Journal of Computational Physics 406 (2020) 109165

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

# The image-based multiscale multigrid solver, preconditioner, and reduced order model

Dewen Yushu, Karel Matouš *

*Department of Aerospace and Mechanical Engineering, Center for Shock Wave-processing of Advanced Reactive Materials, University of Notre Dame, Notre Dame, IN 46556, USA*

ARTICLE INFO

*Article history:*
Received 13 December 2018
Received in revised form 9 October 2019
Accepted 28 November 2019
Available online 4 December 2019

*Keywords:*
Multigrid solver and preconditioner
Data-driven modeling
Reduced order model
Multiscale modeling
Level of detail
Image/Data compression

ABSTRACT

We present a novel image-based multiscale multigrid solver that can efficiently address the computational complexity associated with highly heterogeneous systems. This solver is developed based on an image-based, multiresolution model that enables reliable data flow between corresponding computational grids and provides large data compression. A set of inter-grid operators is constructed based on the microstructural data which remedies the issue of missing coarse grid information. Moreover, we develop an image-based multiscale preconditioner from the multiscale coarse images which does not traverse through any intermediate grid levels and thus leads to a faster solution process. Finally, an image-based reduced order model is designed by prolongating the coarse-scale solution to approximate the fine-scale one with improved accuracy. The numerical robustness and efficiency of this image-based computational framework is demonstrated on a two-dimensional example with high degrees of data heterogeneity and geometrical complexity.



## 1. Introduction

During the past decades, the development of high-performance computers [1,2] has enabled detailed numerical simulations in numerous scientific and engineering fields [3–5] with unprecedented resolution. However, direct numerical modeling (DNM) still requires large computing resources and novel mathematical approaches are needed to make a leap to next generation exascale platforms [6]. As one of the growing fields targeting this issue, the multiscale method has steadily gained popularity among other schemes [7–9]. Mutiscale methods have been utilized in a variety of simulations, such as fluid dynamics [10,11], material science [12,13], biological studies [14,15], chemical reactions [16,17], and others [18,19].

Multiscale modeling balances between capturing detailed features and reducing the underlying computational complexity caused by the large range of spatial scales [7,20]. Similar challenges exist in computer graphics during rendering of complex geometries while maintaining the real-time rendering rate [21,22]. A typical example is the image-rendering technique that generates progressively coarser objects of the fine image, which are called the levels of detail (LODs) [23]. The fundamental concept of LOD is to create a series of representations (i.e., a hierarchy of image details) with less detailed description for small, distant, or unimportant features. This type of technique successfully accommodates complex geometries while maintaining the real-time rendering rate [22,24]. Recently, an image-based mutiscale modeling technique was developed,

* Corresponding author.
*E-mail addresses:* dyushu@nd.edu (D. Yushu), kmatous@nd.edu (K. Matouš).
*URL:* http://www3.nd.edu/~kmatous/ (K. Matouš).

https://doi.org/10.1016/j.jcp.2019.109165
0021-9991/

which not only retains the visual effect, but also preserves microstructural characteristics, such as the first- and second-order probability functions [25].

Computational modeling relies on the conservation laws of mass, momentum, and energy. These balance laws are described mathematically by partial differential equations (PDE). Numerically solving a PDE involves numerical discretization [26] which often results in a system of algebraic equations. To solve the resulting system of equations, one can use either a direct method (e.g., Gaussian elimination [27], LU factorization [27]) or an iterative method (e.g., Jacobi [28], Gauss-Seidel [28], successive over-relaxation (SOR) [29], conjugate gradient (CG) [30], generalized minimal residual (GMRES) [31], multigrid [32,33]).

Among the iterative solvers, the multigrid has high algorithmic efficiency and offers the possibility of solving problems with $N$ unknowns using $\mathcal{O}(N)$ number of operations and storage [32–36]. The multigrid method was initially proposed in the 1960s and developed extensively in the 1970s [37–39]. Originally devised for elliptic boundary value problems, the multigrid method has become more versatile during the past few decades for handling problems with increased complexity. These problems include but are not limited to: solving other types of linear and nonlinear PDEs [40–46], eigenvalue problems [47–49], optimization problems [50,51], multilevel graph algorithms [52,53], optimal control and design [54,55]. Robust multigrid algorithms that exhibit outstanding convergence rate have been developed, which include the black-box method [56–59], the smoothed aggregation method [60–62], the auxiliary space method [63], etc.

Despite the significant improvements, the performance of multigrid methods remains highly problem-dependent [32]. For heterogeneous media where material properties, like elastic modulus, thermal conductivity, or hydraulic conductivity are fluctuating, the multigrid performance becomes fragile. To resolve this problem, work has been done to design a hierarchical basis [64–67]. With this approach, the multigrid convergence rate normally does not depend on the coefficient variations. However, the support of the modified basis function is prohibitively high. Another approach is to develop a matrix dependent mapping strategy, where the inter-grid operators are computed based on the pattern of the coefficient matrix. Improved performance for this type of algorithms has been shown in [56,58,68–72]. However, the performance is strongly correlated with the magnitude of coefficient contrast. Moreover, prolongation operators constructed on simple geometries [58,68] are not often robust enough for complex realistic systems [25,73]. A detailed review about this topic can be found in [74].

In addition to an iterative solver, the multigrid method is also recognized as an efficient preconditioner [75–78]. The multigrid preconditioner [79] has better scaling property with problem size in comparison to direct methods (e.g., incomplete LU [80] and incomplete Cholesky [81]). The multigrid preconditioned system leads to a more favorably clustered spectrum than other iterative methods (e.g., Jacobi [28] and SOR [29]), which greatly accelerates the Krylov subspace solvers [75,79]. Recent work has shown its outstanding performance for problems such as the visco-plastic Stokes problem [82], the Helmholtz equation [83], the bidomain equation [84], and others [85,86]. Meanwhile, the parallelization of the multigrid preconditioners for various high performance computational platforms is also gaining popularity [87,88]. In spite of this, a pure multigrid preconditioner often incurs high setup and storage costs which inhibits its wide usage [79]. This motivates the development of a variant, based on the multiscale preconditioning paradigm.

The multigrid method gives rise to auxiliary algebraic equations with reduced number of degrees of freedom (DOFs), which can be easily linked to the reduced order model (ROM). The overall goal of ROM is to reduce the computational cost of a high fidelity model and accommodate time-critical applications [89]. Advances have been made in ROM over the past decades and various approaches, such as the principal component analysis (PCA) [90,91], the diffusion maps [92,93], and the manifold-based methods [94,95], have been developed and successfully applied in various engineering and scientific fields. Using the multigrid method as a ROM is not new. Specifically, the auxiliary algebraic equations are often utilized for upscaling [96,97] and data homogenization [98]. However, the opposite process, i.e., to prolongate a high fidelity solution on the finest level using a smaller system that includes reduced data, is yet to be investigated.

In this paper, we present a novel multiscale multigrid solver, preconditioner and ROM based on a multiresolution image-driven model developed in our previous work [25]. First, we develop an image-based multiscale multigrid solver for systems of equations associated with highly heterogeneous coefficients. This solver innovatively makes use of the image-based multiscale scheme (i.e., LODs) [25], which removes the simplistic assumptions often made when formulating the inter-grid operators [56,57]. Second, we develop an image-based multiscale preconditioner which utilizes the geometrical coarse scale problem. This results in a much simpler formulation, saves the computational time, and reduces data storage on the intermediate levels. Third, we develop an image-based reduced order model (IROM), where we obtain a fine-scale approximation by upscaling the solution on the coarse level. This greatly reduces the operation counts and memory requirements. To demonstrate our approach, we create a multilevel image series of a 2D heterogeneous domain from real material data (i.e., discrete material micrographs) [25,73], and solve the corresponding 2D elliptic problem on the finest level. The performance of our image-based multiscale multigrid solver is close to the theoretical optimum and is robust at the high coefficient contrast. The image-based multiscale preconditioner greatly reduces the condition number, thus significantly accelerates the CG convergence. The IROM greatly reduces the number of DOFs and captures the solution characteristics at the same time.

The rest of this paper is organized as follows. In Section 2, we describe the image-based multiscale model. In Section 3, we derive the image-based multiscale multigrid solver. In Section 4, we discuss the image-based multiscale preconditioner. In Section 5, we describe the IROM. In Section 6, we present numerical results. Finally, we draw conclusions in Section 7.

## 2. Image-based multiscale model

To develop the efficient multiscale multigrid solver, we use an image-based (i.e., data-driven) paradigm. In particular, we utilize the sharp volumetric billboard (SVB) model originally proposed by Yushu et al., [25]. The SVB data compression method is built from experimental microstructures. It exhibits superior statistical and numerical characteristics, and is well tailored to guide the multigrid data flow.

In this study, a two-dimensional (2D) problem is considered to establish the theoretical background. Fig. 1 shows a series of SVB-LODs, or just simply LODs, created by sequentially applying two filters: i) the down-sampling filter $\mathcal{F}_d$ and ii) the sharpening filter $\mathcal{F}_s$. A series of SVB-LODs is produced by repeating this procedure until the coarsest LOD, $\mathcal{V}^L$, is obtained. Mathematically, we denote the $l$th LOD image by a two-dimensional container, $\mathcal{V}^l$, which stores greyscale values associated with the digital image. We denote $(\cdot)^l$, $l \in \{0, 1, ..., L\}$, as an identifier for LOD, where $L$ represents the coarsest LOD. Note $L$ is a user defined parameter and does not necessarily lead to a single element LOD case. To index the greyscale values, we use spatial indexes $i$ and $j$ which satisfy $i, j \in [1, H^l]$, where $H^l$ is the size of the image in terms of the pixel number along one side. In our case, the image sizes along two directions are chosen to be equal.

The down-sampling filter ($\mathcal{F}_d$) is a linear filter commonly used for image compression [25,99,100]. The sharpening filter ($\mathcal{F}_s$) is developed in [25] to reconstruct the original phase contrast. In short, this filter creates the $l$th SVB-LOD microstructures by approximately solving

$$\begin{aligned} &\textit{Minimize:} \quad && \sum_{\rho} \left| \mathcal{P}^l(\rho) - \mathcal{P}^0(\rho) \right|, \\ &\textit{Subject to:} \quad && \left| \mathcal{V}^l_{ij} - \hat{\mathcal{V}}^l_{ij} \right| \le d, \ \forall i, j \in [1, H^l], \end{aligned} \tag{1}$$

where $\mathcal{P}^l$ is the greyscale probability mass function (PMF) of $\mathcal{V}^l$, $\hat{\mathcal{V}}^l$ is the intermediate image after down-sampling and before sharpening, $d > 0$ is a prescribed maximum range of the greyscale value interchange for every pixel (in this work, we use $d = 1$), and $\rho \in [0, 255]$ is the greyscale value that is related to the material density. A novel numerical algorithm is described in [25] to solve Eq. (1) by matching the greyscale value PMF through a fast sweeping strategy with local volume preservation. Using the SVB method, a series of compressed images (SVB-LODs) are created from the original data (i.e., the experimental micrographs) with a minimal loss of the microstructural information. After establishing the SVB series, the Otsu's method [101] is utilized to binarize the greyscale images, thereby separating the phases. More details on SVB-LOD construction can be found in [25].

In this work, we created the SVB-LODs from experimental data of Ni/Al high energy ball milled composites [25,73]. The SVB images employed in this work are the 0th – 3rd LODs. A series of SVB microstructures is shown in Fig. 2. The associated parameters are listed in Table 1. Note that these microstructures are from a real physical system, meaning the original irregular and tortuous morphology is unaltered. This results in an increased amount of phase boundaries with a sharp change in the coefficient contrast. This also implies significantly increased numerical difficulty for convergence, compared to geometries employed in [98,102,103].

### 2.1. SVB mapping to grids

The SVB hierarchy provides geometrical and physical/material information in all intermediate levels, which are usually missed in multigrid methods for random microstructures that have been developed so far [57,77]. The correspondence of one image and one grid is shown in Fig. 3. One binarized image of the $l$th SVB-LOD is shown in Fig. 3(a). This binarized $l$th SVB-LOD image contains binary values $\bar{\rho} \in \{0, 1\}$. Here, $\bar{\rho} = 0$ indicates the black color and $\bar{\rho} = 1$ indicates the white color. As described earlier in this section, the image size is $H^l \times H^l$ in terms of pixel number. Each pixel represents a squared material element, as depicted in Fig. 3(b). It is then natural to differentiate the two material domains on the $l$th LOD by

$$\begin{aligned} \Omega^l_1 &= \left\{ (i, j) | \mathcal{V}^l_{ij} = 0, \ i, j \in [1, H^l] \right\}, \\ \Omega^l_2 &= \left\{ (i, j) | \mathcal{V}^l_{ij} = 1, \ i, j \in [1, H^l] \right\}, \end{aligned} \tag{2}$$

where the black domain is denoted by $\Omega^l_1$ and the white domain is denoted by $\Omega^l_2$, respectively.

One pixel inside of the image corresponds to one element in the numerical analysis. Accordingly, the material coefficients (e.g., thermal conductivity) vary based on which phase the element lies in. Thus, a discrete coefficient field is obtained for each member of the SVB-LODs. We denote this discrete material coefficient of the $l$th SVB-LOD by $\kappa^l$. The coefficient inside the element $(i, j)$ of the $l$th SVB-LOD image is therefore denoted by $\kappa^l_{i,j}$ (see Fig. 3(c)), where

$$\kappa^l_{i,j} = \begin{cases} \kappa_1, & \text{if } (i, j) \in \Omega^l_1, \\ \kappa_2, & \text{if } (i, j) \in \Omega^l_2. \end{cases} \tag{3}$$

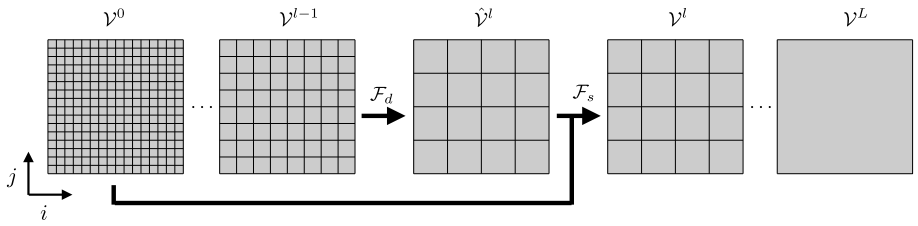


**Fig. 1.** Schematic of creating SVB-LODs using the approach presented in [25].

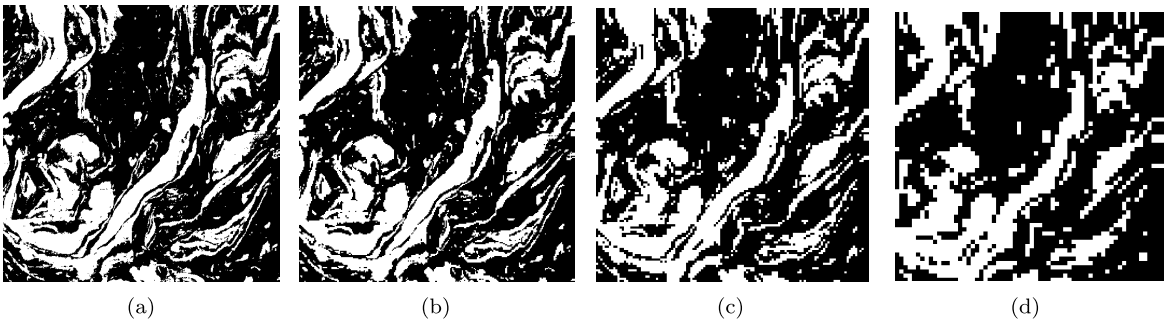


**Fig. 2.** The SVB microstructures of a Ni/Al high energy ball milled composite from different LODs. The Al phase is shown in black. The Ni phase is shown in white. (a) – (d) show 0th to 3rd LOD microstructures, respectively.

