$

$$= \sum_{i=1}^{n_h}\left\|\mathsf{P}_i\left(\sum_{j=i}^{n_h}\widehat{u}_j\mathbf{e}_j\right)\right\|_{\mathsf{A}_h}^2 + \|\mathsf{I}_\mathsf{R}\mathbf{v}_h + \mathbf{v}_H + \mathsf{P}_0\left(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H\right)\|_{\mathsf{A}_H}^2,$$

where $(\widehat{u}_1,\ldots,\widehat{u}_{n_h})^T = \mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H$. Next, we estimate the two parts in the above identity separately.

It is straight-forward to observe that

$$\|\mathsf{P}_0(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H)\|_{\mathsf{A}_H}^2 = (\mathsf{P}_0(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H),\mathsf{P}_0(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H))_{\mathsf{A}_H}$$
$$= (\mathsf{I}_\mathsf{R}\mathsf{A}_h(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H),\mathsf{A}_H^{-1}\mathsf{I}_\mathsf{R}\mathsf{A}_h(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H))$$
$$= (\mathsf{A}_h(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H),\mathsf{I}_\mathsf{P}\mathsf{A}_H^{-1}\mathsf{I}_\mathsf{R}\mathsf{A}_h(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H))$$
$$\leq \|\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H\|_{\mathsf{A}_h}^2, \qquad \text{due to the inequality (28).}$$

Therefore, we obtain that

$$\|\mathsf{I}_\mathsf{R}\mathbf{v}_h + \mathbf{v}_H + \mathsf{P}_0\left(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H\right)\|_{\mathsf{A}_H}^2 \lesssim \|\mathsf{I}_\mathsf{P}\mathsf{I}_\mathsf{R}\mathbf{v}_h + \mathsf{I}_\mathsf{P}\mathbf{v}_H\|_{\mathsf{A}_h}^2 + \|\mathsf{P}_0\left(\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H\right)\|_{\mathsf{A}_H}^2$$
$$\lesssim \|\mathsf{I}_\mathsf{P}\mathsf{I}_\mathsf{R}\mathbf{v}_h + \mathbf{v}_h\|_{\mathsf{A}_h}^2 + \|\mathbf{v}_h - \mathsf{I}_\mathsf{P}\mathbf{v}_H\|_{\mathsf{A}_h}^2$$
$$\text{(31)} \qquad = \|I_H^h v_h + v_h\|_1^2 + \|v_h - Q_H v_h\|_1^2.$$

Let $\Omega_i = \text{supp}\,\phi_i$ and $\mathsf{A}_h = (a_{ij})$ with $a_{ij} = a(\phi_j,\phi_i)$. Using the Cauchy-Schwarz inequality, we have the following estimate

$$\sum_{i=1}^{n_h}\left\|\mathsf{P}_i\left(\sum_{j=i}^{n_h}\widehat{u}_j\right)\right\|_{\mathsf{A}_h}^2 = \sum_{i=1}^{n_h}\left((\mathbf{e}_i^T\mathsf{A}_h\mathbf{e}_i)^{-1}\mathbf{e}_i^T\mathsf{A}_h\left(\sum_{j=i}^{n_h}\widehat{u}_j\right)\mathbf{e}_i^T\mathsf{A}_h\left(\sum_{j=i}^{n_h}\widehat{u}_j\right)\right)$$

$$= \sum_{i=1}^{n_h}a(\phi_i,\phi_i)^{-1}a\left(\phi_i,\sum_{j=i}^{n_h}\widehat{u}_j\phi_j\right)^2$$

$$\leq \sum_{i=1}^{n_h}\int_{\Omega_i}\left|\nabla\sum_{j=i}^{n_h}\widehat{u}_j\phi_j\right|^2 dx \lesssim \sum_{i=1}^{n_h}\sum_{j\in N_k(i)}h^{d-2}\widehat{u}_j^2,$$

where $d$ is the dimension and $N_k(i) = \{j \in \{1,\ldots,n_h\} : \Omega_j \cap \Omega_i \neq \emptyset\}$. Norm equivalence leads to the following estimate that

$$\sum_{j\in N_k(i)}h^d\widehat{u}_j^2 \lesssim \|v_h - Q_H v_h\|_{0,\Omega_i}^2.$$

Therefore, we arrive at the conclusion that

$$\sum_{i=1}^{n_h} \left\| \mathsf{P}_i \left( \sum_{j=i}^{n_h} \widehat{u}_j \right) \right\|_{\mathsf{A}_h}^2 \lesssim h^{-2} \sum_{i=1}^{n_h} \|v_h - Q_H v_h\|_{0,\Omega_i}^2 \lesssim h^{-2} \|v_h - Q_H v_h\|_0^2$$

$$= h^{-2} \|(v_h + I_H^h v_h) - Q_H(v_h + I_H^h v_h)\|_0^2$$

(32) $$\lesssim \|(I - Q_H)(v_h + I_H^h v_h)\|_1^2.$$

By combining (31) and (32), we obtain the desired estimate for $K$, which completes the proof. $\square$

## 5. BLOCK PRECONDITIONERS FOR THE STOKES EQUATION

In this section, we consider efficient Poisson-based preconditioning techniques for the Stokes system with no-slip boundary condition: Find velocity $\mathbf{u}$ and pressure $p$, such that

(33) $$\begin{cases} -\Delta \mathbf{u} - \nabla p &= \mathbf{f} & \text{in } \Omega \\ \nabla \cdot \mathbf{u} &= 0 & \text{in } \Omega \\ \mathbf{u} &= \mathbf{0} & \text{on } \partial\Omega, \end{cases}$$

where $\Omega \subset \mathbb{R}^d$ ($d = 2, 3$) is a bounded polygonal domain and $\mathbf{f}$ is a given function.

