AGU ADVANCING EARTH AND SPACE SCIENCE

Water Resources Research

METHOD

10.1029/2022WR032985

Key Points:

- An efficient and robust iterative solver for discretization of the elliptic mixed-dimensional Laplacian is presented
- The performance is validated on community benchmark problems for flow in fractured porous media
- Robustness is assessed across eight orders of magnitude variation in physical parameters

Correspondence to:

E. Keilegavlen,
Eirik.Keilegavlen@uib.no

Citation:

Hu, X., Keilegavlen, E., & Nordbotten, J. M. (2023). Effective preconditioners for mixed-dimensional scalar elliptic problems. *Water Resources Research*, *59*, e2022WR032985. https://doi.org/10.1029/2022WR032985

Received 7 JUN 2022
Accepted 24 DEC 2022

Effective Preconditioners for Mixed-Dimensional Scalar Elliptic Problems

Xiaozhe Hu[1,2], Eirik Keilegavlen[2], and Jan M. Nordbotten[2]

[1]Department of Mathematics, Tufts University, Medford, MA, USA, [2]Center for Modeling of Coupled Subsurface Dynamics, Department of Mathematics, University of Bergen, Bergen, Norway

Abstract Discretization of flow in fractured porous media commonly lead to large systems of linear equations that require dedicated solvers. In this work, we develop an efficient linear solver and its practical implementation for mixed-dimensional scalar elliptic problems. We design an effective preconditioner based on approximate block factorization and algebraic multigrid techniques. Numerical results on benchmarks with complex fracture structures demonstrate the effectiveness of the proposed linear solver and its robustness with respect to different physical and discretization parameters.

1. Introduction

Mixed-dimensional scalar elliptic equations are the backbone of models across many applications wherein potentially intersecting thin layers are embedded into a material. This application is of particular relevance in subsurface flows, where the thin layers correspond to fracture networks embedded in a porous material.

Flow in fractured porous media can be modeled in several ways. However, three of the main modeling approaches result in essentially mixed-dimensional elliptic equations. In sequence, these are (a) Direct mixed-dimensional modeling (Boon et al., 2018, 2021; Frih et al., 2012), (b) Discrete fracture networks (Erhel et al., 2009; Hyman et al., 2015), and (c) Low-order numerical methods for equidimensional fracture models (Karimi-Fard et al., 2004; Sandve et al., 2012). These modeling approaches are presented in detail in the recent review paper (Berre et al., 2019), and two community benchmark studies have recently been conducted (Berre et al., 2021; Flemisch et al., 2018). The relative merits of the three modeling approaches have been extensively analyzed in the literature. However, for our purpose, it suffices to point out that after discretization by standard numerical methods, the resulting algebraic linear systems have a similar structure.

Despite the importance of modeling flow in fractured porous media, the computational cost of linear solvers has not received much attention, although linear solvers for discrete fracture networks recently were studied in Greer et al. (2022). Indeed, in recent benchmark studies, the computational cost is simply represented by proxy in terms of the condition number of the system matrix. The construction, or even existence, of efficient iterative solvers can not be taken for granted. The material contrasts between porous rock and fracture may be significant, which is known to cause a challenge for many iterative algorithms, see (Chan & Wan, 2000; Graham & Hagger, 1999; Mandel & Brezina, 1996; Nepomnyaschikh, 1999; Oswald, 1999; Trottenberg et al., 2000) and references therein.

This technical note presents an efficient iterative linear solver for the prototypical mixed-dimensional scalar elliptic problem, which serves as a building block for solving other mixed-dimensional problems, for example, (Budiša & Hu, 2021) and (Budiša et al., 2020). It can also be one building block in solvers for multiphysics problems such as poromechanics and fracture deformation, for example, (Ferronato et al., 2019; Franceschini et al., 2019; White et al., 2016). We explore the block structure of the resulting linear systems and design effective block preconditioners based on approximate block factorization and algebraic multigrid (AMG) methods. The successful application requires several essential steps. First, because the blocks corresponding to fractures are usually smaller in size and lower in dimension, we build an approximation Schur complement on the domain via a simple diagonal approximation to the fracture block. Second, for the diagonal blocks in the block preconditioners, we employ existing AMG methods to approximately invert the diagonal blocks via several steps of V-cycles. Finally, the block preconditioners are used to accelerate the Krylov iterative methods, which results in an effective and robust linear solver.