**Table 1**
Parameters of the 0th – 3rd SVB-LOD microstructures and the corresponding grids.

| LOD $l$ | Physical size [$\mu m^2$] | Number of elements $H^l \times H^l$ | Grid spacing $\Delta h^l$ [nm] | Compression ratio |
|---|---|---|---|---|
| 0 | $7.68 \times 7.68$ | $512 \times 512$ | 15 | – |
| 1 | $7.68 \times 7.68$ | $256 \times 256$ | 30 | 4 |
| 2 | $7.68 \times 7.68$ | $128 \times 128$ | 60 | 16 |
| 3 | $7.68 \times 7.68$ | $64 \times 64$ | 120 | 64 |

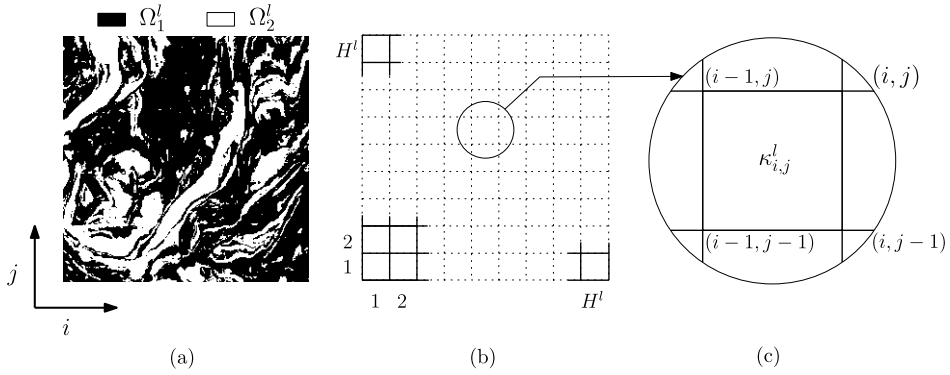


**Fig. 3.** Mapping of one binary SVB image to a structured grid. (a) The $l$th binary SVB-LOD image, $\mathcal{V}^l$. (b) The corresponding $l$th level structured $H^l \times H^l$ grid. (c) Part of the grid with a discrete material coefficient, $\kappa_{i,j}^l$.

Note that for the sake of making the notation system clean, we simply employ the same index for the element and the node. Particularly, the element $(i, j)$ and the node that is located at the top right corner of this element share the same index (see Fig. 3(c)). In this way, the index is dependent on the type of variable, i.e., a cell variable or a nodal variable. This indexing rule also contains information about the relative distance among different types of variables. By making the image sizes as $H^{l-1} = 2H^l$, the series of images can be mapped to a hierarchy of structured grids, which can then be easily employed in a multigrid solver.

It is straightforward to map the SVB images to a series of systems of algebraic equations if we use numerical discretizations implied at each image level independently. In fact, solutions from these images closely reflect and converge to the solution on the finest level (i.e., the 0th LOD), as shown in [25]. We denote the equations obtained from the SVB images as

$$\mathbb{A}^l \mathbb{U}^l = \mathbb{Q}^l, \qquad l \in \{0, 1, ..., L\}, \tag{4}$$

where $\mathbb{A}^l \in \mathbb{R}^{M^l \times M^l}$ is a typical coefficient matrix, $\mathbb{U}^l \in \mathbb{R}^{M^l}$ is the solution vector, and $\mathbb{Q}^l \in \mathbb{R}^{M^l}$ is the forcing term. We will utilize this multiscale model to our advantage when developing the image-based multiscale multigrid solver, preconditioner, and the ROM.

## 3. Image-based multiscale multigrid solver

In this study, we choose to solve an elliptic problem [57,58]. For this purpose, we consider the following boundary value problem

$$\begin{aligned} \nabla \cdot (\kappa(\boldsymbol{x}) \nabla u(\boldsymbol{x})) &= Q(\boldsymbol{x}) \text{ in } \Omega, \\ u(\boldsymbol{x}) &= \bar{u}(\boldsymbol{x}) \text{ on } \partial\Omega, \end{aligned} \tag{5}$$

where $\Omega$ is composed of two distinct phases, $\Omega = \Omega_1 \cup \Omega_2$, $\Omega_1 \cap \Omega_2 = \varnothing$ (see Fig. 3). The $\kappa(\boldsymbol{x})$ is a piecewise scalar field:

$$\kappa(\boldsymbol{x}) = \begin{cases} \kappa_1, & \text{if } \boldsymbol{x} \in \Omega_1, \\ \kappa_2, & \text{if } \boldsymbol{x} \in \Omega_2. \end{cases} \tag{6}$$

Using the standard Galerkin finite element formulation [26], the above boundary value problem can be expressed as a system of linear equations

$$\boldsymbol{A}\boldsymbol{u} = \boldsymbol{Q}, \qquad \boldsymbol{A} \in \mathbb{R}^{M \times M}, \ \boldsymbol{u} \in \mathbb{R}^M, \ \boldsymbol{Q} \in \mathbb{R}^M, \tag{7}$$

where $\boldsymbol{A}$ is a symmetric positive definite sparse matrix, $\boldsymbol{u}$ is the solution vector, and $\boldsymbol{Q}$ is the forcing term. In this case, $M$ denotes the number of DOFs or, equivalently, the number of nodes.

In order to solve this problem, we link the multigrid method with our image-based multiscale model. Specifically, we utilize the multiscale images to provide reliable microstructural information on the coarser levels during the restriction and prolongation processes. As a result, the multigrid method takes advantage of the coarse level information from LODs. Meanwhile, the multiscale images get connected through multigrid components. A schematic of this strategy is shown in Fig. 4.

To demonstrate this point and motivate our image-based multiscale multigrid algorithm, we first highlight the key components of a multigrid algorithm in Section 3.1 and then discuss the novel image-based inter-grid operator in Section 3.2.

### 3.1. Multigrid principles

The multigrid method is generally referred to as the type of numerical algorithm that solves a system of algebraic equations utilizing a hierarchy of grids [32,34,104]. In the multigrid setting, Eq. (7) becomes

$$\boldsymbol{A}^l \boldsymbol{u}^l = \boldsymbol{Q}^l, \tag{8}$$

on the $l$th grid, where $\boldsymbol{A}^l \in \mathbb{R}^{M^l \times M^l}$, $\boldsymbol{u}^l \in \mathbb{R}^{M^l}$, $\boldsymbol{Q}^l \in \mathbb{R}^{M^l}$ and $l \in \{0, 1, ..., L\}$. Here, the algebraic coarsening of a typical algebraic multigrid method is utilized to compute the system of equations on coarser levels (i.e., $1 \leq l \leq L$). Thus, the coarse coefficient matrices are computed algebraically following the Galerkin condition [98,104]

$$\boldsymbol{A}^{l+1} = \boldsymbol{R}_l^{l+1} \boldsymbol{A}^l \boldsymbol{P}_{l+1}^l, \tag{9}$$

where $\boldsymbol{R}_l^{l+1} \in \mathbb{R}^{M^{l+1} \times M^l}$ and $\boldsymbol{P}_{l+1}^l \in \mathbb{R}^{M^l \times M^{l+1}}$ are the inter-grid operators. The operators relate individual grids as:

$$\begin{aligned} \boldsymbol{R}_l^{l+1} &= \text{Restriction operator, grid } l \mapsto \text{grid } l+1, \\ \boldsymbol{P}_{l+1}^l &= \text{Prolongation operator, grid } l+1 \mapsto \text{grid } l. \end{aligned}$$

As is often the case, $\boldsymbol{A}^l = (\boldsymbol{A}^l)^\intercal, \ \forall l \in \{0, 1, ..., L\}$, which requires

$$\boldsymbol{P}_{l+1}^l = (\boldsymbol{R}_l^{l+1})^\intercal. \tag{10}$$

Note that in this work, we focus on the algebraic multigrid. The geometric multigrid is traditionally avoided while solving Eq. (5) with fluctuating coefficients due to the complexity in approximating the smooth component of the solution using subsequent coarser grids [57,71]. In our case, even with the coarse level SVB approximations (i.e., sequence of LODs) [25], the geometrical multigrid is still not preferable. This is because the construction of the geometric multigrid solver for problems with complex microstructures (see Fig. 2) is computationally more intensive than solving the original problem. Specifically, the admissible $\boldsymbol{R}_l^{l+1}$ and $\boldsymbol{P}_{l+1}^l$ need to satisfy

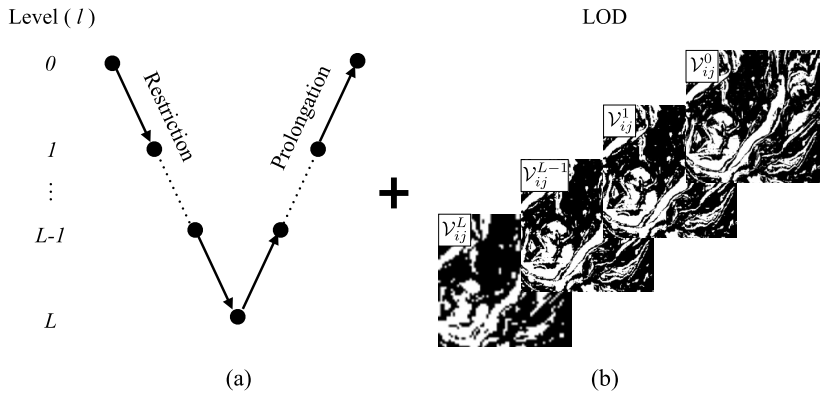


**Fig. 4.** A schematic of the image-based multiscale multigrid solver. (a) Grid levels and a corresponding multigrid cycle. (b) Image levels from the SVB model (see Section 2 and Fig. 2).

$$\mathbb{A}^{l+1} - \boldsymbol{R}_l^{l+1}\mathbb{A}^l\boldsymbol{P}_{l+1}^l = \boldsymbol{0},\ \forall l \in \{0, ..., L-1\}, \tag{11}$$

which results in $\sum_{l=1}^{L} M^l\left(M^l+1\right)/2$ non-linear equations to be iteratively solved, considering the symmetry of $\mathbb{A}^l$ (see Eq. (4)) and Eq. (10). This implies increased complexity compared to solving $M^{l=0}$ unknowns from the original linear problem (i.e., Eq. (7)).

In this work, we employ the V-cycle multigrid (see Fig. 4(a)). The key steps of a multigrid V-cycle are described in Algorithm 1. The user defined parameters are the number of smoothing iterations ($\nu$), the multigrid depth ($L$), and the smoothing algorithm ($\boldsymbol{S}^l$ and $\boldsymbol{F}^l$).

**Algorithm 1** V-cycle multigrid: $\boldsymbol{u}^l \leftarrow \mathcal{MG}(\boldsymbol{u}^l, \boldsymbol{A}^l, \boldsymbol{Q}^l, L, \nu)$.

1: **if** the coarsest grid is reached, $(l = L)$ **then**
2: Direct solve, $\boldsymbol{u}^l \leftarrow \left(\boldsymbol{A}^l\right)^{-1}\boldsymbol{Q}^l$
3: **else**
4: Apply smoothing $\nu$ times, $\boldsymbol{u}^l \leftarrow \boldsymbol{S}^l\boldsymbol{u}^l + \boldsymbol{F}^l\boldsymbol{Q}^l$
5: Apply restriction, $\boldsymbol{Q}^{l+1} \leftarrow \boldsymbol{R}_l^{l+1}(\boldsymbol{Q}^l - \boldsymbol{A}^l\boldsymbol{u}^l)$
6: Initialize, $\boldsymbol{e}^{l+1} \leftarrow \boldsymbol{0}$
7: Update, $\boldsymbol{e}^{l+1} \leftarrow \mathcal{MG}(\boldsymbol{e}^{l+1}, \boldsymbol{A}^{l+1}, \boldsymbol{Q}^{l+1}, L, \nu)$
8: Apply prolongation, $\boldsymbol{e}^l \leftarrow \boldsymbol{P}_{l+1}^l\boldsymbol{e}^{l+1}$
9: Apply correction, $\boldsymbol{u}^l \leftarrow \boldsymbol{u}^l + \boldsymbol{e}^l$
10: Apply smoothing $\nu$ times, $\boldsymbol{u}^l \leftarrow \boldsymbol{S}^l\boldsymbol{u}^l + \boldsymbol{F}^l\boldsymbol{Q}^l$

### 3.2. Image-based inter-grid operator

To prevent the multigrid solver from severe stagnation caused by the oscillatory coefficients in Eq. (6), we have designed a novel image-based inter-grid operator. This operator employs the multiscale information flow from SVB (see Figs. 2 and 4), resulting in a more accurate mapping of variables among grids. This inter-grid operator is built through a two-stage approach: i) the approximation stage and ii) the refinement stage. Accordingly, prolongation of the error between two grids (see Algorithm 1 Line 8) is decomposed into:

$$\text{Approximation} \quad \tilde{\boldsymbol{e}}^l \leftarrow \mathcal{P}_a\boldsymbol{e}^{l+1}, \tag{12a}$$

$$\text{Refinement} \quad \boldsymbol{e}^l \leftarrow \mathcal{P}_r\tilde{\boldsymbol{e}}^l. \tag{12b}$$

Here, $\boldsymbol{e}^{l+1}$ is the known coarse grid error, $\boldsymbol{e}^l$ is the fine grid error to be obtained through prolongation, and $\tilde{\boldsymbol{e}}^l$ is an approximate error that is close to $\boldsymbol{e}^l$. In order to compute $\tilde{\boldsymbol{e}}^l$, a linear operator containing weights from SVB multiscale images ($\mathcal{P}_a \in \mathbb{R}^{M^l \times M^{l+1}}$) is constructed. Next, the refinement operator $\mathcal{P}_r \in \mathbb{R}^{M^l \times M^l}$ is created to refine $\tilde{\boldsymbol{e}}^l$. Thus, the prolongation operator satisfies:

$$\boldsymbol{P}_{l+1}^l = \mathcal{P}_r\mathcal{P}_a. \tag{13}$$

Since the explicit expressions for $\mathcal{P}_a$ and $\mathcal{P}_r$ are not intuitive, we describe both operators by showing the calculation of each component.

During the approximation stage (see Eq. (12a)), $\tilde{\boldsymbol{e}}^l$ is computed through the multiscale image (see Figs. 2 and 3). To compute the weights in $\mathcal{P}_a$, four computational strategies are considered depending on the location of the fine node relative

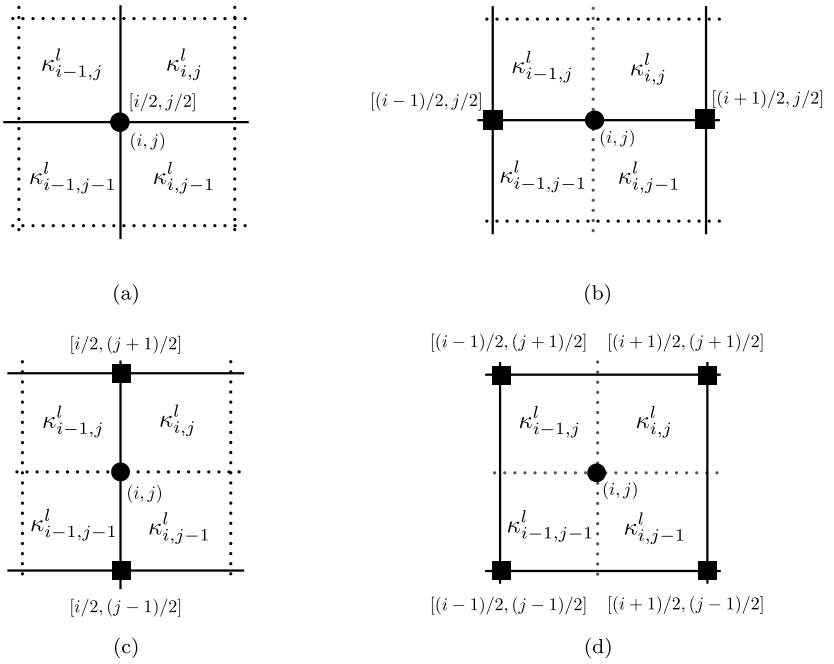


**Fig. 5.** The image-based inter-grid operator scheme. The solid lines (——) denote the coarse grid. The dotted lines (·····) denote the fine grid. The circle (●) marks the nodal value to be prolongated, whose indexes are denoted using $(\cdot, \cdot)$. The square (■) marks the nodal value known from the coarse grid, whose indexes are denoted using $[\cdot, \cdot]$. The $\kappa^l_{i,j}$ is the element-wise material property acquired from the $l$th SVB-LOD microstructure (see Eq. (3)).

to the coarse grid. The location types are (a) coarse grid nodes, (b) center of the horizontal coarse grid lines, (c) center of the vertical coarse grid lines, and (d) center of the coarse element. Each type corresponds to a subplot in Fig. 5 and is represented by a $2 \times 2$ fine grid window centered by the nodal value to be obtained (node $(i, j)$).