There have been extensive discussions on discretizations of the Stokes equation. In this paper, we shall focus on the mixed finite element discretizations; see, for example, [47, 22, 17]. When solving this problem with mixed finite element methods (MFEM), a pair of discrete function spaces for velocity and pressure must be chosen carefully so that it is stable, i.e., satisfying so-called the inf-sup condition. There are a number of important classes of stable pairs. In particular, a family of Hood–Taylor finite elements [28] that approximates the velocity by continuous piecewise $k$-th order polynomials ($P^{k,0}$) and the pressure by continuous piecewise $(k-1)$-th order polynomials ($P^{k-1,0}$) with $k \geq 2$. is known to be stable for full three dimensional Stokes equation [7]. Another important stable elements shown only in 2D, is the Scott–Vogelius elements [42] with $k \geq 4$, which approximates the velocity by continuous piecewise $k$-th order polynomials ($P^{k,0}$) and the pressure by discontinuous piecewise $(k-1)$-th order polynomials ($P^{k-1,-1}$). These pairs of mixed finite elements are very promising because they preserve the incompressibility condition, namely, discrete divergence free condition in the strong sense.

Assume that the coefficient matrix arising in mixed finite element discretizations of (33) can be written as

$$F := \begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix}$$

Here, $B$ is the discrete divergence operator, i.e., $-\nabla_h\cdot$; and, $A$ is the block diagonal matrix with the discrete Laplace matrix $-\Delta_h$ on its diagonal. Let $x_{\mathbf{u}}$ and $x_p$ be the unknown vectors of the velocity field and pressure, respectively. Then we need to solve the system of linear equations

(34) $$Fx = b \qquad \text{or} \qquad F \begin{bmatrix} x_{\mathbf{u}} \\ x_p \end{bmatrix} = \begin{bmatrix} b \end{bmatrix}.$$

$F$ is symmetric and positive semidefinite and we can apply Krylov subspace methods for indefinite problems such as the minimal residual (MINRES) method [36]

and the generalized minimal residual (GMRES) method [41] to solve (34). It is well-known that the convergence rate of such methods is governed by three parameters: the condition numbers of $A$, $B$ and the relative scaling between them. Generally, $\text{cond}(A) = O(h^{-2})$, therefore, a direct application of these methods will yield a slow convergence and the main difficulty in solving the Stokes equations lies in constructing "good" preconditioners for the elliptic operator $A$, which is typically given by the discrete elliptic operator with higher order finite elements.

A lot of efforts have been devoted on solving the saddle point problems arising from mixed finite element methods for the Stokes equation; see [18, 6, 29] and references therein for details. A few efficient multigrid-type solution methods have been proposed for the Stokes equation [14, 51, 50, 8]. In this paper, we focus on the Poisson-based block diagonal and block triangular preconditioners [10, 38]. Recently, parallel version of these preconditioning algorithms attracted a lot of interests due to their efficiency and easiness for implementation; see for example [34, 21].

Denote the Schur complement by $S := BA^{-1}B^T$. The block triangular factorizations of $F$,

$$
\begin{pmatrix} A & B^T \\ B & 0 \end{pmatrix} = \begin{pmatrix} I_{\mathbf{u}} & 0 \\ BA^{-1} & I_p \end{pmatrix} \begin{pmatrix} A & B^T \\ 0 & -S \end{pmatrix}
$$

$$
(35) \qquad = \begin{pmatrix} I_{\mathbf{u}} & 0 \\ BA^{-1} & I_p \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & -S \end{pmatrix} \begin{pmatrix} I_{\mathbf{u}} & A^{-1}B^T \\ 0 & I_p \end{pmatrix},
$$

motivate a block upper triangular preconditioner [10]

$$
(36) \qquad Q_t = \begin{pmatrix} A & B^T \\ 0 & -S \end{pmatrix}^{-1}
$$

and an even simpler block diagonal preconditioner [38]

$$
(37) \qquad Q_d = \begin{pmatrix} A & 0 \\ 0 & -S \end{pmatrix}^{-1}.
$$

In either $Q_t$ or $Q_d$, it requires to obtain certain approximations to $A^{-1}$ and $S^{-1}$. Since $A$ is a discretization of the Laplace operator, we form an approximation to $A^{-1}$ by applying one multilevel V-cycle to $A$. As for the Schur complement part, we approximate $S$ with the pressure mass matrix $M_p$ and solve $M_p$ equation using the conjugate gradient method with diagonal preconditioner. We note that there are different ways to form the preconditioning actions based on (36) and (37); see [6].