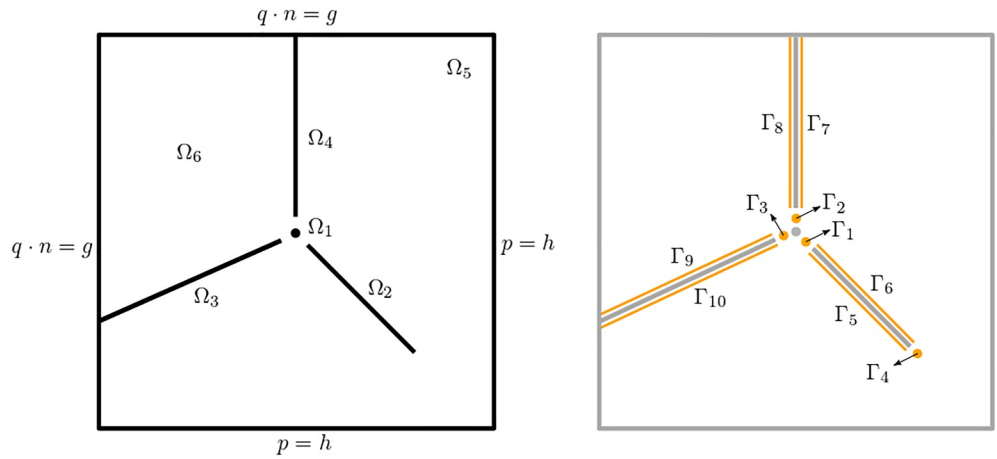

Figure 1. Left: A mixed-dimensional geometry consisting of three fractures, one intersection point, and with the top-dimensional domain split into two. Possible boundary conditions are indicated. Right: Geometry of the interfaces; Γ_4 is included as a placeholder for a no-flux condition on the tip of Ω_2.

The novelty of this technical note is identifying an efficient and robust iterative solver for mixed-dimensional scalar elliptic equations discretized using the unified framework developed in Nordbotten et al. (2019). We support this claim by applying the solver to one 2D benchmark problem with complex fracture networks (Flemisch et al., 2018), and to two 3D benchmark problems (Berre et al., 2021) with structured and complex fracture networks, respectively, and show robustness (in terms of iteration count until convergence) across a range of material parameters and grid resolutions.

2. Preliminaries

2.1. Mixed-Dimensional Scalar Elliptic Problems

Our model for flow in fractured porous media is the same as used in the benchmark for 3d flow (Berre et al., 2021) and several other works, for example, (Boon et al., 2018; Gläser et al., 2022). We model the fractures as $(N-1)$-dimensional objects embedded in the N-dimensional host medium $\Omega \subset \mathbb{R}^N$, with fracture intersections forming lines and points of dimensions $(N-2)$ and $(N-3)$. We refer to these geometric objects as subdomains and let Ω_i denote an arbitrary subdomain, so that the computational domain $\Omega = \bigcup_{i=1}^{m} \Omega_i$. The subdomains are connected via interfaces, denoted Γ_j, $j = \{1, \ldots, M\}$. If a D-dimensional interface Γ_j connects subdomains $\Omega_{\hat{\jmath}}$ and $\Omega_{\check{\jmath}}$, of dimension $D+1$ and D, respectively, we have that geometrically $\Gamma_j = \Omega_{\hat{\jmath}}$, and furthermore denote the part of the boundary of $\Omega_{\hat{\jmath}}$ coinciding with Γ_j as $\Gamma_j = \partial_j \Omega_{\hat{\jmath}}$. Finally, for Ω_i, let $\hat{S}_i$ be the set of interfaces toward neighboring subdomains of higher dimension, such that $j \in \hat{S}_i$ if and only if $\check{\jmath} = i$. The geometry is illustrated in Figure 1, where we note that a combination of fracture geometry and boundary condition can lead to subdomains (of dimension N or lower) being disconnected from Dirichlet boundary conditions.