When the fine grid node coincides with the coarse grid node (see Fig. 5(a)), we get

$$\tilde{\boldsymbol{e}}^l_{i,j} = \boldsymbol{e}^{l+1}_{i/2,j/2}. \tag{14}$$

Note that indexes above follow Fig. 3(b) and denote an identical spacial location in both the $l$th grid and the $(l+1)$th grid. When the fine node is located at the center of the horizontal coarse grid line (see Fig. 5(b)), we write

$$\tilde{\boldsymbol{e}}^l_{i,j} = W^l_{i,j}\boldsymbol{e}^{l+1}_{(i-1)/2,j/2} + E^l_{i,j}\boldsymbol{e}^{l+1}_{(i+1)/2,j/2}, \tag{15}$$

where $W^l_{i,j}$ and $E^l_{i,j}$ are weights calculated from

$$W^l_{i,j} = \frac{\kappa^l_{i-1,j} + \kappa^l_{i-1,j-1}}{d^l_{i,j}}, \quad E^l_{i,j} = \frac{\kappa^l_{i,j} + \kappa^l_{i,j-1}}{d^l_{i,j}}. \tag{16}$$

Here we define $d^l_{i,j} = \kappa^l_{i-1,j} + \kappa^l_{i,j} + \kappa^l_{i-1,j-1} + \kappa^l_{i,j-1}$, $W^l_{i,j}$ and $E^l_{i,j}$ represent the weight from the west and the east (i.e., compass based nomenclature) of the node $(i, j)$, respectively. Note that material coefficients $\kappa^l_{ij}$ are obtained directly from the $l$th SVB-LOD microstructure (see Eq. (3)). Similarly, for the case shown in Fig. 5(c), we have

$$\tilde{\boldsymbol{e}}^l_{i,j} = N^l_{i,j}\boldsymbol{e}^{l+1}_{i/2,(j+1)/2} + S^l_{i,j}\boldsymbol{e}^{l+1}_{i/2,(j-1)/2}, \tag{17}$$

where

$$N^l_{i,j} = \frac{\kappa^l_{i-1,j} + \kappa^l_{i,j}}{d_{i,j}}, \quad S^l_{i,j} = \frac{\kappa^l_{i-1,j-1} + \kappa^l_{i,j-1}}{d_{i,j}}. \tag{18}$$

When the fine grid node is located at the center of the coarse grid element (see Fig. 5(d)), we get

$$\begin{aligned}\tilde{\boldsymbol{e}}^l_{i,j} = {} & NW^l_{i,j}\boldsymbol{e}^{l+1}_{(i-1)/2,(j+1)/2} + NE^l_{i,j}\boldsymbol{e}^{l+1}_{(i+1)/2,(j+1)/2} \\ & + SW^l_{i,j}\boldsymbol{e}^{l+1}_{(i-1)/2,(j-1)/2} + SE^l_{i,j}\boldsymbol{e}^{l+1}_{(i+1)/2,(j-1)/2},\end{aligned} \tag{19}$$

and the weights are

$$NW^l_{i,j} = \frac{1}{2}(N^l_{i,j}W^l_{i,j+1} + W^l_{i,j}N^l_{i-1,j}), \quad NE^l_{i,j} = \frac{1}{2}(E^l_{i,j}N^l_{i+1,j} + N^l_{i,j}E^l_{i,j+1}),$$
$$SW^l_{i,j} = \frac{1}{2}(W^l_{i,j}S^l_{i-1,j} + S^l_{i,j}W^l_{i,j-1}), \quad SE^l_{i,j} = \frac{1}{2}(S^l_{i,j}E^l_{i,j-1} + E^l_{i,j}S^l_{i+1,j}). \tag{20}$$

Note again that in the approximation stage, the material coefficients $\kappa^l_{i,j}$ (e.g., thermal conductivities) are obtained directly from the SVB-LOD images (see Fig. 2 and Eq. (3)). Thus, the approximation stage has the geometric character of the error transfer. This is different from the black-box prolongation and restriction operators [56–59,98] which do not have access to the individual microstructural levels (see Figs. 2 and 3).

With known $\tilde{\boldsymbol{e}}^l$ from the first stage (i.e., approximation), we can express $\boldsymbol{e}^l$ with respect to its neighboring approximate error components by utilizing the residual equation (see Eq. (12b)). We name this process as the refinement stage, which is mathematically expressed as

$$\boldsymbol{e}^l_\alpha = -\frac{1}{\boldsymbol{A}^l_{\alpha\alpha}} \sum_{\beta\in[1,\alpha)\cup(\alpha,M^l]} \boldsymbol{A}^l_{\alpha\beta}\tilde{\boldsymbol{e}}^l_\beta, \quad \forall\alpha\in[1,M^l], \tag{21}$$

where $\boldsymbol{A}^l_{\alpha\beta}$ is computed recursively from Eq. (9) starting form $\boldsymbol{A}^0$, $\alpha$ and $\beta\in[1,M^l]$ are the row and column indexes, respectively. Note that the subscript $\alpha \longleftrightarrow (i,j)$ depends on the numbering of the nodes. The above equation couples each nodal error component with their neighboring approximate error components, as a result of the finite element discretization. Thus, the refinement stage has the algebraic character of the error transfer. Moreover, Eq. (21) weakly preserves the flux continuity as shown in [98]. We call this a refinement stage because it is a close approximation of

$$\sum_{\beta=1}^{M^l} \boldsymbol{A}^l_{\alpha\beta}\boldsymbol{e}^l_\beta = 0, \quad \forall\alpha\in[1,M^l]. \tag{22}$$

Equation (22) would yield an ideal prolongation and the residual would approach zero. Substituting Eqs. (14), (15), (17) and (19) into Eq. (21) results in the formulation of the novel image-based prolongation operator, $\boldsymbol{P}^l_{l+1} = \mathcal{P}_r\mathcal{P}_a$. Knowing the prolongation operator, the restriction operator can be acquired by simply taking the transpose (see Eq. (10)). With this new image-based inter-grid operator, we obtain the image-based multiscale multigrid solver.

Our formulation remedies the matrix-dependent approaches [56–58,98] by eliminating the simplistic assumption that error components are equal along the coarse grid lines. At the same time, the geometrical (i.e., image-based) information is utilized to provide a more realistic and physically meaningful approximation for the system of equations between grids as will be shown in Section 6.1. In addition, this formulation preserves the continuity of the flux (e.g., the heat flux) and satisfies Eq. (5) on the fine grid. In order to keep the discussion concise, the proof of the flux continuity is included in Appendix A.

### 3.3. Convergence rate estimates

The asymptotic convergence rate of the V-cycle multigrid (Algorithm 1) can be estimated as follows

$$\Psi = -\log_{10}\left(\lambda_{\max}\left(\boldsymbol{\mathcal{M}}^l\right)\right), \tag{23}$$

where $\Psi$ represents the lower estimate of the convergence rate in terms of digits per iteration, $\boldsymbol{\mathcal{M}}^l\in\mathbb{R}^{M^l\times M^l}$ denotes the iteration operator of the multigrid algorithm (see Appendix B), the $\lambda_{\max}(\cdot)$ denotes the spectral radius [105,106], and

$$\lambda_{\max}\left(\boldsymbol{\mathcal{M}}^l\right) = \max\{|\lambda| : \lambda \text{ is the eigenvalue of } \boldsymbol{\mathcal{M}}^l\}. \tag{24}$$

It can be shown that Eq. (23) is a lower bound of the actual convergence rate, i.e.,

$$\Psi \le -\log_{10}\left(\frac{||\boldsymbol{\epsilon}^l_{n+1}||}{||\boldsymbol{\epsilon}^l_n||}\right), \tag{25}$$

where $||\cdot||$ denotes the $L^2$-norm and $\boldsymbol{\epsilon}^l_n$ denotes the error vector after the $n$th iteration. The proof of Eq. (25) is included in Appendix B.

## 4. Image-based multiscale preconditioner

The general idea of a preconditioning procedure for iterative solvers is to modify the system of algebraic equations (Eq. (7)) such that it is well conditioned. Different from other classical multigrid preconditioners, in our work, we prolongate the coarse problem directly from SVB images (see Fig. 2) to formulate a preconditioner for the fine grid problem. This image-based multiscale preconditioner has the following definition

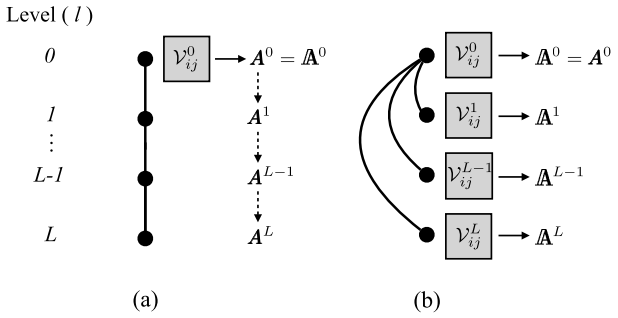


**Fig. 6.** A schematic of the multiscale preconditioners. (a) A typical multigrid preconditioner and associated algebraic coefficient matrices. (b) The image-based multiscale preconditioner and associated geometric coefficient matrices. The solid arrow (→) denotes the process of using finite element method on a certain SVB-LOD microstructure to obtain a geometric system (i.e., Eqs. (4) and (27)). The dashed arrow (- - →) denotes the process of algebraic coarsening following Eq. (9).

$$\boldsymbol{M}_S^{-1}(L) = \left(\prod_{l=0}^{L-1} \boldsymbol{P}_{l+1}^{l}\right) \left(\mathbb{A}^L\right)^{-1} \left(\prod_{l=0}^{L-1} \boldsymbol{R}_{L-l-1}^{L-l}\right) + \left(\mathrm{diag}(\boldsymbol{A}^0)\right)^{-1}, \tag{26}$$

where $L > 1$, $\boldsymbol{M}_S^{-1}(L) \in \mathbb{R}^{M^0 \times M^0}$ is the preconditioner on the finest (0th) level, $\boldsymbol{P}_{l+1}^{l}$ and $\boldsymbol{R}_{l}^{l+1}$ are the image-based inter-grid operators (see Section 3.2). The matrix $\mathbb{A}^L$ (see Eq. (4)) is the traditional positive definite matrix obtained from the finite element method

$$\mathbb{A}^L = \int_{\Omega} \boldsymbol{\mathcal{B}}^{\intercal} \kappa^L \boldsymbol{\mathcal{B}} \, \mathrm{d}\Omega, \tag{27}$$

where $\kappa^L$ is from the $L$th SVB-LOD microstructure (Section 2 and Eq. (3)) and $\boldsymbol{\mathcal{B}}$ is the spatial gradient matrix (i.e., the derivatives of shape functions). Here, we compute $\left(\mathbb{A}^L\right)^{-1}$ using a direct LDL$^{\intercal}$ solver [107]. Different iterative techniques are also possible.

There are several remarks about this formulation. First, this preconditioner is written as a summation of the coarse and the fine level parts. The former is the approximate inverse obtained from the prolongation of the coarsest level. The latter is the inverse of the finest diagonal matrix, which is utilized as a regularization term in order to retain the positive-definiteness of the preconditioner. Second, it should be noted that $\mathbb{A}^L$ ($\neq \boldsymbol{A}^L$) arises from the numerical discretization and integration utilizing the $L$th level image (Eqs. (4) and (27)), not from the algebraic computations following Eq. (9). These characteristics imply that instead of traversing through all coefficient matrices on all intermediate levels, only the diagonal of the finest coefficient matrix and the coarsest coefficient matrix will be utilized to compute the preconditioner. This feature is demonstrated in Fig. 6, where intermediate levels are bypassed for the novel image-based multiscale preconditioner (Fig. 6(b)), while classical multigrid preconditioners need to iterate through every intermediate levels (Fig. 6(a)). This direct LOD access leads to the computational speedup as shown in Section 6.3.

## 5. Image-based reduced order model

Early work using Eq. (4) has shown that SVB microstructures preserve statistical and physical solution characteristics with high data compression [25]. Moreover, with the new image-based inter-grid operators (Section 3.2), we are able to guide the mapping of variables between grids with increased accuracy. As a result, we can develop the IROM that utilizes the coarse grid solution to extrapolate the solution on the fine grid. A schematic of this IROM is shown in Fig. 7.

This process can be mathematically described as

$$\boldsymbol{u}_S(L) = \left(\prod_{l=0}^{L-1} \boldsymbol{P}_{l+1}^{l}\right) \mathbb{U}^L, \tag{28}$$

where $L \geq 1$, $\boldsymbol{u}_S(L) \in \mathbb{R}^{M^0}$ represents the approximate solution on the finest (0th) level, $\boldsymbol{P}_{l+1}^{l}$ is the image-based prolongation operator (see Section 3.2), and $\mathbb{U}^L$ is the coarse level solution from Eq. (4). In this work, we compute $\mathbb{U}^L$ from Eq. (4) by

$$\mathbb{U}^L = \left(\mathbb{A}^L\right)^{-1} \mathbb{Q}^L. \tag{29}$$

Here, $\mathbb{Q}^L$ is the forcing vector obtained from the finite element method,

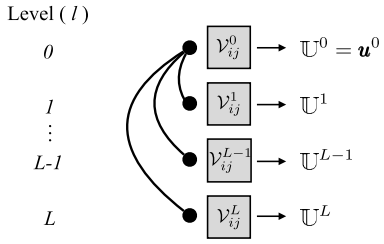


**Fig. 7.** A schematic of the IROM. The solid arrow (→) denotes the process of using the finite element method on a certain SVB-LOD microstructure to obtain a geometric system (i.e., Eq. (4)).

$$\mathbb{Q}^L = \int_\Omega \mathcal{N} Q \, \mathrm{d}\Omega, \tag{30}$$

where $\mathcal{N}$ is the matrix of shape functions and $Q$ is the source term from Eq. (5). Again, we utilize a direct LDL$^\intercal$ solver [107] to invert $\mathbb{A}^L$ in Eq. (29), but different (i.e., iterative) solution methods can be utilized.

Using Eq. (28), the original fine problem is transferred to a coarse one that has reduced number of elements (see Table 1). In this way, the computation cost is significantly reduced. Moreover, with the new image-based inter-grid operator, microstructural information is conveyed through the mapping of variables between grids. Therefore, missing information during data compression provided by the reduced system, Eq. (29), is largely recovered through the prolongation of the coarse solution field. A solution field obtained on a coarser level can therefore gain improved accuracy through our novel prolongation based IROM.

## 6. Numerical example

In this section, we present numerical results of the image-based multiscale multigrid solver, preconditioner, and IROM described in Sections 3 to 5, respectively. As an illustration, we solve Eq. (5) in a heterogeneous domain $\Omega$ (see Fig. 2). We choose $Q(\boldsymbol{x}) = 1.0$ in $\Omega$ and $\bar{u}(\boldsymbol{x}) = 0$ on $\partial\Omega$. All fields are unitless for simplicity. The coefficient $\kappa_1 = 1.0$, and $\kappa_2$ ranges between 1 and 10000 to examine sufficient variability in the coefficient contrast. The goal is to solve Eq. (7) using the finest finite element discretization, i.e., $\boldsymbol{A}^0\boldsymbol{u}^0 = \boldsymbol{Q}^0$ (Eq. (8)) or equivalently $\mathbb{A}^0\mathbb{U}^0 = \mathbb{Q}^0$ (Eq. (4)). Traditional bi-linear quadrilateral finite elements are utilized [26]. All of our implementations have been done utilizing the C++ package Eigen [107]. The 1st-3rd SVB-LOD images (see Fig. 2) are employed as auxiliary LODs. We consider the solution to be converged if

$$\frac{||\boldsymbol{r}_n||}{||\boldsymbol{r}_0||} < 10^{-6}, \tag{31}$$

where $\boldsymbol{r}_0$ is the initial residual vector, $\boldsymbol{r}_n$ is the residual vector after the $n$th iteration, and the $||\cdot||$ denotes the Euclidean norm.

In order to evaluate the efficiency of our algorithms, we show the speedup for the image-based multiscale multigrid solver (Section 6.2) as well as the image-based multiscale preconditioner (Section 6.3). The speedup is computed as

$$\text{Speedup} = \frac{t_{\text{ref}}}{t}, \tag{32}$$

where $t_{\text{ref}}$ denotes the computer time spent by the reference algorithm and the $t$ denotes the computer time spent by novel algorithms. The reference algorithm is chosen among the most fundamental algorithms for easy comparison. We use speedup instead of the actual computer time in order to limit effects of the software implementation and the hardware architecture. In this paper, we show the average speedup of 10 runs. The maximum coefficient of variation is 3.46%.

### 6.1. Numerical characteristics of the image-based inter-grid operator

In this section, we show the numerical properties of the image-based inter-grid operator (see Section 3.2) and compare them to the properties of the inter-grid operator employed in the black-box multigrid method [56–59,98]. The black-box multigrid employs matrix-dependent inter-grid operators that are computed from the finest coefficient matrix. This method has been considered the state-of-art for the targeted problem. We denote our image-based multiscale multigrid method by $\mathcal{MG}_S(L)$. Similarly, the black-box multigrid is denoted by $\mathcal{MG}_{Box}(L)$. We show the matrix properties of the 0th – 3rd grid (i.e., $0 \le l \le 3$) for both methods. The properties are shown with $\kappa_2/\kappa_1 = 10000$.