## 6. Numerical experiments

In this section, we test the performance of the proposed Geometric-Algebraic Multigrid (GAMG) method (Algorithm 2.1 with presmoothing; see Remark 2.1) and compare it with the corresponding AMG method with focus on their robustness, efficiency, and parallel weak scalability. For this purpose, we use two simple test problems—one is the 3D Poisson equation and the other is the 3D Stokes equation—on a unit cube with unstructured tetrahedral meshes. We pay special attention to the performance of both methods for higher-order finite element discretizations.

6.1. **Implementation.** All numerical tests are carried out on the LSSC-Ⅲ cluster at the State Key Laboratory of Scientific and Engineering Computing (LSEC), Chinese Academy of Sciences. The LSSC-Ⅲ cluster has 282 computing nodes: Each node has two Intel Quad Core Xeon X5550 2.66GHz processors and 24GB shared memory; all nodes are connected via Gigabit Ethernet and DDR InfiniBand. To make a fair comparison, we use zero right-hand side and start from a random initial guess in our tests. In this section, "#It" denotes the number of iterations, "DOF" denotes the degree of freedom, "CPU" denotes the computation wall time in seconds, and "RAM" denotes the memory usage per processor in MB.

Our implementation is based on several open-source numerical packages. The finite element discretization for the Poisson equation and the Stokes equation is implemented using PHG [3, 56]. PHG is a toolbox for developing parallel adaptive finite element programs on unstructured tetrahedral meshes and it is under active development at LSEC. PHG is also employed to build the two-level GAMG setting, namely to generate the transfer operators (prolongation and restriction).

The solvers are implemented using PETSc [2] and BoomerAMG in *hypre* [1]. Due to limited space, we only report the results for the Flexible GMRES (FGM-RES) method [39, 40] in PETSc and BoomerAMG with the PMIS method [45] for coarsening, the Extended+i+cc method [44] for interpolation, and the hybrid Gauss-Seidel method for smoothing. This particular setting provides good efficiency and scalability for the linear systems in our numerical tests. For more numerical tests for various choices of iterative methods and different types of AMG methods, we refer to Lee, Leng and Zhang [31].

**Remark 6.1** (Number of Smoothing Sweeps)**.** In multigrid method, there is a trade-off in number of smoothing sweeps used in each cycle. More smoothing sweeps in one cycle will cost more computation time in each multigrid cycle but may reduce total number of cycles. Another consideration about the number of smoothing sweeps is that for linear system that is harder to solve, more smoothing sweeps might lead to better convergence rate. In this paper, we use only one pre and post smoothing sweep in our experiments.

6.2. **Test Problem 1—the Poisson equation.** The GAMG solver for the Poisson equation is implemented as follows: We first pass the linear systems to the FGMRES iterative method of PETSc, then we use one multilevel V-cycle as the preconditioner for FGMRES. In each multilevel cycle, the coarse level problem (corresponding to the $P^{1,0}$ finite element space) is solved with BoomerAMG in *hypre*. The AMG preconditioner for the Poisson problem is simple, we pass the linear system to the FGMRES method of PETSc and employ BoomerAMG as a preconditioner. In both methods, the stopping criteria is that the relative residual is less than $10^{-6}$.

In Table 1 in §1, we have shown that the AMG method could be very sensitive to the strength threshold $\theta$ when applied to the linear systems arising in higher order finite element discretizations. The strength threshold or strong threshold determines strength of connections, i.e., a point (variable) $i$ strongly depends on $j$ if

$$-a_{i,j} > \theta \max_{k \neq i}(-a_{i,k}).$$

The default value of $\theta$ in BoomerAMG is 0.25, which usually works well for 2D Laplace operators and a larger value, like 0.5, is suggested for 3D Laplace operators.

But neither of them works for the $P^{4,0}$ finite element discretization. Hence we start by testing both AMG and GAMG methods for various values of $\theta$ and report CPU time and memory consumption in Table 2, 3, 4.

We notice that: (1) The AMG method under consideration performs reasonably well even for high-order finite element methods. (2) However, it is very sensitive to the strength threshold $\theta$ for the fourth order finite element method. (3) The GAMG method, on the other hand, is very robust with respect to $\theta$. And, in general, it converges faster (from 1.5 times up to 30 times, see Table 5) and consumes less memory (by 10% to 50%) compared with the AMG method. (4) For large 3D linear systems arising from higher-order finite element discretizations, large strength threshold often works much better. In the rest of the comparisons, we will fix the parameter $\theta = 0.8$, which is the best choice for AMG, but not necessarily for GAMG.