With this notion of geometry, the model for single-phase fluid flow in Ω_i is given in strong form as,

$$\nabla_\parallel \cdot (-K_{i,\parallel} \nabla_\parallel p_i) - \sum_{j \in \hat{S}_i} \lambda_j = f_i. \tag{1}$$

Here, K_i, p_i, and f_i represent permeability, pressure, and source terms in subdomain Ω_i, with subscript $\parallel$ indicating that the permeability is in the direction tangential to Ω_i. Likewise, the subscripts $\parallel$ on ∇ indicates that the gradient operators act in the direction tangential to Ω_i. The flux over Γ_j, denoted λ_j, is governed by a law on the form

$$\lambda_j = -\kappa_j \left(p_{\check{\jmath}} - p_{\hat{\jmath}} \right), \tag{2}$$

The constant κ_j can be considered related to the normal transmissivity across Γ_j. Thus the terms λ_j represent flow through Γ_j from $\Omega_{\hat{\jmath}}$ to Ω_i. This same flow through Γ_j then appears as a Neumann boundary term on $\partial_j \Omega_{\hat{\jmath}}$,

$$\left(-K_{\hat{j},\parallel}\nabla_{\parallel}p_{\hat{j}}\right)\cdot\mathbf{n}_{\hat{j}}=\lambda_j \tag{3}$$

where $\mathbf{n}_{\hat{j}}$ is the outer normal vector on the boundary of $\Omega_{\hat{j}}$.

Equation 1 remain valid also for the highest-dimensional domains (when $\hat{S}_i=\varnothing$, and as such the summation is void), and also for the lowest-dimensional point intersections (when there is no parallel directions, and as such the differential operators are void).

2.2. Discretization

We discretize the flow model following the unified framework developed in Nordbotten et al. (2019), wherein the subdomains are discretized independently with the interface fluxes treated as interior boundary terms. The framework is compatible with various discretization methods for elliptic equations; herein, we apply the finite volume multipoint flux approximation method (Aavatsmark, 2002) to discretize subdomain problems, but mixed and virtual finite elements have also been used (Nordbotten et al., 2019).

3. Preconditioners for Mixed-Dimensional Problems

In this section, we introduce the multigrid-based solver for solving the discretized mixed-dimensional problem Equations 1 and 2. Our solver takes advantage of the block structure of the linear systems after discretization. More precisely, we use the block structure to perform a block factorization and develop preconditioners based on the block factorization via appropriate approximated Schur complement. Finally, the multigrid method is used to obtain an efficient and practical preconditioner.

3.1. Block Structure of the Linear Systems of Equations

Since Equation 1 is defined in the domain and Equation 2 is defined on the interface, our the discretized linear systems naturally inherits the following two-by-two block structure,

$$\mathcal{A}p=f\Leftrightarrow\begin{pmatrix}A_{\Omega\Omega} & A_{\Omega\Gamma}\\ A_{\Gamma\Omega} & A_{\Gamma\Gamma}\end{pmatrix}\begin{pmatrix}p_\Omega\\ p_\Gamma\end{pmatrix}=\begin{pmatrix}f_\Omega\\ f_\Gamma\end{pmatrix}, \tag{4}$$

where the subscripts Ω and Γ denote the blocks related to the domain and the interface, respectively. Correspondingly p_Ω and p_Γ denote the unknowns in all the subdomain Ω_i and all the interfaces Γ_i, respectively. The diagonal blocks $A_{\Omega\Omega}$ and $A_{\Gamma\Gamma}$ denote the discrete problems in the domain and interfaces, respectively. The off-diagonal blocks $A_{\Omega\Gamma}$ and $A_{\Gamma\Omega}$ denote the interaction and coupling between the domain and the interface. Moreover, from the unified framework, we are guaranteed that $A_{\Omega\Gamma}=A_{\Gamma\Omega}^T$. However, for the sake of generality, we will not exploit this property in the following discussion.