In Table 2, we list the numerical properties of $\boldsymbol{P}_l^{l-1}$ and $\boldsymbol{A}^l$ with $0 \le l \le 3$ for both $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$ methods. To assess the quality of the coarse coefficient matrices, we utilize both the spectral norm, $||\cdot||_2$, and the distance correlation coefficient, dCor$(\cdot,\cdot)$, which is a measure of dependence between two random objects/matrices of arbitrary, not necessarily equal, dimension. It can be seen that the $||\boldsymbol{P}_l^{l-1}||_2$ values from $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$ are significantly different from

**Table 2**
The numerical properties of $\boldsymbol{P}_l^{l-1}$ and $\boldsymbol{A}^l$. The results are shown for both $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$ methods. The $||\cdot||_2$ denotes the spectral norm of a matrix [108]. The dCor$(\cdot,\cdot)$ denotes the distance correlation coefficient between two matrices [109,110]. Here, the $||\mathbb{A}^l||_2$ and $||\boldsymbol{A}^0||_2$ are shown for comparison.

| $l$ | $\|\|\mathbb{A}^l\|\|_2$ | $\mathcal{MG}_S(L)$ | | | $\mathcal{MG}_{Box}(L)$ | | |
|---|---|---|---|---|---|---|---|
| | | $\|\|\boldsymbol{P}_l^{l-1}\|\|_2$ | $\|\|\boldsymbol{A}^l\|\|_2$ | dCor$(\boldsymbol{A}^l, \boldsymbol{A}^0)$ | $\|\|\boldsymbol{P}_l^{l-1}\|\|_2$ | $\|\|\boldsymbol{A}^l\|\|_2$ | dCor$(\boldsymbol{A}^l, \boldsymbol{A}^0)$ |
| 0 | $3.944 \times 10^4$ | – | $3.944 \times 10^4$ | 1.0 | – | $3.944 \times 10^4$ | 1.0 |
| 1 | $3.853 \times 10^4$ | 3.870 | $3.937 \times 10^4$ | 0.992 | 2.999 | $3.856 \times 10^4$ | 0.989 |
| 2 | $3.761 \times 10^4$ | 4.366 | $3.826 \times 10^4$ | 0.967 | 2.878 | $3.940 \times 10^4$ | 0.960 |
| 3 | $3.397 \times 10^4$ | 6.970 | $3.693 \times 10^4$ | 0.860 | 18.754 | $3.349 \times 10^4$ | 0.823 |

**Table 3**
The memory use in terms of number of nonzeros (i.e., nnz$(\cdot)$) in matrices $\boldsymbol{P}_l^{l-1}$ and $\boldsymbol{A}^l$. The results are shown for both $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$ methods. Here, the nnz$(\boldsymbol{A}^0)$ is shown as a reference.

| $l$ | $\mathcal{MG}_S(L)$ | | $\mathcal{MG}_{Box}(L)$ | |
|---|---|---|---|---|
| | nnz$(\boldsymbol{P}_l^{l-1})$ | nnz$(\boldsymbol{A}^l)$ | nnz$(\boldsymbol{P}_l^{l-1})$ | nnz$(\boldsymbol{A}^l)$ |
| 0 | – | $2.346 \times 10^6$ | – | $2.346 \times 10^6$ |
| 1 | $1.104 \times 10^6$ | $1.611 \times 10^6$ | $5.914 \times 10^5$ | $5.914 \times 10^5$ |
| 2 | $6.538 \times 10^5$ | $1.258 \times 10^6$ | $1.482 \times 10^5$ | $1.482 \times 10^5$ |
| 3 | $3.034 \times 10^5$ | $7.084 \times 10^5$ | $3.725 \times 10^4$ | $3.725 \times 10^4$ |

each other for all $l$ values. This is because of the difference in our image-based approach (see Section 3.2) compared to the matrix-dependent approach (see [56–59,98]). Table 2 shows that $||\boldsymbol{A}^l||_2$ values for $\mathcal{MG}_S(L)$ are closer to $||\boldsymbol{A}^0||_2$ compared to $\mathcal{MG}_{Box}(L)$. Similarly, the dCor$(\boldsymbol{A}^l, \boldsymbol{A}^0)$ values for $\mathcal{MG}_S(L)$ are closer to 1.0 compared to $\mathcal{MG}_{Box}(L)$ for all grid levels. This indicates that the coarse systems from $\mathcal{MG}_S(L)$ (i.e., $\boldsymbol{A}^l$, $l \in \{1, 2, 3\}$) are statistically/numerically similar or closer to the original system (i.e., $\boldsymbol{A}^0$) than those produced by $\mathcal{MG}_{Box}(L)$. Thus, $\mathcal{MG}_S(L)$ preserves the original property of the original system better than $\mathcal{MG}_{Box}(L)$ (see Table 2). Moreover, one can see that $||\boldsymbol{A}^l||_2$ from $\mathcal{MG}_S(L)$ converges to $||\boldsymbol{A}^0||_2$ monotonically as do the matrices of the level-wise problems (i.e., $\mathbb{A}^l$ from Eq. (4)). However, $\mathcal{MG}_{Box}(L)$ results in nonmonotonic characteristics in terms of the spectral norm, which are less physical due to its heuristic construction.

In Table 3, we compare the memory usage for $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$ methods. Specifically, the nonzeros of the matrices $\boldsymbol{P}_l^{l-1}$ and $\boldsymbol{A}^l$, $0 \leq l \leq 3$ are listed. The nnz$(\boldsymbol{A}^0)$ is shown as a reference. Table 3 shows that the image-based inter-grid operator of $\mathcal{MG}_S(L)$ is generally more dense than the inter-grid operator of $\mathcal{MG}_{Box}(L)$. This is because the image-based approach does not assume the correlation of the error components nor lumps the coefficient matrix (see Eq. (21)). As a result, the $\boldsymbol{A}^l$ of $\mathcal{MG}_S(L)$ is more dense than that of $\mathcal{MG}_{Box}(L)$ for all $l$ values (see Eq. (9)).

### 6.2. Image-based multiscale multigrid solver

In this section, we show the performance of the image-based multiscale multigrid solver described in Section 3. As a comparison, we also show the performance of the black-box multigrid method [56–59,98]. For all the schemes, we employ the Gauss Seidel method as a smoother. We choose $\nu = 5$ to ensure sufficient smoothing while preventing the smoothing procedure from dominating the computation. For simplicity, we start multigrid iterations with the zero initial solution vector. Here, we denote the bilinear multigrid by $\mathcal{MG}_B(L)$. As a demonstration, we show the performance of all multigrid methods for $L \in \{1, 2, 3\}$, respectively. For each case, we show the performance with $\kappa_2/\kappa_1 \in \{10, 100, 1000, 10000\}$.

Fig. 8 shows the lower bound of the convergence rate, $\Psi$, with the varying coefficient contrast for $\mathcal{MG}_S(1)$, $\mathcal{MG}_{Box}(1)$, and $\mathcal{MG}_B(1)$, respectively. We show that $\mathcal{MG}_S(1)$ has the largest $\Psi$ (i.e., the best estimated convergence rate) for all coefficient contrasts among all solvers. Moreover, the convergence rate estimate of $\mathcal{MG}_S(1)$ does not significantly decrease with the coefficient contrast, especially when $\kappa_2/\kappa_1 \geq 100$. As a comparison, $\mathcal{MG}_{Box}(1)$ exhibits smaller convergence rate than $\mathcal{MG}_S(1)$ for all heterogeneous coefficients ($\kappa_2/\kappa_1 \neq 1$). As expected, the $\mathcal{MG}_B(1)$ has the lowest convergence rate estimate, which significantly decreases with the increased coefficient contrast and leads to a high degree of stagnation. Therefore, in what follows we will compare only $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$ and omit the performance of $\mathcal{MG}_B(L)$.

Fig. 9 shows the relative residual of the one auxiliary grid solvers $\mathcal{MG}_S(1)$ and $\mathcal{MG}_{Box}(1)$ with varying coefficient contrast as a function of the iteration number. The result with constant coefficients ($\kappa_2/\kappa_1 = 1$) is shown in gray color to demonstrate the optimal convergence rate [32,111] for this case. It is indicated in Fig. 9(a) that $\mathcal{MG}_S(1)$ converges to the solution within 5 iterations despite drastic change of $\kappa_2/\kappa_1$. The convergence rate of $\mathcal{MG}_S(1)$ decreases slightly with the increase of $\kappa_2/\kappa_1$. However, this phenomenon is not significant, especially in the case when $\kappa_2/\kappa_1 \geq 100$ (see Fig. 8 also). This shows the robustness of $\mathcal{MG}_S(1)$ to resolve problems with highly fluctuating coefficients. Moreover, the convergence rate is close to the constant coefficient case (i.e., $\kappa_2/\kappa_1 = 1$). Therefore, $\mathcal{MG}_S(1)$ is very close to the theoretical optimum. The performance of $\mathcal{MG}_{Box}(1)$ is shown in Fig. 9(b). It can be seen that $\mathcal{MG}_{Box}(1)$ converges to the solution within 9 iterations for all $\kappa_2/\kappa_1$. However, comparing $\mathcal{MG}_S(1)$ and $\mathcal{MG}_{Box}(1)$, one can see the slower convergence rate of

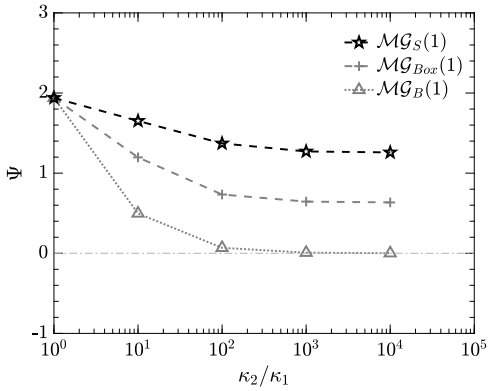


**Fig. 8.** The lower bound of the convergence rate, $\Psi$, as a function of the coefficient contrast. The result is shown for the $\mathcal{MG}_S(1)$, $\mathcal{MG}_{Box}(1)$, and $\mathcal{MG}_B(1)$, respectively.

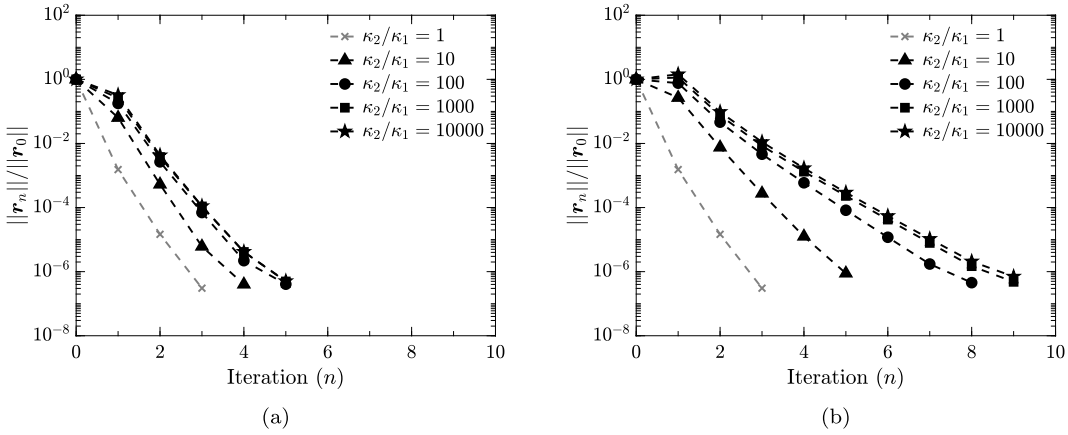


**Fig. 9.** The relative residual as a function of the iteration number with varying $\kappa_2/\kappa_1$. (a) The one auxiliary grid $\mathcal{MG}_S(1)$ solver. (b) The one auxiliary grid $\mathcal{MG}_{Box}(1)$ solver.

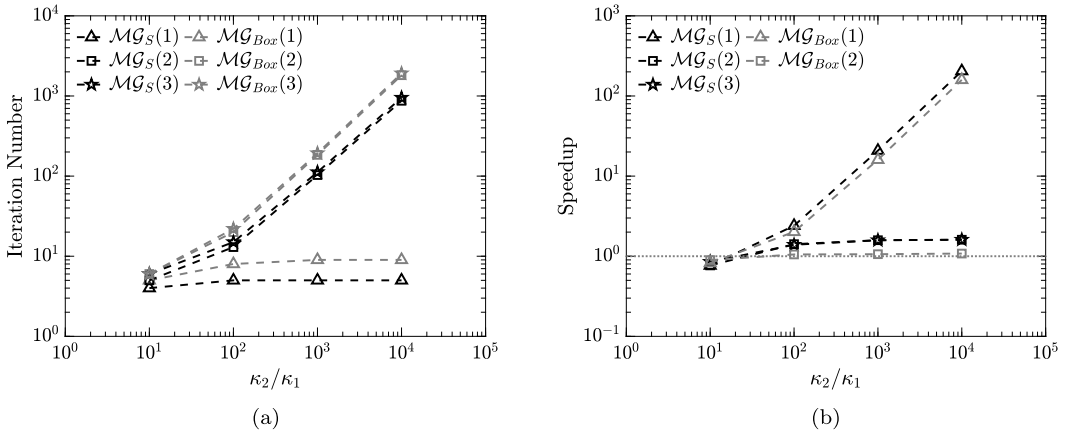


**Fig. 10.** The performance of $\mathcal{MG}_S(L)$, $L \in \{1, 2, 3\}$ with the varying coefficient contrast. (a) Iteration number required to solve the problem. (b) Speedup of the $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$.

$\mathcal{MG}_{Box}(1)$ for all $\kappa_2/\kappa_1$ values (see Fig. 8 also). Moreover, $\mathcal{MG}_{Box}(1)$ is more sensitive to the increased coefficient contrast, i.e., the convergence rate decreases with the increase of the coefficient contrast.

The number of iterations required to converge to the solution utilizing $\mathcal{MG}_S(L)$, $L \in \{1, 2, 3\}$, is shown in Fig. 10(a) in terms of the coefficient contrast. The performance of $\mathcal{MG}_{Box}(L)$ is shown in gray color as a reference. It can be concluded that the image-based multiscale multigrid solver generally performs better than the black-box multigrid solver regardless of

**Table 4**
The performance of $\mathcal{MG}_S(L)$ and $\mathcal{MG}_{Box}(L)$, $L \in \{1, 2, 3\}$, with the varying coefficient contrast. The iteration number required to solve the problem and the speedup are shown for both methods.

| $L$ | $\kappa_2/\kappa_1$ | $\mathcal{MG}_S$ | | $\mathcal{MG}_{Box}$ | |
|---|---|---|---|---|---|
| | | Iteration | Speedup | Iteration | Speedup |
| 1 | $10^1$ | 4 | 0.765 | 5 | 0.822 |
| | $10^2$ | 5 | 2.404 | 8 | 2.014 |
| | $10^3$ | 5 | 20.802 | 9 | 15.994 |
| | $10^4$ | 5 | 204.477 | 9 | 158.844 |
| 2 | $10^1$ | 5 | 0.768 | 6 | 0.895 |
| | $10^2$ | 13 | 1.413 | 20 | 1.052 |
| | $10^3$ | 103 | 1.584 | 184 | 1.060 |
| | $10^4$ | 871 | 1.606 | 1817 | 1.080 |
| 3 | $10^1$ | 6 | 0.851 | 6 | – |
| | $10^2$ | 15 | 1.390 | 22 | – |
| | $10^3$ | 112 | 1.583 | 193 | – |
| | $10^4$ | 952 | 1.607 | 1915 | – |

the multigrid depth. It is shown that $\mathcal{MG}_S(1)$ has the optimal performance among all the depths (i.e., $L = 1, 2$, and 3). The iteration number is nearly constant with increased $\kappa_2/\kappa_1$. The $\mathcal{MG}_S(2)$ and $\mathcal{MG}_S(3)$ require increased number of iterations, and the iteration number increases sublinearly with the coefficient contrast (i.e., the slopes are 0.76 and 0.75, respectively). However, $\mathcal{MG}_S(3)$ requires similar number of iterations to $\mathcal{MG}_S(2)$, in spite of the increased number of auxiliary grids. The number of iterations for both methods is also listed in Table 4. One can see that $\mathcal{MG}_S(L)$ needs significantly fewer number of iterations than $\mathcal{MG}_{Box}(L)$ at high coefficient contrasts. Moreover, this advantage becomes more pronounced with increased number auxiliary grids ($L$). For instance, $\mathcal{MG}_S(3)$ needs 963 fewer iterations than $\mathcal{MG}_{Box}(3)$ in the case of $\kappa_2/\kappa_1 = 10000$. This is the result of the image-based properties of the inter-grid operators as demonstrated in Section 6.1 (see Table 2).