TABLE 2. Iteration number, CPU time, and memory usage of the AMG and GAMG preconditioned Krylov subspace method for the 3D Poisson equation on unstructured tetrahedral mesh ($P^{2,0}$ finite element, about 430K DOF per processing core).

| Method ($\theta$) | 1 Core | | | 8 Cores | | | 64 Cores | | |
|---|---|---|---|---|---|---|---|---|---|
| | #It | CPU | RAM | #It | CPU | RAM | #It | CPU | RAM |
| AMG (0.2) | 7 | 11.75 | 1141 | 6 | 22.82 | 1424 | 6 | 32.71 | 1320 |
| GAMG (0.2) | 7 | 4.16 | 790 | 7 | 9.25 | 1170 | 7 | 8.57 | 1072 |
| AMG (0.4) | 7 | 8.00 | 1124 | 7 | 18.13 | 1428 | 7 | 25.50 | 1294 |
| GAMG (0.4) | 7 | 4.16 | 790 | 7 | 9.39 | 1170 | 7 | 8.79 | 1073 |
| AMG (0.6) | 8 | 5.92 | 1080 | 8 | 15.06 | 1381 | 8 | 20.29 | 1255 |
| GAMG (0.6) | 7 | 4.09 | 790 | 7 | 8.55 | 1174 | 7 | 8.19 | 1074 |
| AMG (0.8) | 11 | 5.10 | 1022 | 13 | 14.64 | 1280 | 13 | 17.85 | 1187 |
| GAMG (0.8) | 9 | 4.35 | 790 | 9 | 9.13 | 1168 | 9 | 9.49 | 1072 |

TABLE 3. Iteration number, CPU time, and memory usage of the AMG and GAMG preconditioned Krylov subspace method for the 3D Poisson equation on unstructured tetrahedral mesh ($P^{3,0}$ finite element, about 650K DOF per processing core).

| Method ($\theta$) | 1 Core | | | 8 Cores | | | 64 Cores | | |
|---|---|---|---|---|---|---|---|---|---|
| | #It | CPU | RAM | #It | CPU | RAM | #It | CPU | RAM |
| AMG (0.2) | 8 | 32.24 | 2345 | 8 | 42.98 | 2133 | 7 | 75.84 | 2419 |
| GAMG (0.2) | 12 | 13.14 | 1216 | 12 | 14.11 | 1410 | 11 | 19.46 | 1522 |
| AMG (0.4) | 9 | 23.94 | 2278 | 8 | 35.15 | 2219 | 8 | 65.24 | 2507 |
| GAMG (0.4) | 12 | 13.16 | 1217 | 12 | 14.94 | 1412 | 11 | 19.99 | 1523 |
| AMG (0.6) | 10 | 16.76 | 1929 | 10 | 27.02 | 2067 | 10 | 48.07 | 2291 |
| GAMG (0.6) | 12 | 13.12 | 1222 | 12 | 14.86 | 1412 | 11 | 20.21 | 1523 |
| AMG (0.8) | 13 | 13.74 | 1696 | 14 | 23.29 | 1831 | 15 | 38.54 | 2137 |
| GAMG (0.8) | 12 | 13.08 | 1216 | 12 | 14.86 | 1411 | 11 | 20.19 | 1525 |

TABLE 4. Iteration number, CPU time, and memory usage of the AMG and GAMG preconditioned Krylov subspace method for the 3D Poisson equation on unstructured tetrahedral mesh ($P^{4,0}$ finite element, about 460K DOF per processing core).

| Method ($\theta$) | 1 Core | | | 8 Cores | | | 64 Cores | | |
|---|---|---|---|---|---|---|---|---|---|
| | #It | CPU | RAM | #It | CPU | RAM | #It | CPU | RAM |
| AMG (0.2) | 19 | 42.47 | 1986 | 28 | 73.22 | 2043 | 93 | 438.8 | 2352 |
| GAMG (0.2) | 16 | 16.45 | 1023 | 18 | 20.03 | 1117 | 17 | 25.77 | 1534 |
| AMG (0.4) | 16 | 22.12 | 1560 | 36 | 57.40 | 1840 | >500 | >800 | 2258 |
| GAMG (0.4) | 16 | 16.41 | 1023 | 18 | 20.05 | 1117 | 17 | 25.10 | 1535 |
| AMG (0.6) | 17 | 16.44 | 1679 | 25 | 33.83 | 1683 | 245 | 356.2 | 2085 |
| GAMG (0.6) | 16 | 16.42 | 1023 | 18 | 19.07 | 1117 | 17 | 25.64 | 1539 |
| AMG (0.8) | 17 | 13.08 | 1319 | 19 | 22.79 | 1555 | 19 | 36.50 | 1971 |
| GAMG (0.8) | 16 | 16.47 | 1023 | 18 | 20.93 | 1118 | 17 | 25.23 | 1534 |

TABLE 5. Speedup of GAMG compared with AMG for solving the discrete Poisson equation with finite element methods on unstructured tetrahedral meshes (solved using 64 processing cores).

| Finite Element | DOF | $\theta = 0.2$ | $\theta = 0.4$ | $\theta = 0.6$ | $\theta = 0.8$ |
|---|---|---|---|---|---|
| $P^{2,0}$ | 27M | 3.8 | 2.9 | 2.5 | 1.9 |
| $P^{3,0}$ | 43M | 3.9 | 3.3 | 2.4 | 1.9 |
| $P^{4,0}$ | 31M | 17.0 | 32.0 | 13.9 | 1.5 |