A block factorization-based approach is a natural choice for solving the linear systems with block structure, such as Equation 4. There are two types of block factorization for a two-by-two block system which give different Schur complements. Here, since $A_{\Gamma\Gamma}$ is diagonal and easy to invert, we use the following form:

$$\mathcal{A}=\begin{pmatrix}A_{\Omega\Omega} & A_{\Omega\Gamma}\\ A_{\Gamma\Omega} & A_{\Gamma\Gamma}\end{pmatrix}=\begin{pmatrix}I & A_{\Omega\Gamma}A_{\Gamma\Gamma}^{-1}\\ 0 & I\end{pmatrix}\begin{pmatrix}S_{\Omega\Omega} & 0\\ 0 & A_{\Gamma\Gamma}\end{pmatrix}\begin{pmatrix}I & 0\\ A_{\Gamma\Gamma}^{-1}A_{\Gamma\Omega} & I\end{pmatrix}=:\mathcal{U}\mathcal{D}\mathcal{L}, \tag{5}$$

where $S_{\Omega\Omega}$ is the well-known Schur complement defined as

$$S_{\Omega\Omega}:=A_{\Omega\Omega}-A_{\Omega\Gamma}A_{\Gamma\Gamma}^{-1}A_{\Gamma\Omega}. \tag{6}$$

Such a block factorization Equation 5 serves the building block of our proposed block preconditioner. The block structure Equation 4 is common for all discretizations that follow the unified approach introduced in Nordbotten et al. (2019). Thus the block preconditioner based on Equation 5 should be applicable to more general discretization methods rather than the cell centered finite volumes considered herein. However, the preconditioner

applied to the subdomain problems must be changed. For our discretization, subdomain problems $S_{\Omega\Omega}$ and $A_{\Gamma\Gamma}$ are positive (semi-)definite, and any efficient scalar elliptic solver can be applied, such as our choice, the AMG method. If the subdomain discretization is of saddle-point type, then an efficient saddle-point solver should be applied.

3.2. Factorization-Based Block Preconditioner

Based on the block factorization Equation 5, we can immediately propose several block preconditioners, for example, $\mathcal{B}_D = \mathcal{D}^{-1}$ can be used as a block diagonal preconditioner, $\mathcal{B}_U := (\mathcal{U}\mathcal{D})^{-1}$ and $\mathcal{B}_L := (\mathcal{D}\mathcal{L})^{-1}$ can be used as block upper and lower triangular preconditioners, respectively. For the sake of simplicity, in this subsection, we focus our discussion on $\mathcal{B}_L$ (and its variants).

If we use $\mathcal{B}_L$ as a right preconditioner for Krylov iterative methods, such as general minimal residual (GMRes) method, we need to look at the spectrum of the preconditioned linear system $\mathcal{A}\mathcal{B}_L$. From Equation 5, we have $\mathcal{A}\mathcal{B}_L = \mathcal{U}\mathcal{D}\mathcal{L}(\mathcal{D}\mathcal{L})^{-1} = \mathcal{U}$. This immediately implies that the eigenvalues of $\mathcal{A}\mathcal{B}_L$ are $\lambda(\mathcal{A}\mathcal{B}_L) = \lambda(\mathcal{U}) = 1$. Therefore, $\mathcal{B}_L$ is an efficient and robust preconditioner for solving Equation 4.

Note that the action of

$$\mathcal{B}_L = \begin{pmatrix} S_{\Omega\Omega} & 0 \\ A_{\Gamma\Omega} & A_{\Gamma\Gamma} \end{pmatrix}^{-1}$$

requires computing the Schur complement $S_{\Omega\Omega}$, the inverse of the Schur complement $S_{\Omega\Omega}^{-1}$, and $A_{\Gamma\Gamma}^{-1}$. All these steps could be quite expensive and we instead approximate them one by one.

For approximating the Schur complement, we use the fact that $A_{\Gamma\Gamma}$ is dominated by the diagonal term stemming from the discretization of the left-hand-side of Equation 2, with off-diagonal terms from the construction of $p_{\hat{j}}$ in Equation 2 will be comparatively small (and may be zero, depending on the discretization method applied in $\Omega_{\hat{j}}$). See (Nordbotten et al., 2019) for details. Numerically, for the benchmark problems tested in Section 4, the diagonal entries are usually several magnitudes larger than the off-diagonal entries. We therefore replace $A_{\Gamma\Gamma}$ in Equation 6 by its diagonal $\mathrm{diag}(A_{\Gamma\Gamma})$ and approximate the Schur complement as

$$\tilde{S}_{\Omega\Omega} := A_{\Omega\Omega} - A_{\Omega\Gamma}(\mathrm{diag}(A_{\Gamma\Gamma}))^{-1}A_{\Gamma\Omega}. \tag{7}$$