The speedup of image-based multiscale multigrid solver is shown in Fig. 10(b) and Table 4. The performance of $\mathcal{MG}_{Box}(L)$, $L \in \{1, 2\}$, is shown in gray color as a reference. Here, we measure the speedup using Eq. (32), where the computer time of $\mathcal{MG}_{Box}(3)$ is taken as the reference. As noted earlier, we use the speedup instead of the actual computer time to limit the software and hardware ambiguities. It can be concluded that $\mathcal{MG}_S(L)$ generally gains larger speedup than $\mathcal{MG}_{Box}(L)$ at each $L$. Specifically, $\mathcal{MG}_S(1)$ gains the most speedup, which grows rapidly with the $\kappa_2/\kappa_1$ increase. The speedup of $\mathcal{MG}_S(2)$ and $\mathcal{MG}_S(3)$ is smaller and reaches 1.61, which is still significant numerical acceleration. It can also be observed that at the low coefficient contrast, e.g., $\kappa_2/\kappa_1 = 10$, $\mathcal{MG}_S(L)$, $L \in \{1, 2, 3\}$ consume longer computer time than $\mathcal{MG}_{Box}(3)$ (i.e., Speedup $< 1$). This is because formulating the inter-grid operators for the image-based multiscale multigrid solver (see Section 3.2) has increased complexity than that of $\mathcal{MG}_{Box}(3)$, which becomes an important factor at low $\kappa_2/\kappa_1$.

### 6.3. Image-based multiscale preconditioner

To examine the performance of the image-based multiscale preconditioner, we present the number of iterations needed for preconditioned conjugate gradient method. Specifically, we show the iteration number using preconditioners with the coarse component from different levels, i.e., $M_S^{-1}(L)$, $L \in \{1, 2, 3\}$ (see Fig. 6(b)). As an illustration, the results are also shown for varying coefficient contrasts, i.e., $\kappa_2/\kappa_1 \in \{10, 100, 1000, 10000\}$. The same set of problems are solved using the incomplete Cholesky preconditioner (IC), the Jacobi preconditioner (JA), and the black-box multigrid preconditioner ($\mathcal{MG}_{Box}(L)$, see Fig. 6(a)) as a comparison. The performance of conjugate gradient method with no preconditioning (CG) is shown as a reference. The performance is evaluated by the number of iterations, as well as the speedup (see Eq. (32)). Here, the computer time of CG is chosen as the reference ($t_{\text{ref}} = t_{\text{CG}}$). Again, the speedup is chosen over the actual computer time to limit the bias from the software implementation and the hardware architecture.

The eigenvalue distributions are shown in Fig. 11. The horizontal axis shows the eigenvalue indexes and the vertical axis shows the eigenvalues. The corresponding maximum and minimum eigenvalues are also included in Fig. 11. The eigenvalue distribution of $\boldsymbol{A}^0$ is shown in Fig. 11(a). The eigenvalues range between $8.8 \times 10^{-3}$ and $3.9 \times 10^4$ (see Fig. 11(a)), which is caused by the significant variance of material coefficients. The eigenvalues with JA are plotted in Fig. 11(b). The large eigenvalues are efficiently removed, however, all eigenvalues still scatter in a large range ($8.7 \times 10^{-6} - 1.5$). The eigenvalues with IC are plotted in Fig. 11(c), which are better distributed than JA and range between $1.2 \times 10^{-4}$ and 1.6 (see Fig. 11).

The eigenvalues of the system with $M_S^{-1}(1)$ preconditioning are shown in Fig. 11(d). It is observed that almost all eigenvalues are clustered around 1 and a few eigenvalues are scattered between 0.84 and 1.0, which indicates a well-conditioned system. The eigenvalue distributions with $M_S^{-1}(2)$ and $M_S^{-1}(3)$ preconditioning are shown in Figs. 11(e) and 11(f). The eigenvalues range between 0.19 and 1.0 for $L = 2, 3$. The matrix eigenvalues with $\mathcal{MG}_{Box}(L)$ preconditioning are similar to those

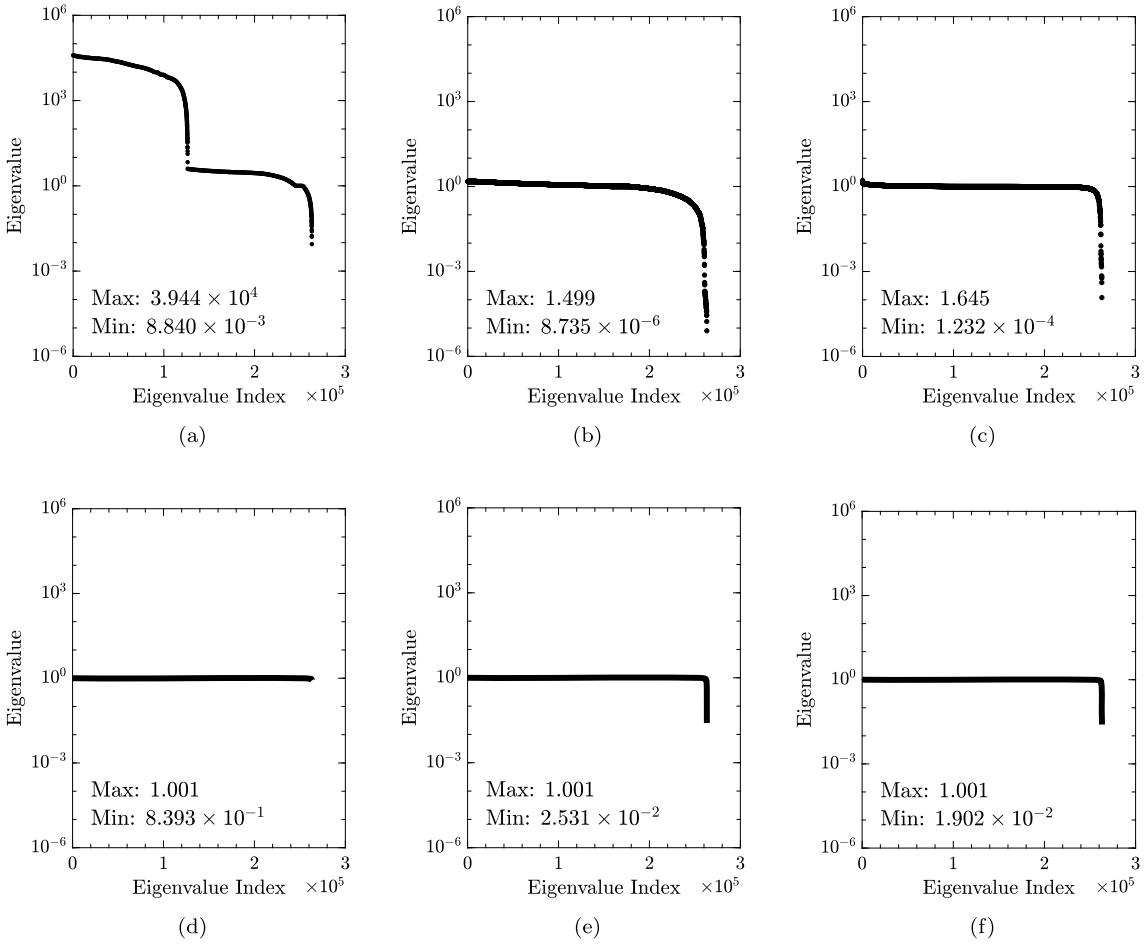


**Fig. 11.** The eigenvalue distribution of the coefficient matrices. The coefficient contrast is $\kappa_2/\kappa_1 = 10000$. All eigenvalues are displayed in a descending order. (a) No preconditioning. (b) Jacobi preconditioning. (c) IC preconditioning. (d) $M_S^{-1}(1)$ preconditioning. (e) $M_S^{-1}(2)$ preconditioning. (f) $M_S^{-1}(3)$ preconditioning.

of $M_S^{-1}(L)$ and are not displayed. The eigenvalues range between 0.98 and 1.0 with $L = 1$ and between $2.3 \times 10^{-2}$ and 1.0 for $L = 2, 3$. It can be concluded that both $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioners have close performance in reducing the range of eigenvalues at a certain $L$, while $\mathcal{MG}_{Box}(L)$ is slightly better in redistributing the eigenvalues. However, both $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioners have greatly reduced the range of eigenvalues compared to IC and JA.

The condition number of the coefficient matrix is shown in Fig. 12 as a function of varying coefficient contrast. It can be seen from Fig. 12(a) that the condition number remains almost constant with $M_S^{-1}(1)$ preconditioning despite the increase of $\kappa_2/\kappa_1$. The $M_S^{-1}(2)$ and $M_S^{-1}(3)$ have close performance, while both experience an increase of the condition number with the $\kappa_2/\kappa_1$ increase. Similar phenomenon can be observed for the $\mathcal{MG}_{Box}(L)$ preconditioners. The IC results in a better-conditioned system than JA in general (see Fig. 12(b)). However, both IC and JA only slightly decrease the condition number of the original system, while both $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioners have reduced the original condition number by orders of magnitude (see Fig. 12(a)). The actual condition numbers of all cases are listed in Table 5.

The performance of preconditioned conjugate gradient method for varying coefficient contrast using $M_S^{-1}(L)$, $\mathcal{MG}_{Box}(L)$ preconditioner ($L \in \{1, 2, 3\}$), IC, and JA preconditioners is shown in Fig. 13. Specifically, Fig. 13(a) displays that $M_S^{-1}(1)$ converges to the solution within 6 iterations despite large $\kappa_2/\kappa_1$ value. The $M_S^{-1}(2)$ and $M_S^{-1}(3)$ are more fragile than $M_S^{-1}(1)$ and need increased number of iterations with increased $\kappa_2/\kappa_1$. However, both preconditioners can readily reduce the number of CG iterations by more than 2 orders of magnitude. The $\mathcal{MG}_{Box}(L)$ preconditioner requires slightly fewer number of iterations than $M_S^{-1}(L)$ for all $L$ values (see Fig. 13(a) and Table 5). Both IC and JA require significantly more iterations than $M_S^{-1}(L)$ at all coefficient contrasts (see Fig. 13(b)). This observation is also consistent with the result of the condition number shown in Fig. 12(b) and Table 5.

The speedup of the preconditioned CG methods is shown in Fig. 14. It can be concluded that the $M_S^{-1}(L)$, with $L \in \{1, 2, 3\}$ generally gains larger speedup than IC and JA. The speedup of $M_S^{-1}(1)$ is the most prominent and increases from

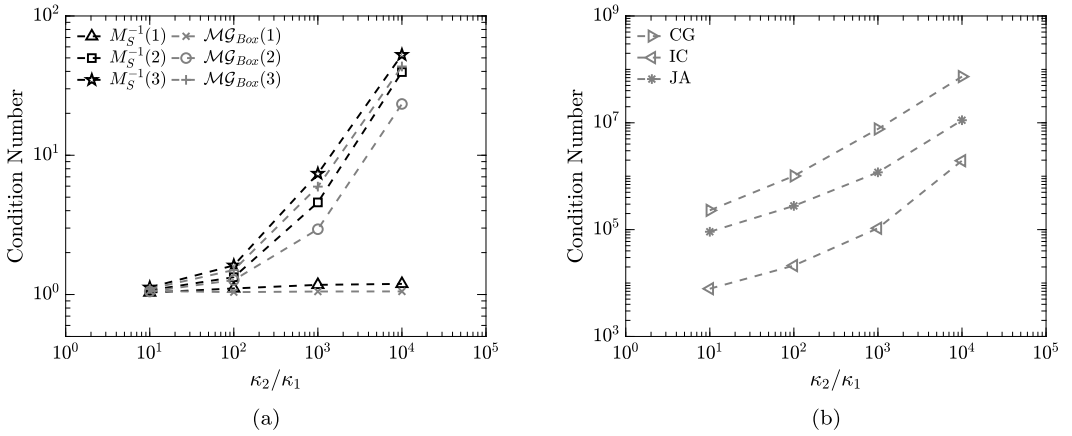


**Fig. 12.** The condition number of the coefficient matrix before and after preconditioning with varying coefficient contrasts. (a) The $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioners with $L \in \{1, 2, 3\}$. (b) The IC, JA preconditioners and CG. The condition numbers are also listed in Table 5.

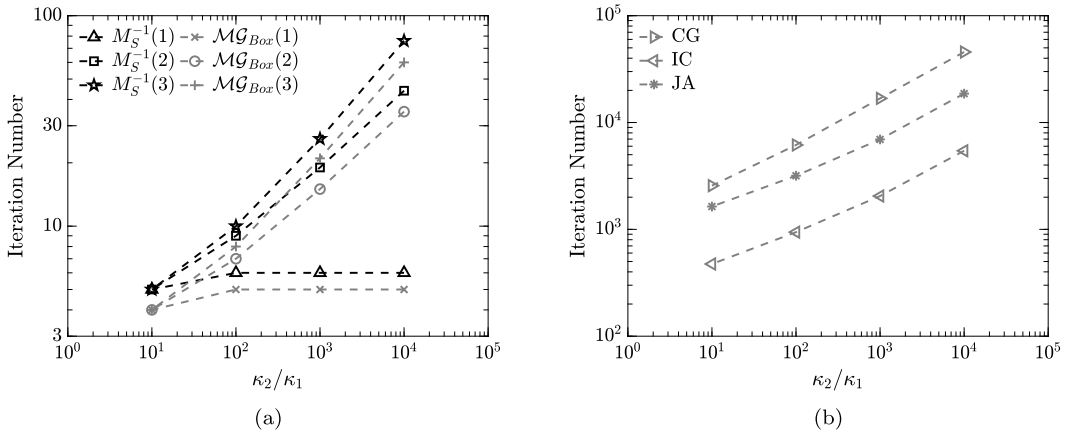


**Fig. 13.** The number of iterations required to converge to the solution using different preconditioners. The results are shown for a varying coefficient contrast. (a) The $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioners with $L \in \{1, 2, 3\}$. (b) The IC, JA preconditioners and CG. The number of iterations are also listed in Table 5.

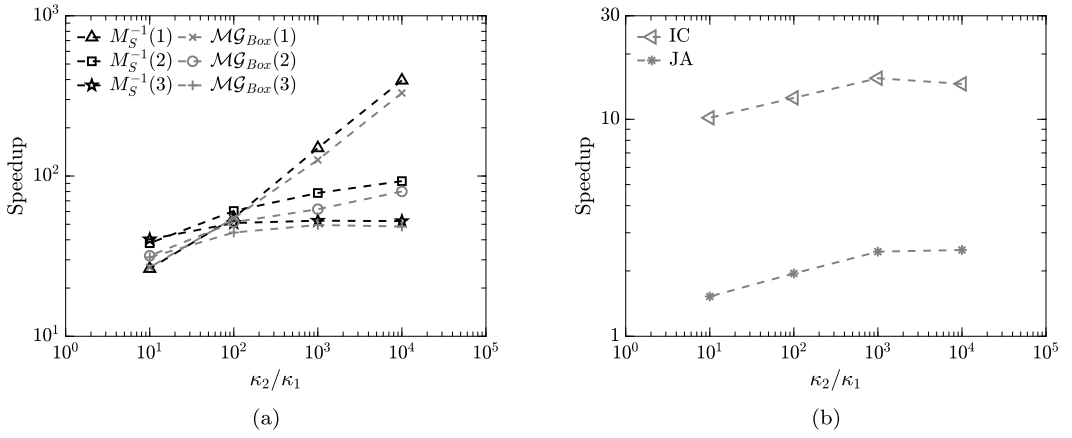


**Fig. 14.** The speedup of different preconditioners with respect to a varying coefficient contrast. (a) The $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioners with $L \in \{1, 2, 3\}$. (b) The IC, and JA preconditioners. The speedup is also listed in Table 5.