Now, let $L$ to be the number of levels in multilevel hierarchy and level 1 is the finest level. The operator complexity, $C_{op} := \sum_{l=1}^{L} nnz(A_l) / nnz(A_1)$, is the ratio between the total number of nonzeros (nnz) of all levels and the number of nonzeros of the finest level. The operator complexity is an important indicator of expense of multigrid type methods, not only for the storage requirements of multilevel preconditioners, but also for computational complexity for applying them. In our experiments, we use the PMIS coarsening strategy and the Extended+i+cc interpolation method in both AMG and GAMG. As summarized in Table 6, we notice that GAMG action is much cheaper than AMG. In fact, for $P^{3,0}$ and $P^{4,0}$, the operator complexity of GAMG is close to 1.0. This is due to the coarse level space, $P^{1,0}$-finite element space, contains considerably less degree of freedom and gives much less number of nonzeros in the coefficient matrices. In Table 6, the $P^{1,0}$–DOF column gives the degree of freedom for the corresponding coarse level.

In the rest of this subsection, we consider weak scalability of the proposed GAMG method. The maximal number of processing cores is 1024 (on 128 nodes) and the maximal degree of freedom in our tests are about $5 \times 10^8$. We notice that both the AMG and GAMG preconditioned Krylov subspace methods yield good optimality and scalability; see Figures 2, 3, and 4. We note that this comparison was done with the "good" parameter ($\theta = 0.8$); otherwise, the performance of AMG will deteriorate quickly for $P^{4,0}$ finite element.

TABLE 6. Operator complexities of AMG and GAMG (PMIS and Extended+i+cc) in single core tests.

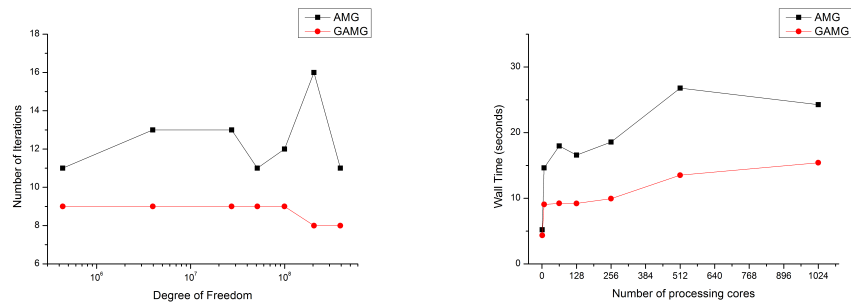| Finite Element | DOF | $P^{1,0}$–DOF | $C_{op}$ | | |
|---|---|---|---|---|---|
| | | | AMG | $P^{1,0}$–AMG | GAMG |
| $P^{1,0}$ | 429877 | — | 2.00 | — | — |
| $P^{2,0}$ | 435825 | 59495 | 1.67 | 1.76 | 1.12 |
| $P^{3,0}$ | 736608 | 31087 | 1.56 | 1.96 | 1.02 |
| $P^{4,0}$ | 460899 | 8165 | 1.36 | 1.70 | 1.01 |



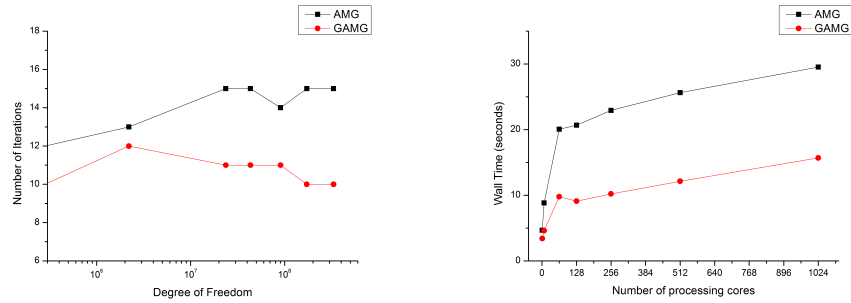FIGURE 2. Parallel (weak) scalability of AMG and GAMG for $P^{2,0}$ FEM for the Poisson equation.



FIGURE 3. Parallel (weak) scalability of AMG and GAMG for $P^{3,0}$ FEM for the Poisson equation.

6.3. **Test Problem 2—the Stokes equation.** In this section, we consider the Hood-Taylor family mixed finite element methods for the steady Stokes flow on a 3D lid driven cavity domain. Again, we test the AMG and GAMG methods with "good" strength threshold $\theta = 0.8$. We choose to stop the outer FGMRES iteration if the relative residual is smaller than $10^{-8}$.