We comment that since the second term on the left-hand-side of Equation 7 is from lower dimensions, comparing with $A_{\Omega\Omega}$, the sparsity pattern of $\tilde{S}_{\Omega\Omega}$ only becomes denser near the interfaces (fractures) and the rest remains the same. This approximation leads to the following block lower triangular preconditioner

$$\tilde{\mathcal{B}}_L = \begin{pmatrix} \tilde{S}_{\Omega\Omega} & 0 \\ A_{\Gamma\Omega} & A_{\Gamma\Gamma} \end{pmatrix}^{-1}.$$

Unfortunately, exactly inverting $\tilde{\mathcal{B}}_L$ is expensive and, therefore, we approximately invert the two blocks on the main diagonal, and obtain the following block lower triangular preconditioner which we use in our numerical experiments,

$$\mathcal{M}_L = \begin{pmatrix} Q_{\Omega\Omega}^{-1} & 0 \\ A_{\Gamma\Omega} & Q_{\Gamma\Gamma}^{-1} \end{pmatrix}^{-1}, \tag{8}$$

where $Q_{\Omega\Omega}$ and $Q_{\Gamma\Gamma}$ approximate $\tilde{S}_{\Omega\Omega}^{-1}$ (essentially, $S_{\Gamma\Gamma}^{-1}$) and $A_{\Gamma\Gamma}^{-1}$, respectively.

3.3. Practical Implementation

In practice, given a right hand side $r = (r_\Omega, r_\Gamma)^T$, an algorithm for the action of the precondtioner $\mathcal{M}_L$ is shown in Algorithm 1.

Algorithm 1. Action of the Preconditioner $\mathcal{M}_L$: $z \leftarrow \mathcal{M}_L r$

1: Solve in the domain: $z_\Omega \leftarrow Q_{\Omega\Omega} r_\Omega$
2: Update the interface residual: $r_\Gamma \leftarrow r_\Gamma - A_{\Gamma\Omega} z_\Omega$
3: Solve on the interface: $z_\Gamma \leftarrow Q_{\Gamma\Gamma} r_\Gamma$
4: Output the update: $z \leftarrow (z_\Omega, z_\Gamma)^T$

The algorithm requires properly chosen $Q_{\Omega\Omega}$ and $Q_{\Gamma\Gamma}$. Since we are solving the scalar elliptic problem, we do this by applying one V-cycle of an AMG method to $\tilde{S}_{\Omega\Omega}$ and $A_{\Gamma\Gamma}$, respectively. Specifically, our implementation uses smoothed aggregation AMG (SA-AMG) methods to balance the computational complexity and convergence behavior (Brezina & Mandel, 2001; Brezina et al., 2012; Hu & Vassilevski, 2019; Vaněk et al., 1996). In general, other variants of AMG methods can be applied and similarly the number of V-cycle steps can be modified. In addition, one could also take advantage of the fact that the second term of Equation 7 is a low-rank update from lower dimensions and design special tailored solvers to define $Q_{\Omega\Omega}$ and further improve the overall efficiency. However, in our numerical experiments, it seems that one V-cycle is sufficient to provide a good approximation and, as a result, leads to an effective preconditioner.

We comment that $\tilde{S}_{\Omega\Omega}$ and $A_{\Gamma\Gamma}$ have block structures themselves since we put different subdomains Ω_i together. Therefore, it is possible to design special geometric and algebraic MG methods for approximating $\tilde{S}_{\Omega\Omega}$ and $A_{\Gamma\Gamma}$. For example, in the setup phase of SA-AMG, one could carefully design coarsening strategies so that aggregations will be constructed within the subdomains and interfaces of the same dimension and then form aggregations that possibly cross different dimensions. In addition, if the subdomains Ω_i and Γ_i are in the one-dimensional space, the cost of directly inverting the corresponding block is relatively negligible. In this work, our choice, SA-AMG implemented in the HAZmath package (Hu et al., 2022), already provides a good performance, and the specially tailored strategies suggested above do not appear to be necessary.