**Table 5**
The performance of $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioner, $L \in \{1, 2, 3\}$, with the varying coefficient contrasts. The condition number of the coefficient matrix after preconditioning, the iteration number required to solve the problem using preconditioned conjugate gradient method, and the speedup are listed for both methods.

| $L$ | $\kappa_2/\kappa_1$ | $M_S^{-1}(L)$ | | | $\mathcal{MG}_{Box}(L)$ | | |
|---|---|---|---|---|---|---|---|
| | | Condition number | Iteration | Speedup | Condition number | Iteration | Speedup |
| 1 | $10^1$ | 1.034 | 5 | 26.401 | 1.016 | 4 | 26.907 |
| | $10^2$ | 1.106 | 6 | 54.856 | 1.041 | 5 | 54.564 |
| | $10^3$ | 1.174 | 6 | 149.518 | 1.051 | 5 | 125.963 |
| | $10^4$ | 1.193 | 6 | 395.193 | 1.053 | 5 | 329.764 |
| 2 | $10^1$ | 1.073 | 5 | 38.108 | 1.051 | 4 | 31.781 |
| | $10^2$ | 1.332 | 9 | 60.519 | 1.270 | 7 | 51.548 |
| | $10^3$ | 4.586 | 19 | 78.369 | 2.943 | 15 | 62.061 |
| | $10^4$ | 39.554 | 44 | 92.958 | 23.347 | 35 | 80.141 |
| 3 | $10^1$ | 1.125 | 5 | 40.297 | 1.098 | 4 | 31.077 |
| | $10^2$ | 1.618 | 10 | 50.908 | 1.500 | 8 | 44.407 |
| | $10^3$ | 7.379 | 26 | 52.645 | 5.953 | 21 | 49.482 |
| | $10^4$ | 52.638 | 76 | 52.305 | 43.382 | 60 | 48.406 |

26.40 to 395.19 with the increase of the coefficient contrast (see also Table 5). The $M_S^{-1}(2)$ and $M_S^{-1}(3)$ have the most speedup when $\kappa_2/\kappa_1 < 100$, while this advantage is taken over by $M_S^{-1}(1)$ when $\kappa_2/\kappa_1 \geq 100$. This is because at moderate contrast computing $\left(\mathbb{A}^1\right)^{-1}$ for $M_S^{-1}(1)$ dominates the computation, while computing $\left(\mathbb{A}^2\right)^{-1}$ for $M_S^{-1}(2)$ and $\left(\mathbb{A}^3\right)^{-1}$ for $M_S^{-1}(3)$ are less demanding. However, at the high contrast, the iteration process dominates for $M_S^{-1}(2)$ and $M_S^{-1}(3)$ while it remains constant for $M_S^{-1}(1)$. The $\mathcal{MG}_{Box}(L)$ preconditioners gain less speedup than $M_S^{-1}(L)$ for all $L$. This is because the $\mathcal{MG}_{Box}(L)$ preconditioners include smoothing steps on all the intermediate levels (see Fig. 6(a)), which make them more expensive than $M_S^{-1}(L)$. In addition, the IC exhibits less speedup than $M_S^{-1}(L)$ and $\mathcal{MG}_{Box}(L)$ preconditioners, which ranges between 10.16 and 14.56. The JA is the slowest and its speedup ranges between 1.52 and 2.50. Again, all the speedup data are listed in Table 5 for easy reference.

### 6.4. Image-based reduced order model

To examine the performance of the IROM, we show the upscaled solution $\boldsymbol{u}_S(L)$, where $L \in \{1, 2, 3\}$. The coefficient contrast is kept constant, $\kappa_2/\kappa_1 = 10000$. The solution fields from the IROM are computed following the descriptions in Section 5. As a comparison, we also display the solution on the finest grid ($\boldsymbol{u}^0 = \mathbb{U}^0$), which is computed by solving Eq. (4) using a direct LDL$^\intercal$ solver [107].

The solution field on the finest grid ($\boldsymbol{u}^0 = \mathbb{U}^0$) is shown in Fig. 15(a). The solution fields from the IROM ($\boldsymbol{u}_S(L)$) are shown in Figs. 15(b) to 15(d), respectively. It can be seen that all of the $\boldsymbol{u}_S(L)$, with $L \in \{1, 2, 3\}$ are very close to $\boldsymbol{u}^0$. Fig. 16 shows a detailed comparison among the solution fields from IROM ($\boldsymbol{u}_S(L)$) and the fine grid solution ($\boldsymbol{u}^0$). In particular, Figs. 16(a) and 16(b) display the $\boldsymbol{u}_S(L)$, $L \in \{1, 2, 3\}$ along the centered horizontal line ($y = 3.84$ μm) and the centered vertical line ($x = 3.84$ μm), respectively. The $\boldsymbol{u}^0$ is shown in both Figs. 16(a) and 16(b) as a reference. It can be seen that all of the $\boldsymbol{u}_S(L)$, with $L \in \{1, 2, 3\}$ are generally in good agreement with $\boldsymbol{u}^0$. There is increased error in regions where the solution becomes highly oscillatory.

To quantitatively compare the solution fields, we compute the $L^2$-norm of the error

$$||\varepsilon||_2 = \sqrt{\frac{\int_\Omega \left(u^0(\boldsymbol{x}) - \tilde{u}(\boldsymbol{x})\right)^2 \, \mathrm{d}\Omega}{\int_\Omega \left(u^0(\boldsymbol{x})\right)^2 \, \mathrm{d}\Omega}} \times 100\%, \tag{33}$$

where $u^0(\boldsymbol{x})$ is the scalar solution field interpolated from $\boldsymbol{u}^0$ using the finite element shape functions, $\tilde{u}(\boldsymbol{x})$ denotes the scalar solution field interpolated either from the IROM (i.e., $\boldsymbol{u}_S(L)$ in Eq. (28)) or the SVB images (i.e., $\mathbb{U}^L$ in Eq. (4)). Fig. 17 shows the $L^2$-norm error as a function of the reduced level (i.e., $L$). As can be seen from Fig. 17, the $\mathbb{U}^L$ generally preserves the solution characteristics well despite the large data compression. Typically, $\mathbb{U}^3$ introduces only 9.01% error after reducing the number of DOFs 64 times (see Table 1). However the IROM, $\boldsymbol{u}_S(L)$ (from Eq. (28)) yields less error than $\mathbb{U}^L$ (from Eq. (4)) with the same compression ratio. As shown in Fig. 17, the $\boldsymbol{u}_S(1)$ only introduces 0.16% error and $\boldsymbol{u}_S(3)$ controls the error within 6.72%. This implies that the IROM restore detailed solution characteristics that are filtered during direct data compression. For example, in the case of $L = 2$, the solution $\boldsymbol{u}_S(2)$ improves the solution $\mathbb{U}^2$ by 5.26%. Note that $\boldsymbol{u}_S(L)$ can serve as an initial guess to accelerate the multigrid or any other iterative solvers.

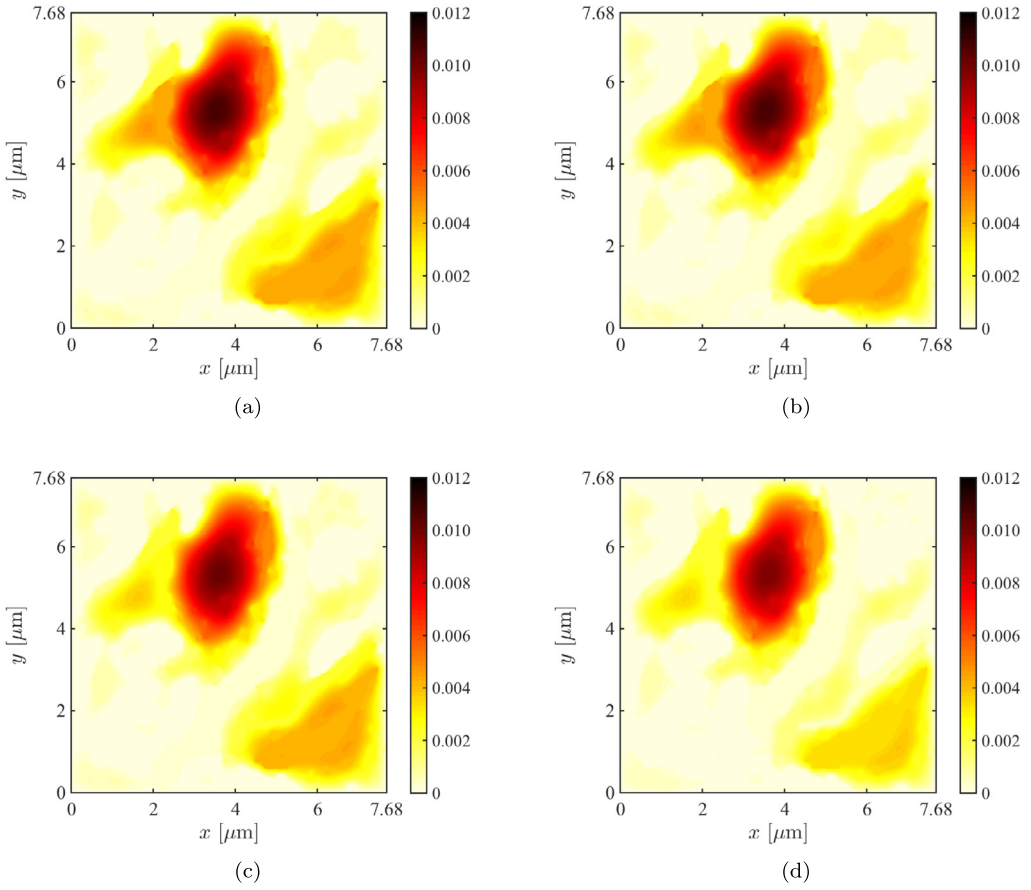


**Fig. 15.** The extrapolated solution $\boldsymbol{u}_S(L)$ with the coefficient contrast $\kappa_2/\kappa_1 = 10000$. (a) The solution on the 0th SVB-LOD image. The solution field from IROM, (b) $\boldsymbol{u}_S(1)$, (c) $\boldsymbol{u}_S(2)$, (d) $\boldsymbol{u}_S(3)$. (For interpretation of the colors in the figure, the reader is referred to the web version of this article.)

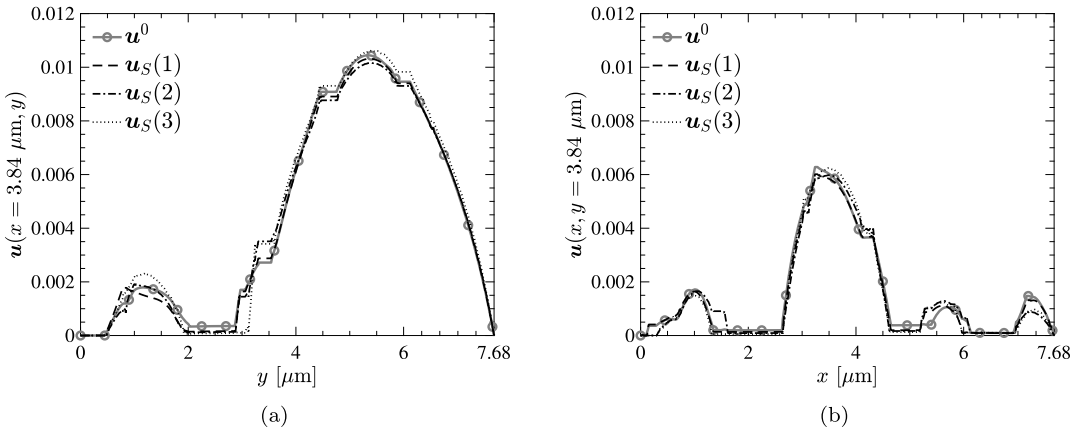


**Fig. 16.** The comparison between the fine grid solution and solutions from the IROM. The coefficient contrast is $\kappa_2/\kappa_1 = 10000$. (a) $x = 3.84$ μm. (b) $y = 3.84$ μm.

## 7. Conclusion

In this work, we propose a novel image-based multiscale multigrid solver, preconditioner, and the reduced order model. An image-based inter-grid operator is developed via incorporating the microstructural information from the multiresolution scheme (i.e., data driven SVB model). A two-stage approach is established for computing the inter-grid operator, which pre-

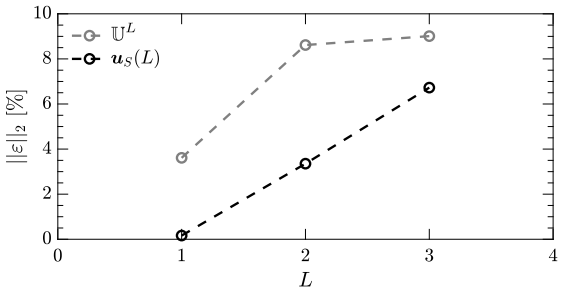


**Fig. 17.** The $L^2$-norm error of the coarse image solutions and the IROM.

serves the flux along the grid lines. This multigrid solver is robust for extreme coefficient contrasts and exhibits near-optimal convergence rate.

A new image-based multiscale preconditioner is developed utilizing the coarse SVB image and the image-based inter-grid operator. Thus, relaxations on the intermediate grids are omitted, resulting in a simpler formulation and a lighter computational demand per iteration. This preconditioner shows high efficiency for ill-conditioned systems, and exhibits greatly improved performance compared to traditional preconditioners such as the Jacobi and/or the Incomplete Cholesky.

The IROM reduces the number of DOFs by converting the fine level problem to a coarse grid one. It is demonstrated that the IROM reduces the error from the geometrical coarse solutions and restores detailed solution characteristics that are filtered due to the direct data compression.

This work opens a new possibility for solving a system of linear equations associated with data heterogeneity, which is a fundamental problem in a large array of engineering and science disciplines. Moreover, the multiscale image-based approach is applicable to other fields such as uncertainty quantification, data compression, and adaptive multiscale modeling. The development of the 3D image-based multiscale approach with a larger system size and its parallelization are both important future directions.

## Declaration of competing interest

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

## Acknowledgement

This work was supported by the Department of Energy, National Nuclear Security Administration, under the award No. DENA0002377 as part of the Predictive Science Academic Alliance Program II. We would like to give special thanks to Dr. Waad Subber for his help in the development of the multiscale preconditioner.

## Appendix A. The flux conservation by the prolongation operator

In Section 3.2, an approximate error, $\tilde{\boldsymbol{e}}^l$, on the fine grid is obtained from the microstructural information from the SVB multiscale image (see Eq. (12a)). In this section, we prove that Eqs. (14) to (20) preserves the continuity of the flux in a weak (integral) form.

The continuity of normal (x-direction) flux along the central vertical interface (see Fig. 18(a)) yields

$$\lim_{x\to x_i^-}\int_{y_{j-1}}^{y_{j+1}} -\kappa^l(\boldsymbol{x})\frac{\partial\tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\,\mathrm{d}y = \lim_{x\to x_i^+}\int_{y_{j-1}}^{y_{j+1}} -\kappa^l(\boldsymbol{x})\frac{\partial\tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\,\mathrm{d}y, \tag{34}$$

where the left and right integrals are approximated by

$$\begin{aligned}
\lim_{x\to x_i^-}\int_{y_{j-1}}^{y_{j+1}} -\kappa^l(\boldsymbol{x})\frac{\partial\tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\,\mathrm{d}y &\approx -(\kappa^l_{i-1,j-1}+\kappa^l_{i-1,j})\left.\frac{\partial\tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\right|_{x_{i-1/2},y_j}\Delta h^l,\\
\lim_{x\to x_i^+}\int_{y_{j-1}}^{y_{j+1}} -\kappa^l(\boldsymbol{x})\frac{\partial\tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\,\mathrm{d}y &\approx -(\kappa^l_{i,j-1}+\kappa^l_{i,j})\left.\frac{\partial\tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\right|_{x_{i+1/2},y_j}\Delta h^l.
\end{aligned} \tag{35}$$

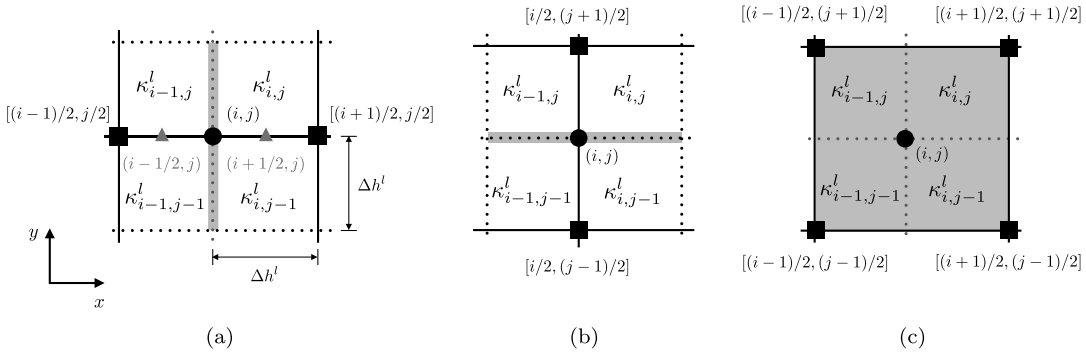


**Fig. 18.** The schematic of computing $\tilde{\boldsymbol{e}}^l$ for the image-based inter-grid operator. (a) The flux is continuous across the vertical interface shown in gray color. The ▲ marks the virtual nodes at the center of fine grid lines (see Eqs. (35) and (36)). (b) The flux is continuous across the horizontal interface shown in gray color. (c) The flux is preserved inside of the coarse cell.