Similar to the Poisson solver described in §6.2, the GAMG solver for Stokes test is implemented as follows: We pass the linear systems to the FGMRES solver in PETSc and we apply the block triangular preconditioner $Q_t$ described in §5 for the
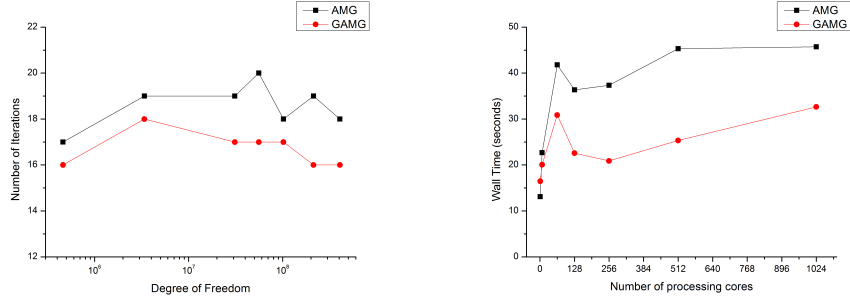
FIGURE 4. Parallel (weak) scalability of AMG and GAMG for $P^{4,0}$ FEM for the Poisson equation.

FGMRES method. The performance of block diagonal preconditioner $Q_d$ can be found in [31]. For the lower-right block (corresponding to the Schur complement), we solve it with the diagonal preconditioned PCG method in PETSc. For the upper-left blocks (corresponding to the Poisson equation), we approximate it with one GAMG V-cycle for the discrete Poisson equation. The AMG solver for Stokes test is similar except that the upper-left block was solved with one AMG V-cycle.

From Figure 5, 6, and 7, we immediately notice that: (1) In general, the AMG preconditioner performs reasonably well, even for high-order elements. (2) However, the convergence rate of AMG deteriorates with the size of problems and with the order of the mixed finite element. (3) The GAMG preconditioner yields much better convergence rate as well as scalability, especially for higher order elements. In particular, the iteration number does not increase as DOF increases.
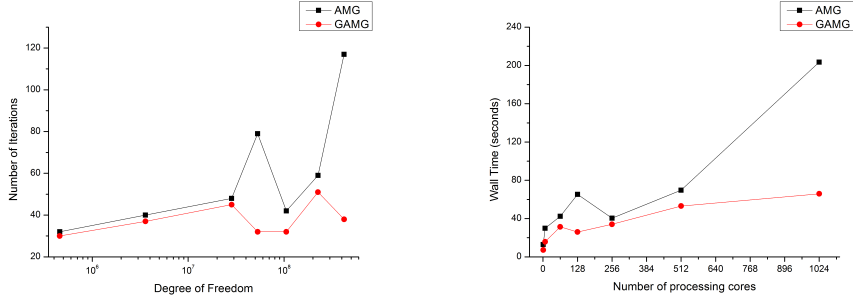


FIGURE 5. Algorithm optimality and parallel (weak) scalability of AMG and GAMG preconditioned FGMRES methods for the $P^{2,0} - P^{1,0}$ Hood-Taylor mixed finite element.

**Remark 6.2** (Intermediate Approximation Spaces)**.** In this paper, we only use two-level approximation with $P^{1,0}$ finite element space as the coarse level. One can imagine that intermediate (larger) auxiliary spaces could be used to improve performance. Since the convergence rate of the proposed two-level algorithm is
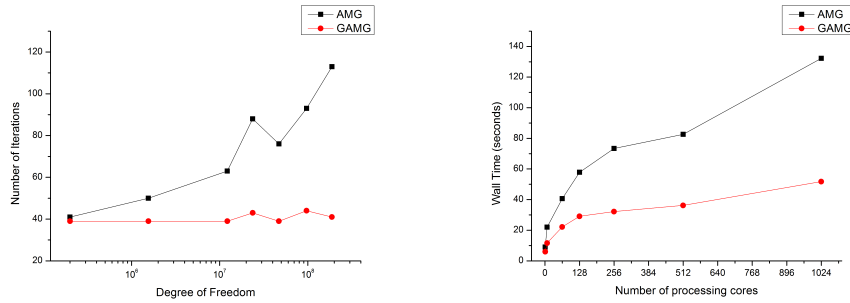
FIGURE 6. Algorithm optimality and parallel (weak) scalability of AMG and GAMG preconditioned FGMRES methods for the $P^{3,0} - P^{2,0}$ Hood-Taylor mixed finite element.
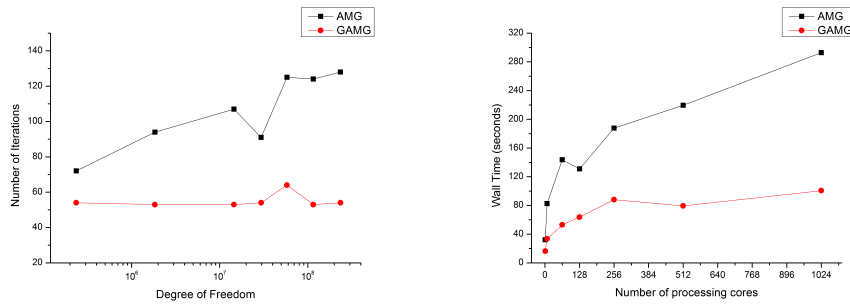


FIGURE 7. Algorithm optimality and parallel (weak) scalability of AMG and GAMG preconditioned FGMRES methods for the $P^{4,0} - P^{3,0}$ Hood-Taylor mixed finite element.

optimal (does not deteriorate as size of the problem increases) in our experiments, we decide not to do it in order to keep the implementation as simple as possible.