Finally, we note that if the spatial discretization is changed, the Schur complement $S_{\Omega\Omega}$ is changed as well and, therefore, $Q_{\Omega\Omega}$, which approximates $S_{\Omega\Omega}^{-1}$, must be adjusted accordingly. For instance, if mixed formulations are used in the subdomains, which leads to a saddle-point structure in the spatial discretization, then an efficient saddle-point solver should be applied as $Q_{\Omega\Omega}$.

4. Numerical Results

In this section, we present numerical results to demonstrate the effectiveness of the proposed block preconditioner for solving the linear systems of equations after discretizing the mixed-dimensional scalar elliptic problem Equations 1 and 2. We use the preconditioner $\mathcal{M}_L$ Equation 8 to accelerate the GMRes method. One V-cycle SA-AMG with one step of Gauss-Seidel iteration as both pre-and post-smoothing steps is used to define $Q_{\Omega\Omega}$. In all our numerical experiments, we use a zero initial guess, and the GMRes method terminates when the relative residual is smaller than 10^{-6}. Numerical experiments are conducted on a Linux laptop with an Intel Core i7-10510U processor and 40 GB of RAM. The software packages used are PorePy (Keilegavlen et al., 2021) (for the discretization of the mixed-dimensional scalar elliptic problems) and HAZmath (Hu et al., 2022) (for the preconditioners and iterative solvers for solving the linear systems of equations). The meshes are generated by Gmsh (Geuzaine & Remacle, 2009), using PorePy's interface to Gmsh to control the mesh size. The runscripts used to produce the results presented below are available at (Keilegavlen & Hu, 2022).

4.1. 2D Complex Fracture Example

For the first example, we choose a 2D example with a complex fracture configuration (Flemisch et al., 2018) to demonstrate the robustness of the block preconditioner on a realistic fracture network. Such a complex fracture structure often occurs in geological rock simulations, where the geometrical and physical properties of the fracture network can significantly influence the stability of the linear solvers. In particular, as we can see from Figure 2 (Left), the fractured porous medium domain where tips and very acute intersections may decrease the

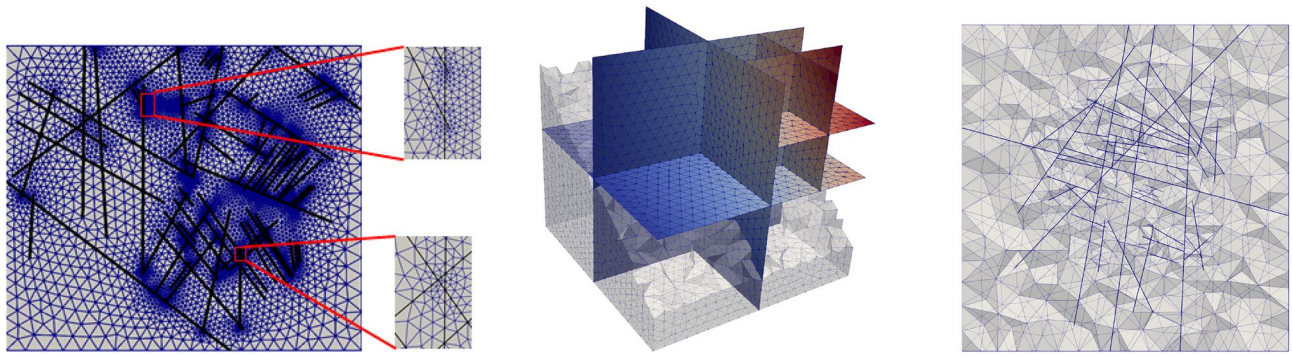

Figure 2. Left: Fracture geometry (black lines) and grid for the complex 2d case. Middle: Fracture geometry and grid for the regular 3d case; 2d objects are colored by the pressure solution. Right: Fracture network (shown as bold lines) and grid for the 3d field case, top view.

shape regularity of the mesh. Hence, this test case is well suited to show the robustness of the block preconditioner with respect to challenging geometric configurations that are common in realistic fracture geometries.