Here, $\Delta h^l$ is the grid spacing along both $x$ and $y$ directions and the half indexes (i.e., $x_{i+1/2}$ and $x_{i-1/2}$) identify the virtual points located at the center of the fine grid lines (see Fig. 18). The derivative in Eq. (35) is evaluated using the forward Euler approximation,

$$\left.\frac{\partial \tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\right|_{x_i, y_j} \approx \frac{\tilde{\boldsymbol{e}}^l_{i+1/2,j} - \tilde{\boldsymbol{e}}^l_{i-1/2,j}}{\Delta h^l}. \tag{36}$$

Substituting Eqs. (35) and (36) into Eq. (34) yields

$$\tilde{\boldsymbol{e}}^l_{i,j} = \frac{(\kappa^l_{i-1,j-1} + \kappa^l_{i-1,j})\tilde{\boldsymbol{e}}^l_{i-1,j} + (\kappa^l_{i,j-1} + \kappa^l_{i,j})\tilde{\boldsymbol{e}}^l_{i+1,j}}{\kappa^l_{i-1,j-1} + \kappa^l_{i-1,j} + \kappa^l_{i,j-1} + \kappa^l_{i,j}}, \tag{37}$$

where $\tilde{\boldsymbol{e}}^l_{i-1,j} = \boldsymbol{e}^{l+1}_{(i-1)/2,j/2}$ and $\tilde{\boldsymbol{e}}^l_{i+1,j} = \boldsymbol{e}^{l+1}_{(i+1)/2,j/2}$, which are computed from the coarse grid value (see Eq. (14)), and the coefficients of $\tilde{\boldsymbol{e}}^l_{i-1,j}$ and $\tilde{\boldsymbol{e}}^l_{i+1,j}$ are identical to that in Eq. (15). To this end, Eq. (37) is equivalent to Eq. (15), which preserves continuity of the flux across the vertical interface in Fig. 18(a). Similarly, we can prove that Eq. (17) satisfies

$$\lim_{y\to y_j^-} \int_{x_{i-1}}^{x_{i+1}} -\kappa^l(\boldsymbol{x}) \frac{\partial \tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial y}\, \mathrm{d}x = \lim_{y\to y_j^+} \int_{x_{i-1}}^{x_{i+1}} -\kappa^l(\boldsymbol{x}) \frac{\partial \tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial y}\, \mathrm{d}x, \tag{38}$$

which is the continuity condition for y-directional normal flux (see Fig. 18(b)).

For the fine grid nodes located at the center of the coarse cell, we require the preservation of the flux inside of a coarse cell (see Fig. 18(c)). This condition can be mathematically described in a weak form

$$\int_{\Omega_e^{l+1}} \nabla \cdot (\kappa^l(\boldsymbol{x}) \nabla \tilde{\boldsymbol{e}}^l(\boldsymbol{x}))\, \mathrm{d}\Omega = 0, \tag{39}$$

where the integral is evaluated locally in a coarse cell, $\Omega_e^{l+1}$, centered by a node $(i, j)$. After applying the Divergence theorem, we have

$$\oint_{\partial\Omega_e^{l+1}} \kappa^l(\boldsymbol{x}) \nabla \tilde{\boldsymbol{e}}^l(\boldsymbol{x}) \cdot \boldsymbol{n}\, \mathrm{d}\partial\Omega = 0, \tag{40}$$

where $\boldsymbol{n}$ is the normal vector. Equation (40) is equivalent to

$$\begin{aligned} &\lim_{x\to x_{i+1}^-} \int_{y_{j-1}}^{y_{j+1}} \kappa^l(\boldsymbol{x}) \frac{\partial \tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\, \mathrm{d}y - \lim_{x\to x_{i-1}^+} \int_{y_{j-1}}^{y_{j+1}} \kappa^l(\boldsymbol{x}) \frac{\partial \tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial x}\, \mathrm{d}y + \\ &\lim_{y\to y_{j+1}^-} \int_{x_{i-1}}^{x_{i+1}} \kappa^l(\boldsymbol{x}) \frac{\partial \tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial y}\, \mathrm{d}x - \lim_{y\to y_{j-1}^+} \int_{x_{i-1}}^{x_{i+1}} \kappa^l(\boldsymbol{x}) \frac{\partial \tilde{\boldsymbol{e}}^l(\boldsymbol{x})}{\partial y}\, \mathrm{d}y = 0. \end{aligned} \tag{41}$$

Here, we approximate the line integral and the derivatives are from Eqs. (35) and (36), respectively. After substituting and rearranging, we get

$$\begin{aligned}\tilde{\boldsymbol{e}}^l_{i,j} = &\frac{\kappa^l_{i,j}+\kappa^l_{i,j-1}}{2d^l_{i,j}}\tilde{\boldsymbol{e}}^l_{i+1,j} + \frac{\kappa^l_{i-1,j}+\kappa^l_{i,j}}{2d^l_{i,j}}\tilde{\boldsymbol{e}}^l_{i,j+1} \\ &+ \frac{\kappa^l_{i-1,j}+\kappa^l_{i-1,j-1}}{2d^l_{i,j}}\tilde{\boldsymbol{e}}^l_{i-1,j} + \frac{\kappa^l_{i-1,j-1}+\kappa^l_{i,j-1}}{2d^l_{i,j}}\tilde{\boldsymbol{e}}^l_{i,j-1},\end{aligned} \tag{42}$$

where

$$\begin{aligned}\tilde{\boldsymbol{e}}^l_{i+1,j} &= N^l_{i+1,j}\boldsymbol{e}^{l+1}_{(i+1)/2,(j+1)/2} + S^l_{i+1,j}\boldsymbol{e}^{l+1}_{(i+1)/2,(j-1)/2}, \\ \tilde{\boldsymbol{e}}^l_{i-1,j} &= N^l_{i-1,j}\boldsymbol{e}^{l+1}_{(i-1)/2,(j+1)/2} + S^l_{i-1,j}\boldsymbol{e}^{l+1}_{(i-1)/2,(j-1)/2}, \\ \tilde{\boldsymbol{e}}^l_{i,j+1} &= W^l_{i,j+1}\boldsymbol{e}^{l+1}_{(i-1)/2,(j+1)/2} + E^l_{i,j+1}\boldsymbol{e}^{l+1}_{(i+1)/2,(j+1)/2}, \\ \tilde{\boldsymbol{e}}^l_{i,j-1} &= W^l_{i,j-1}\boldsymbol{e}^{l+1}_{(i-1)/2,(j-1)/2} + E^l_{i,j-1}\boldsymbol{e}^{l+1}_{(i+1)/2,(j-1)/2}.\end{aligned} \tag{43}$$

The error terms in Eq. (43) are obtained from Eqs. (15) and (17), respectively. After substituting Eq. (43) into Eq. (42) and rearranging, Eq. (42) is identical to Eq. (19), which implies conservation of the flux in a coarse element.

## Appendix B. Convergence rate estimates

In this section, we show that Eq. (23) is the lower bound of the actual convergence rate [32,104]. To do this, we summarize one iteration of the V-cycle multigrid (see Algorithm 1) as

$$\boldsymbol{u}^l_{n+1} = \boldsymbol{\mathcal{M}}^l\boldsymbol{u}^l_n + \boldsymbol{\mathcal{G}}^l\boldsymbol{Q}^l, \tag{44}$$

where the subscript denotes the iteration number, $\boldsymbol{\mathcal{M}}^l$ and $\boldsymbol{\mathcal{G}}^l \in \mathbb{R}^{M^l\times M^l}$ are iteration operators according to the multigrid algorithm (see Algorithm 1). Typically, in a two-grid setting, the definitions of $\boldsymbol{\mathcal{M}}^l$ and $\boldsymbol{\mathcal{G}}^l$ are

$$\begin{aligned}\boldsymbol{\mathcal{M}}^l &= \boldsymbol{S}^l\left(\boldsymbol{I}^l - \boldsymbol{P}^l_{l+1}\left(\boldsymbol{A}^{l+1}\right)^{-1}\boldsymbol{R}^{l+1}_l\boldsymbol{A}^l\right)\boldsymbol{S}^l, \\ \boldsymbol{\mathcal{G}}^l &= \boldsymbol{S}^l\left(\boldsymbol{I}^l - \boldsymbol{P}^l_{l+1}\left(\boldsymbol{A}^{l+1}\right)^{-1}\boldsymbol{R}^{l+1}_l\boldsymbol{A}^l\right)\boldsymbol{F}^l + \boldsymbol{S}^l\boldsymbol{P}^l_{l+1}\left(\boldsymbol{A}^{l+1}\right)^{-1}\boldsymbol{R}^{l+1}_l + \boldsymbol{F}^l,\end{aligned} \tag{45}$$

where $\boldsymbol{I}^l \in \mathbb{R}^{M^l\times M^l}$ is the identity matrix, $\boldsymbol{S}^l$ and $\boldsymbol{F}^l \in \mathbb{R}^{M^l\times M^l}$ are the smoothers from the smoothing algorithm (see Algorithm 1). Typically, the Gauss-Seidel method yields

$$\begin{aligned}\boldsymbol{S}^l &= \left(-\left(\mathrm{diag}\left(\boldsymbol{A}^l\right) + \mathrm{tril}\left(\boldsymbol{A}^l\right)\right)^{-1}\mathrm{triu}\left(\boldsymbol{A}^l\right)\right)^{\nu}, \\ \boldsymbol{F}^l &= \left(\sum_{n=0}^{\nu-1}\left(-\left(\mathrm{diag}\left(\boldsymbol{A}^l\right) + \mathrm{tril}\left(\boldsymbol{A}^l\right)\right)^{-1}\mathrm{triu}\left(\boldsymbol{A}^l\right)\right)^n\right)\left(\mathrm{diag}\left(\boldsymbol{A}^l\right) + \mathrm{tril}\left(\boldsymbol{A}^l\right)\right)^{-1},\end{aligned} \tag{46}$$

where diag$(\cdot)$, tril$(\cdot)$, and triu$(\cdot)$ denote the diagonal, upper triangular, and lower triangular part of the matrix, respectively.

As a stationary iterative method, the exact solution is unchanged by the iteration

$$\boldsymbol{u}^l = \boldsymbol{\mathcal{M}}^l\boldsymbol{u}^l + \boldsymbol{\mathcal{G}}^l\boldsymbol{Q}^l, \tag{47}$$

where $\boldsymbol{u}^l$ is the exact solution (see Eq. (8)). Subtracting Eq. (44) from Eq. (47) yields

$$\boldsymbol{\epsilon}^l_{n+1} = \boldsymbol{\mathcal{M}}^l\boldsymbol{\epsilon}^l_n, \tag{48}$$

where $\boldsymbol{\epsilon}^l_n$ and $\boldsymbol{\epsilon}^l_{n+1}$ are the error vectors after the $n$th and the $(n+1)$th iterations, respectively. Taking the $L^2$-norm on both sides of Eq. (48) gives

$$||\boldsymbol{\epsilon}^l_{n+1}|| = ||\boldsymbol{\mathcal{M}}^l\boldsymbol{\epsilon}^l_n|| \le \lambda_{\max}\left(\boldsymbol{\mathcal{M}}^l\right)||\boldsymbol{\epsilon}^l_n||, \tag{49}$$

where $\lambda_{\max}(\cdot)$ denotes the spectrum radius [105,106]. Equation (49) is equivalent to

$$\lambda_{\max}\left(\boldsymbol{\mathcal{M}}^l\right) \ge \frac{||\boldsymbol{\epsilon}^l_{n+1}||}{||\boldsymbol{\epsilon}^l_n||}. \tag{50}$$

Therefore, we have

$$\Psi = -\log_{10}\left(\lambda_{\max}\left(\mathcal{M}^l\right)\right) \leq -\log_{10}\left(\frac{||\boldsymbol{\epsilon}^l_{n+1}||}{||\boldsymbol{\epsilon}^l_n||}\right), \tag{51}$$

which sets the lower bound of the convergence rate in terms of digits per iteration.