## 7. Conclusions

In this paper, we investigate an auxiliary space preconditioning method for high-order finite element discretizations of the Laplace operator in 3D. Modern parallel AMG techniques like PMIS/Extended+i+cc give good parallel scalability, but only if their parameters like strong strength threshold are chosen appropriately. On the contrary, the proposed auxiliary space preconditioner is very robust with respect to coarsening parameters, especially when applied as a building block of Poisson-based solvers for the Stokes equation in 3D. Furthermore, the proposed method yields smaller operator complexity, which leads to less memory usage and computational complexity.

## Acknowledgement

## References

[1] HYPRE (High Performance Preconditioners). http://www.llnl.gov/casc/linear solvers.
[2] PETSC (Portable, Extensible Toolkit for Scientic Computation). http://www-unix.mcs.anl.gov/petsc.
[3] PHG (Parallel Hierarchical Grid). http://lsec.cc.ac.cn/phg/.
[4] O. Axelsson. *Iterative solution methods*. Cambridge University Press, Cambridge, 1994.
[5] A. H. Baker, R. D. Falgout, T. V. Kolev, and U. M. Yang. Scaling hypre's multigrid solvers to 100000 cores. In M. B. et Al., editor, *High Performance Scientific Computing: Algorithms and Applications*, volume 27344.
[6] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numerica*, 14:1–137, May 2005.
[7] D. Boffi. Three-Dimensional Finite Element Methods for the Stokes problems. *SIAM J. Numer. Anal.*, 34:664–670, 1997.
[8] D. Braess and R. Sarazin. An efficient smoother for the Stokes problem. *Applied Numerical Mathematics*, 23(1):3–19, Feb. 1997.
[9] J. H. Bramble. *Multigrid Methods*, volume 294 of *Pitman Research Notes in Mathematical Sciences*. Longman Scientific & Technical, Essex, England, 1993.
[10] J. H. Bramble and J. E. Pasciak. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Math. Comp.*, 50(181):1–17, 1988.
[11] A. Brandt. *Multigrid techniques: 1984 guide with applications to fluid dynamics*. GMD–Studien Nr. 85. Gesellschaft für Mathematik und Datenverarbeitung, St. Augustin, 1984.
[12] A. Brandt. Algebraic multigrid theory: The symmetric case. *Appl. Mathematics of Computation.*, 19:23–56, 1986.
[13] A. Brandt. *Multiscale scientific computation: Review 2001*. T.J.Barth et.al eds. Springer. Springer, 2002.
[14] A. Brandt and N. Dinar. Multigrid Solutions to Elliptic Flow Prob- lems. In S. Parter, editor, *Numerical Methods for Partial Differential Equations*, pages 53—-147. 1979.
[15] A. Brandt, S. F. McCormick, and J. W. Ruge. Algebraic multigrid (AMG) for sparse matrix equations. In D. J. Evans, editor, *Sparsity and Its Applications*. Cambridge University Press, Cambridge, 1984.
[16] J. J. Brannick and R. D. Falgout. Compatible relaxation and coarsening in algebraic multigrid. *SIAM J. Sci. Comput.*, 32(3):1393–1416, 2010.
[17] F. Brezzi and M. Fortin. *Mixed and hybrid finite element methods*. Springer-Verlag, 1991.
[18] H. Elman, D. Silvester, and A. Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics*. Oxford University Press, USA, 2005.
[19] R. D. Falgout. An introduction to algebraic multigrid. *Computing in Science and Engineering*, 8:24–33, 2006.
[20] R. D. Falgout, J. E. Jones, and U. Meier Yang. Pursuing scalability for hypre's conceptual interfaces. *ACM Trans. Math. Software*, 31(3):326–350, 2005.
[21] T. Geenen, M. ur Rehman, S. P. MacLachlan, G. Segal, C. Vuik, A. P. van den Berg, and W. Spakman. Scalable robust solvers for unstructured FE geodynamic modeling applications: Solving the Stokes equation for models with large localized viscosity contrasts. *Geochemistry Geophysics Geosystems*, 10(9):1–12, Sept. 2009.
[22] V. Girault and P. A. Raviart. *Finite element methods for Navier–Stokes equations*. Springer-Verlag, Berlin, 1986. Theory and algorithms.
[23] M. Griebel. Multilevel algorithms considered as iterative methods on semidefinite systems. *SIAM Journal on Scientific and Statistical Computing*, 15:547–565, 1994.

[24] M. Griebel, B. Metsch, D. Oeltz, and M. A. Schweitzer. Coarse grid classification: A parallel coarsening scheme for algebraic multigrid methods. *Numerical Linear Algebra with Applications*, 13(2–3):193–214, 2006. Also available as SFB 611 preprint No. 225, Universität Bonn, 2005.

[25] W. Hackbusch. *Multigrid Methods and Applications*, volume 4 of *Computational Mathematics*. Springer–Verlag, Berlin, 1985.

[26] J. Heys, T. Manteuffel, S. McCormick, and L. Olson. Algebraic multigrid for higher-order finite elements. *Journal of Computational Physics*, 204(2):520 – 532, 2005.