In the numerical experiments, we set the permeability of the matrix to be 1. The tangential and normal permeability of the fractures, denoted $K_{\|}$ and κ, respectively, are constants throughout the whole network. To show the robustness of the block preconditioner with respect to the discretization parameter h and physical parameters $K_{\|}$ and κ, we perform a set of tests in which we vary the values of those parameters.

Table 1 (Left) summarizes the numerical results. Each row in the table represents a set of tests where the tangential and normal fracture permeability values are fixed and the mesh size h is changing. It should be mentioned that these are mesh sizes prescribed to PorePy, but in practice, local geometric details combined with Gmsh's internal functionality to generate high-quality meshes may lead to further local grid refinement. When $h = 20$, the number of unknowns in all the subdomain, N_Ω, is 14,817 and the number of unknowns in all the interfaces, N_Γ, is 3,088. Therefore, the total number of degrees of freedom, N, is 17,905. When $h = 10$, we have $N_\Omega = 24,628$, $N_\Gamma = 3,710$, and $N = 28,338$. Finally, when $h = 5$, $N_\Omega = 60,892$, $N_\Gamma = 5,516$, and $N = 66,408$. As we can see, the numbers of GMRes iterations grow slightly but remain under 35 iterations, which demonstrates the robustness

Table 1
Performance of General Minimal Residual With Block Preconditioner $\mathcal{M}_L$ When Varying Mesh Size h (or Mesh Size of the Fracture h_f), Tangential Fracture Permeability $K_{\|}$, and Normal Fracture Permeability κ

		2D complex			3D regular			3D field		
$K_{\|}$	κ	$h = 20$	$h = 10$	$h = 5$	$h = 0.2$	$h = 0.1$	$h = 0.05$	$h_f = 80$	$h_f = 40$	$h_f = 20$
10^{-4}	10^{-4}	14	14	32	7	8	10	18	16	34
10^{-4}	1	13	12	22	16	20	26	35	49	44
10^{-4}	10^{4}	11	12	10	9	10	12	68	69	44
1	10^{-4}	13	13	33	8	11	13	13	15	32
1	1	12	13	22	8	8	10	43	57	50
1	10^{4}	12	11	10	8	10	11	71	73	47
10^{4}	10^{-4}	10	10	12	8	8	9	12	17	28
10^{4}	1	26	27	35	8	10	11	32	40	45
10^{4}	10^{4}	21	22	27	8	8	10	187	72	56
CPU time: AVG.		6.09e−2	1.06e−1	3.99e−1	4.13e−2	1.65e−1	9.06e−1	3.06e0	7.15e0	3.01e1
CPU time: STD.		1.60e−2	2.27e−2	1.33e−1	8.80e−3	4.66e−2	2.69e−1	2.65e0	3.06e0	4.04e0
CPU time: CV.		0.26	0.21	0.33	0.21	0.28	0.30	0.87	0.43	0.13

Note. Left: 2D complex fracture example. Middle: 3D Regular fracture example. Right: 3D field example. (CPU time includes forming the approximation Schur complement and solving the linear system. AVG., average; STD., standard derivation; CV., coefficient of variation. The unit for the CPU time AVG. and STD. is seconds).

of the preconditioner with respect to the mesh size h. On the other hand, each column of the table presents a set of tests where the mesh size is fixed, and the tangential and normal fracture permeability values vary. Again, the numbers of GMRes iteration remain stable as expected, which demonstrate the robustness of the block preconditioner with respect to the physical parameters. We also report the average (AVG.), the standard derivation (STD.), and the coefficient of variation (CV.) of the CPU time (including forming the approximate Schur complement Equation 7) in Table 1. The average CPU time indicates a nearly optimal computational complexity $\mathcal{O}\left(N^{1.45}\right)$, and the relatively small STD and CV also confirm the robustness of the preconditioner. Overall, Table 1 (Left) illustrates the effectiveness of the block preconditioner for this 2D complex fracture benchmark. For comparison, the same AMG method, when applied to solve $\mathcal{A}$ directly, fails to reach the tolerance within 200 iteration for all cases.