## References


[1] J.S. Vetter, Contemporary High Performance Computing: From Petascale Toward Exascale, Chapman and Hall/CRC, 2013.
[2] E. Strohmaier, J. Dongarra, H. Simon, M. Meuer, H. Meuer, The top500 supercomputer list, https://www.top500.org/, 2018.
[3] A. Bandyopadhyay, R. Pati, S. Sahu, F. Peper, D. Fujita, Massively parallel computing on an organic molecular layer, Nat. Phys. 6 (5) (2010) 369.
[4] M. Mosby, K. Matouš, Hierarchically parallel coupled finite strain multiscale solver for modeling heterogeneous layers, Int. J. Numer. Methods Eng. 102 (3–4) (2015) 748–765.
[5] J.C. Dietrich, S. Tanaka, J.J. Westerink, C.N. Dawson, R.A. Luettich, M. Zijlema, L.H. Holthuijsen, J.M. Smith, L.G. Westerink, H.J. Westerink, Performance of the unstructured-mesh, SWAN+ ADCIRC model in computing hurricane waves and surge, J. Sci. Comput. 52 (2) (2012) 468–497.
[6] P.F. Fischer, K. Heisey, M. Min, Scaling limits for PDE-based simulation, in: 22nd AIAA Computational Fluid Dynamics Conference, 2015, pp. 3049–3058.
[7] K. Matouš, M.G.D. Geers, V.G. Kouznetsova, A. Gillman, A review of predictive nonlinear theories for multiscale modeling of heterogeneous materials, J. Comput. Phys. 330 (2017) 192–220.
[8] E. Weinan, B. Engquist, X. Li, W. Ren, E. Vanden-Eijnden, Heterogeneous multiscale methods: a review, Commun. Comput. Phys. 2 (3) (2007) 367–450.
[9] A. Brandt, Multiscale scientific computation: review 2001, in: Multiscale and Multiresolution Methods, Springer, 2002, pp. 3–95.
[10] T. Hou, Y. Efendiev, H. Tchelepi, L.J. Durlofsky, Multiscale simulation framework for coupled fluid flow and mechanical deformation, Tech. Rep., California Institute of Technology/Stanford University/Texas A and M University, Pasadena, CA (United States)/CA (United States)/College Station, TX (United States), 2016.
[11] M. Ţene, Y. Wang, H. Hajibeygi, Adaptive algebraic multiscale solver for compressible flow in heterogeneous porous media, J. Comput. Phys. 300 (2015) 679–694.
[12] M. Mosby, K. Matouš, Computational homogenization at extreme scales, Extr. Mech. Lett. 6 (2016) 68–74.
[13] S. Yip, M.P. Short, Multiscale materials modelling at the mesoscale, Nat. Mater. 12 (9) (2013) 774–777.
[14] P.D. Dans, J. Walther, H. Gómez, M. Orozco, Multiscale simulation of DNA, Curr. Opin. Struct. Biol. 37 (2016) 29–45.
[15] C. Obiol-Pardo, J. Gomis-Tena, F. Sanz, J. Saiz, M. Pastor, A multiscale simulation system for the prediction of drug-induced cardiotoxicity, J. Chem. Inf. Model. 51 (2) (2011) 483–492.
[16] S. Gur, T. Danielson, Q. Xiong, C. Hin, S. Pannala, G. Frantziskonis, A. Savara, C.S. Daw, Wavelet-based surrogate time series for multiscale simulation of heterogeneous catalysis, Chem. Eng. Sci. 144 (2016) 165–175.
[17] M. Andersson, J. Yuan, B. Sundén, Review on modeling development for multiscale chemical reactions coupled transport phenomena in solid oxide fuel cells, Appl. Energy 87 (5) (2010) 1461–1476.
[18] A. Lotfi, Combination of epsilon and Ritz methods with multiscaling basis for solving a class of fractional optimal control problems, J. Comput. Phys. 366 (2018) 107–119.
[19] Z. Yi, P.H. Bauer, Adaptive multiresolution energy consumption prediction for electric vehicles, IEEE Trans. Veh. Technol. 66 (11) (2017) 10515–10525.
[20] E. Weinan, Principles of Multiscale Modeling, Cambridge University Press, 2011.
[21] P. Decaudin, F. Neyret, Volumetric billboards, in: Computer Graphics Forum, vol. 28, Wiley Online Library, 2009, pp. 2079–2089.
[22] X. Décoret, F. Durand, F.X. Sillion, J. Dorsey, Billboard clouds for extreme model simplification, in: ACM Transactions on Graphics (TOG), vol. 22, ACM, 2003, pp. 689–696.
[23] D. Luebke, M. Reddy, J.D. Cohen, A. Varshney, B. Watson, R. Huebner, Level of Detail for 3D Graphics, Morgan Kaufmann Publishers, 2003.
[24] S. Behrendt, C. Colditz, O. Franzke, J. Kopf, O. Deussen, Realistic real-time rendering of landscapes using billboard clouds, in: Computer Graphics Forum, vol. 24, Wiley Online Library, 2005, pp. 507–516.
[25] D. Yushu, S. Lee, K. Matouš, Sharp volumetric billboard based characterization and modeling of complex 3D Ni/Al high energy ball milled composites, Mech. Mater. 108 (2017) 93–106.
[26] T.J.R. Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Courier Corporation, 2012.
[27] J. Stoer, R. Bulirsch, Introduction to Numerical Analysis, vol. 12, Springer Science and Business Media, 2013.
[28] S.C. Chapra, R.P. Canale, Numerical Methods for Engineers, vol. 2, McGraw-Hill, New York, 1988.
[29] A. Hadjidimos, Successive overrelaxation (SOR) and related methods, J. Comput. Appl. Math. 123 (1) (2000) 177–199.
[30] T.A. Straeter, On the extension of the Davidon-Broyden class of rank one, quasi-Newton minimization methods to an infinite dimensional Hilbert space with applications to optimal control problems, Ph.D. thesis, North Carolina State University at Raleigh, 1971.
[31] Y. Saad, M.H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 7 (3) (1986) 856–869.
[32] U. Trottenberg, C.W. Oosterlee, A. Schuller, Multigrid, Academic Press, 2000.
[33] K. Stüben, U. Trottenberg, Multigrid methods: fundamental algorithms, model problem analysis and applications, in: Multigrid Methods, Springer, 1982, pp. 1–176.
[34] S.F. McCormick, Multigrid Methods, Society for Industrial and Applied Mathematics, 1987.
[35] H.K. Versteeg, W. Malalasekera, An Introduction to Computational Fluid Dynamics: The Finite Volume Method, Pearson Education, 2007.
[36] K. Stüben, A review of algebraic multigrid, J. Comput. Appl. Math. 128 (1) (2001) 281–309.
[37] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Math. Comput. 31 (138) (1977) 333–390.
[38] A. Brandt, N. Dinar, Multigrid solutions to elliptic flow problems, in: Numerical Methods for Partial Differential Equations, Elsevier, 1979, pp. 53–147.
[39] A. Brandt, S. McCoruick, J. Huge, Algebraic multigrid (AMG) for sparse matrix equations, in: Sparsity and Its Applications, 1985, p. 257.
[40] C. Lubich, A. Ostermann, Multi-grid dynamic iteration for parabolic equations, BIT Numer. Math. 27 (2) (1987) 216–234.
[41] D. Drikakis, O.P. Iliev, D.P. Vassileva, A nonlinear multigrid method for the three-dimensional incompressible Navier–Stokes equations, J. Comput. Phys. 146 (1) (1998) 301–321.
[42] D.J. Mavriplis, Multigrid solution of the two-dimensional Euler equations on unstructured triangular meshes, AIAA J. 26 (7) (1988) 824–831.
[43] J.C. Osborn, R. Babich, J. Brannick, R.C. Brower, M.A. Clark, S.D. Cohen, C. Rebbi, Multigrid solver for clover fermions, arXiv preprint, arXiv:1011.2775.
[44] A. Frommer, K. Kahl, S. Krieg, B. Leder, M. Rottmann, Adaptive aggregation-based domain decomposition multigrid for the lattice Wilson–Dirac operator, SIAM J. Sci. Comput. 36 (4) (2014) A1581–A1608.
[45] P. Vaněk, J. Mandel, M. Brezina, Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems, Computing 56 (3) (1996) 179–196.

[46] M. Esmaily, L. Jofre, A. Mani, G. Iaccarino, A scalable geometric multigrid solver for nonsymmetric elliptic systems with application to variable-density flows, J. Comput. Phys. 357 (2018) 142–158.
[47] A. Borzì, G. Borzì, Algebraic multigrid methods for solving generalized eigenvalue problems, Int. J. Numer. Methods Eng. 65 (8) (2006) 1186–1196.
[48] D. Kushnir, M. Galun, A. Brandt, Efficient multilevel eigensolvers with applications to data analysis tasks, IEEE Trans. Pattern Anal. Mach. Intell. 32 (8) (2010) 1377.
[49] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, H. van der Vorst, Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide, Society for Industrial and Applied Mathematics, 2000.
[50] S.G. Nash, A multigrid approach to discretized optimization problems, Optim. Methods Softw. 14 (1–2) (2000) 99–116.
[51] T. Dreyer, B. Maar, V. Schulz, Multigrid optimization in applications, J. Comput. Appl. Math. 120 (1–2) (2000) 67–84.
[52] C.A. Bouman, K.D. Sauer, Nonlinear multigrid methods of optimization in Bayesian tomographic image reconstruction, in: Neural and Stochastic Methods in Image and Signal Processing, vol. 1766, International Society for Optics and Photonics, 1992, pp. 296–307.
[53] M. Galun, R. Basri, I. Yavneh, Review of methods inspired by algebraic-multigrid for data and image analysis applications, Numer. Math., Theory Methods Appl. 8 (2) (2015) 283–312.
[54] S.B. Hazra, Multigrid one-shot method for aerodynamic shape optimization, SIAM J. Sci. Comput. 30 (3) (2008) 1527–1547.
[55] C. Nita, S. Vandewalle, J. Meyers, Multigrid optimization for DNS-based optimal control in turbulent channel flows, J. Comput. Phys. 366 (2018) 14–32.
[56] J.E. Dendy Jr., Black box multigrid, J. Comput. Phys. 48 (3) (1982) 366–386.
[57] J.E. Dendy Jr., J.D. Moulton, Black box multigrid with coarsening by a factor of three, Numer. Linear Algebra Appl. 17 (2–3) (2010) 577–598.
[58] P.M. De Zeeuw, Matrix-dependent prolongations and restrictions in a blackbox multigrid solver, J. Comput. Appl. Math. 33 (1) (1990) 1–27.
[59] J.E. Dendy Jr., Two multigrid methods for three-dimensional problems with discontinuous and anisotropic coefficients, SIAM J. Sci. Stat. Comput. 8 (5) (1987) 673–685.
[60] P. Vaněk, Fast multigrid solver, Appl. Math. 40 (1) (1995) 1–20.
[61] P. Vaněk, M. Brezina, Nearly optimal convergence result for multigrid with aggressive coarsening and polynomial smoothing, Appl. Math. 58 (4) (2013) 369–388.
[62] P. Vaněk, M. Brezina, J. Mandel, Convergence of algebraic multigrid based on smoothed aggregation, Numer. Math. 88 (3) (2001) 559–579.
[63] L. Grasedyck, L. Wang, J. Xu, A nearly optimal multigrid method for general unstructured grids, Numer. Math. 134 (3) (2016) 637–666.
[64] L. Berlyand, H. Owhadi, Flux norm approach to finite dimensional homogenization approximations with non-separated scales and high contrast, Arch. Ration. Mech. Anal. 198 (2) (2010) 677–721.
[65] H. Owhadi, L. Zhang, Localized bases for finite-dimensional homogenization approximations with nonseparated scales and high contrast, Multiscale Model. Simul. 9 (4) (2011) 1373–1398.
[66] Y. Efendiev, J. Galvis, T.Y. Hou, Generalized multiscale finite element methods (GMsFEM), J. Comput. Phys. 251 (2013) 116–135.
[67] Y. Efendiev, J. Galvis, R. Lazarov, M. Moon, M. Sarkis, Generalized multiscale finite element method. Symmetric interior penalty coupling, J. Comput. Phys. 255 (2013) 1–15.
[68] R.E. Alcouffe, A. Brandt, J.E. Dendy Jr., J.W. Painter, The multi-grid method for the diffusion equation with strongly discontinuous coefficients, SIAM J. Sci. Stat. Comput. 2 (4) (1981) 430–454.
[69] J. Fish, V. Belsky, Multigrid method for periodic heterogeneous media part 1: convergence studies for one-dimensional case, Comput. Methods Appl. Mech. Eng. 126 (1–2) (1995) 1–16.
[70] J. Fish, V. Belsky, Multi-grid method for periodic heterogeneous media part 2: multiscale modeling and quality control in multidimensional case, Comput. Methods Appl. Mech. Eng. 126 (1–2) (1995) 17–38.
[71] J. Fish, V. Belsky, Generalized aggregation multilevel solver, Int. J. Numer. Methods Eng. 40 (23) (1997) 4341–4361.
[72] A. Reusken, Multigrid with matrix-dependent transfer operators for a singular perturbation problem, Computing 50 (3) (1993) 199–211.
[73] C.E. Shuck, M. Frazee, A. Gillman, M.T. Beason, I.E. Gunduz, K. Matouš, R. Winarski, A.S. Mukasyan, X-ray nanotomography and focused-ion-beam sectioning for quantitative three-dimensional analysis of nanocomposites, J. Synchrotron Radiat. 23 (4) (2016) 990–996.
[74] T.F. Chan, W. Wan, Robust multigrid methods for nonsmooth coefficient elliptic linear systems, J. Comput. Appl. Math. 123 (1) (2000) 323–352.
[75] O. Tatebe, The multigrid preconditioned conjugate gradient method, in: The 6th Copper Mountain Conference on Multigrid Methods, NASA, Copper Mountain, CO, April 4–9, 1993, pp. 621–634.
[76] R.D. Falgout, U.M. Yang, hypre: a library of high performance preconditioners, in: International Conference on Computational Science, Springer, 2002, pp. 632–641.
[77] V.E. Henson, U.M. Yang, BoomerAMG: a parallel algebraic multigrid solver and preconditioner, Appl. Numer. Math. 41 (1) (2002) 155–177.
[78] D.A. Knoll, W.J. Rider, A multigrid preconditioned Newton–Krylov method, SIAM J. Sci. Comput. 21 (2) (1999) 691–710.
[79] M. Benzi, Preconditioning techniques for large linear systems: a survey, J. Comput. Phys. 182 (2) (2002) 418–477.
[80] Y. Saad, ILUT: a dual threshold incomplete LU factorization, Numer. Linear Algebra Appl. 1 (4) (1994) 387–402.
[81] C.-J. Lin, J.J. Moré, Incomplete Cholesky factorizations with limited memory, SIAM J. Sci. Comput. 21 (1) (1999) 24–45.
[82] D.A. May, J. Brown, L. Le Pourhiet, A scalable, matrix-free multigrid preconditioner for finite element discretizations of heterogeneous Stokes flow, Comput. Methods Appl. Mech. Eng. 290 (2015) 496–523.
[83] Y.A. Erlangga, C.W. Oosterlee, C. Vuik, A novel multigrid based preconditioner for heterogeneous Helmholtz problems, SIAM J. Sci. Comput. 27 (4) (2006) 1471–1492.
[84] R.W. dos Santos, G. Plank, S. Bauer, E.J. Vigmond, Parallel multigrid preconditioner for the cardiac bidomain model, IEEE Trans. Biomed. Eng. 51 (11) (2004) 1960–1968.
[85] G. Becker, C.M. Siefert, R.S. Tuminaro, H. Sun, D.M. Valiveti, A. Mohan, J. Yin, H. Huang, High resolution viscous fingering simulation in miscible displacement using a p-adaptive discontinuous Galerkin method with algebraic multigrid preconditioner, J. Comput. Phys. 374 (2018) 495–514.
[86] P.T. Lin, J.N. Shadid, J.J. Hu, R.P. Pawlowski, E.C. Cyr, Performance of fully-coupled algebraic multigrid preconditioners for large-scale VMS resistive MHD, J. Comput. Appl. Math. 344 (2018) 782–793.
[87] T.A. Wiesner, J.N. Shadid, E.C. Cyr, J.J. Hu, R.S. Tuminaro, MueLu-a multigrid framework for multiphysics preconditioners, Tech. Rep., Sandia National Lab. (SNL-NM)/Sandia National Laboratories, Albuquerque, NM (United States)/Livermore, CA, 2017.
[88] G. Haase, M. Liebmann, C.C. Douglas, G. Plank, A parallel algebraic multigrid solver on graphics processing units, in: High Performance Computing and Applications, Springer, 2010, pp. 38–47.
[89] D.J. Lucia, P.S. Beran, W.A. Silva, Reduced-order modeling: new approaches for computational physics, Prog. Aerosp. Sci. 40 (1–2) (2004) 51–117.
[90] J. Shlens, A tutorial on principal component analysis, arXiv preprint, arXiv:1404.1100.
[91] R. Vidal, Y. Ma, S.S. Sastry, Generalized Principal Component Analysis, vol. 5, Springer, 2016.
[92] R.R. Coifman, S. Lafon, Diffusion maps, Appl. Comput. Harmon. Anal. 21 (1) (2006) 5–30.
[93] R. Talmon, I. Cohen, S. Gannot, R.R. Coifman, Diffusion maps for signal processing: a deeper look at manifold-learning techniques based on kernels and graphs, IEEE Signal Process. Mag. 30 (4) (2013) 75–86.
[94] S. Bhattacharjee, K. Matouš, A nonlinear manifold-based reduced order model for multiscale analysis of heterogeneous hyperelastic materials, J. Comput. Phys. 313 (2016) 635–653.

[95] H. Xiao, J.-L. Wu, J.-X. Wang, R. Sun, C.J. Roy, Quantifying and reducing model-form uncertainties in Reynolds-averaged Navier–Stokes simulations: a data-driven, physics-informed Bayesian approach, J. Comput. Phys. 324 (2016) 115–136.
[96] P.S. Vassilevski, Coarse spaces by algebraic multigrid: multigrid convergence and upscaling error estimates, Adv. Adapt. Data Anal. 3 (2011) 229–249.
[97] S.P. MacLachlan, J.D. Moulton, Multilevel upscaling through variational coarsening, Water Resour. Res. 42 (2) (2006).
[98] J.D. Moulton, J.E. Dendy Jr., J.M. Hyman, The black box multigrid numerical homogenization algorithm, J. Comput. Phys. 142 (1998) 80–108.
[99] T. Porter, T. Duff, Compositing digital images, in: ACM Siggraph Computer Graphics, vol. 18, ACM, 1984, pp. 253–259.
[100] A. Mammen, Transparency and antialiasing algorithms implemented with the virtual pixel maps technique, IEEE Comput. Graph. Appl. 9 (4) (1989) 43–55.
[101] N. Otsu, A threshold selection method from gray-level histograms, IEEE Trans. Syst. Man Cybern. 9 (1) (1979) 62–66.
[102] R. Scheichl, P.S. Vassilevski, L.T. Zikatanov, Multilevel methods for elliptic problems with highly varying coefficients on nonaligned coarse grids, SIAM J. Numer. Anal. 50 (3) (2012) 1675–1694.
[103] B. Aksoylu, Z. Yeter, Robust multigrid preconditioners for cell-centered finite volume discretization of the high-contrast diffusion equation, Comput. Vis. Sci. 13 (5) (2010) 229–245.
[104] W.L. Briggs, V.E. Henson, S.F. McCormick, A Multigrid Tutorial, Society for Industrial and Applied Mathematics, 2000.
[105] D. Zill, W.S. Wright, M.R. Cullen, Advanced Engineering Mathematics, Jones and Bartlett Publishers, 2011.
[106] L. Rade, B. Westergren, Mathematics Handbook for Science and Engineering, Springer Science and Business Media, 2013.
[107] G. Guennebaud, B. Jacob, Eigen v3, http://eigen.tuxfamily.org, 2010.
[108] N.J. Higham, Estimating the matrix p-norm, Numer. Math. 62 (1) (1992) 539–555.
[109] J. Josse, Measuring multivariate association and beyond, Stat. Surv. 10 (2016) 132–167.
[110] G.J. Székely, M.L. Rizzo, N.K. Bakirov, Measuring and testing dependence by correlation of distances, Ann. Stat. 35 (6) (2007) 2769–2794.
[111] M.T. Heath, Scientific Computing: An Introductory Survey, vol. 80, Society for Industrial and Applied Mathematics, 2018.