[27] R. Hiptmair and J. Xu. Nodal auxiliary space preconditioning in H(curl) and H(div) spaces. *SIAM J. Numer. Anal.*, 45(6):2483–2509 (electronic), 2007.

[28] P. Hood and C. Taylor. A numerical solution of the Navier-Stokes equations using the finite element technique. *Internat. J. Comput. & Fluids*, 1:73–100, 1973.

[29] M. Larin and A. Reusken. A comparative study of efficient iterative solvers for generalized Stokes equations. *Numer. Linear Algebra Appl.*, 15(1):13–34, 2008.

[30] Y. Lee, J. Wu, J. Xu, and L. Zikatanov. A sharp convergence estimate of the method of subspace corrections for singular systems. *Mathematics of Computation*, 77 (262):831–850, 2008.

[31] Y.-J. Lee, W. Leng, and C.-S. Zhang. Numerical study of a parallel geometric-algebraic multigrid method for the Stokes equation. In preparation.

[32] Y.-J. Lee, J. Wu, J. Xu, and L. Zikatanov. Convergence analysis on iterative methods for semidefinite systems. *J. Comp. Math.*, 26:797–815, 2008.

[33] Y.-J. Lee, J. Xu, and C.-S. Zhang. Stable Finite Element Discretizations for Viscoelastic Flow Models. In *Handbook of Numerical Analysis*, volume XVI. 2011.

[34] D. A. May and L. Moresi. Preconditioned iterative methods for Stokes flow problems arising in computational geodynamics. *Physics of the Earth and Planetary Interiors*, 171(1-4):33–47, Dec. 2008.

[35] A. Muresan and Y. Notay. Analysis of aggregation-based multigrid. *SIAM Journal on Scientific Computing*, 30(2):1082–1103, 2008.

[36] C. C. Paige and M. A. Saunders. Solutions of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12(4):617–629, 1975.

[37] J. W. Ruge and K. Stüben. *Algebraic multigrid*, volume 3 of *Frontiers Appl. Math.*, pages 73–130. SIAM, Philadelphia, PA, 1987.

[38] T. Rusten and R. Winther. A preconditioned iterative method for saddle point problems. *SIAM J. Matrix Anal. Appl.*, 13(3):887–904, 1992.

[39] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Sci. Comput.*, 14:461–469, 1993.

[40] Y. Saad. *Iterative methods for sparse linear systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, second edition, 2003.

[41] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7:856–869, 1986.

[42] L. R. Scott and M. Vogelius. Conforming finite element methods for incompressible and nearly incompressible continua. In *Large-scale computations in fluid mechanics, Part 2 (La Jolla, Calif., 1983)*, volume 22 of *Lectures in Appl. Math.*, pages 221–244. Amer. Math. Soc., Providence, RI, 1985.

[43] S. Shu, D. Sun, and J. Xu. An algebraic multigrid method for higher order finite element discretizations. *Computing*, 77(4):347–377, 2006.

[44] H. D. Sterck, R. D. Falgout, J. W. Nolting, and U. M. Yang. Distance-two interpolation for parallel algebraic multigrid. *Numerical Linear Algebra with Applications*, 15(2-3):115–139, 2008.

[45] H. D. Sterck, U. M. Yang, and J. J. Heys. Reducing complexity in parallel algebraic multigrid preconditioners. *SIAM J. Matrix Anal. Appl.*, 27(4):1019–1039, 2006.

[46] K. Stüben. An introduction to algebraic multigrid. In U. Trottenberg, C. W. Oosterlee, and A. Schüller, editors, *Multigrid*, pages 413–532. Academic Press, London, 2000.

[47] R. Temam. *Navier-Stokes Equations*. North Holland, 1977.

[48] U. Trottenberg, C. W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press Inc., San Diego, CA, 2001. With contributions by A. Brandt, P. Oswald and K. Stüben.

[49] P. Vaněk, J. Mandel, and M. Brezina. Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, 56:179–196, 1996.

[50] S. P. Vanka. Block-implicit multigrid solution of Navier-Stokes equations in primitive variables. *J. Comput. Phys.*, 65(1):138–158, 1986.

[51] G. Wittum. Multi-Grid Methods for Stokes and Navier-Stokes Equations Transforming Smoothers: Algorithm and Numerical Results. *Numerische Mathematik*, 54:543–563, 1989.

[52] J. Xu. Iterative methods by space decomposition and subspace correction. *SIAM Review*, 34:581–613, 1992.

[53] J. Xu. The auxiliary space method and optimal multigrid preconditioning techniques for unstructured meshes. *Computing*, 56:215–235, 1996.

[54] J. Xu. Fast poisson-based solvers for linear and nonlinear pdes. In R. Bhatia, editor, *Proceedings of the international congress of mathematicians*, volume 4. World Sci. Publ., 2010.

[55] J. Xu and L. Zikatanov. On an energy minimazing basis in algebraic multigrid methods. *Computing and visualization in sciences*, 2004. submitted.

[56] L.-B. Zhang. A Parallel Algorithm for Adaptive Local Refinement of Tetrahedral Meshes Using Bisection. *Numer. Math. Theory Methods Appl.*, 2:65–89, 2009.