4.2. 3D Regular Fracture Example

Now we consider a 3D problem from a benchmark study (Berre et al., 2021). The 3D geometry is a unit cube as shown in Figure 2 (Middle). The fracture network consists of 9 fracture planes, 69 intersection lines, and 27 intersection points. Again, to show the robustness of the block preconditioner with respect to the discretization and physical parameters, we vary the values of the tangential and normal fracture permeability and the mesh size, keeping the permeability of the porous medium unitary. For $h = 0.2$, $N_\Omega = 3,285$, $N_\Gamma = 1,707$, and $N = 4,992$. For $h = 0.1$, $N_\Omega = 10,244$, $N_\Gamma = 4,077$, and $N = 14,321$. Finally, for $h = 0.05$, $N_\Omega = 45,612$, $N_\Gamma = 9,909$, and $N = 55,521$.

Table 1 (Middle) shows the number of GMRes iterations and CPU time for this 3D regular fracture example. We can see that the numerical results are consistent with our 2D example, that is, the block preconditioners show robustness with respect to the mesh size h, the tangential fracture permeability $K_\parallel$, and normal fracture permeability κ. Furthermore, the average CPU time shows a nearly optimal $\mathcal{O}\left(N^{1.28}\right)$ computational complexity. For comparison, the same AMG method, when applied to solve $\mathcal{A}$ directly, fails to reach the tolerance within 200 iterations for 22 out of 27 cases. For the 5 convergent cases, AMG takes more than 100 iterations on average.

4.3. 3D Field Example

Our last example is a simulation of a 3D field benchmark with a realistic fracture network (Berre et al., 2021). The domain is $\Omega = (-500, 350) \times (100, 1{,}500) \times (-100, 500)$, and a cross-section of the domain is shown in Figure 2 (Right) where we can see the complex fracture network and computational grid. In this example, we again set the permeability in the matrix to be 1 and vary the values of the tangential and normal fracture permeability and the mesh size in our numerical experiments, which are the same as in previous examples. However, due to the complexity of this example and computational cost consideration, we fix the far-field mesh size of the matrix to be 100 and vary the degree of mesh refinement toward the fractures, mesh size in the fractures (denoted by h_f) only. For $h_f = 80$, $N_\Omega = 44,539$, $N_\Gamma = 16,791$, and $N = 61,330$. For $h_f = 40$, $N_\Omega = 114,426$, $N_\Gamma = 35,143$, and $N = 149,569$. For $h_f = 20$, $N_\Omega = 463,922$, $N_\Gamma = 99,513$, and $N = 563,435$. Larger values of h_f give more irregular elements, especially near the intersection of the fractures and many tightly packed fractures, thus, the performance of the linear solver is expected to improve as h_f is decreased.

The results for the 3D field example are reported in Table 1 (Right). As expected, when h_f is large, the mesh quality influences the overall performance, and the number of GMRes iterations varies quite a bit, see the column of $h_f = 80$ in Table 1 (Right). When the fracture mesh size h_f gets smaller, the number of GMRes iterations stabilizes, which can be further confirmed by the fact that the CV gets smaller. The performance is robust with respect to the discretization and physical parameters, as we expected. Due to the complex geometry, the overall number of iterations is higher than the 3D regular fracture example. However, the average CPU time indicates a nearly optimal $\mathcal{O}\left(N^{1.03}\right)$ computational complexity, which is better than the 3D regular fracture example. Overall, for complex fracture networks, we suggest investing in constructing a more regular mesh and then applying the proposed block preconditioners in the iterative solvers. Finally, for comparison, the same AMG method, when applied to solve $\mathcal{A}$ directly, fails to converge within 200 iteration for all cases.

5. Conclusions

Based on the block structure of the linear systems arising from discretizing the mixed-dimensional scalar elliptic problems, we are able to develop block preconditioners based on approximate factorization. We first properly

approximate the Schur complement to obtain a block preconditioner and then apply the AMG methods to invert the diagonal blocks in our practical implementation. Several benchmarks in 2D and 3D are considered. From the numerical results, the GMRes methods accelerated by our proposed preconditioner are robust with respect to the physical and discretization parameters and complex fracture structures, making it attractive for real-world applications.

Data Availability Statement

The data and source code for the results presented herein are available at https://doi.org/10.5281/zenodo.6593919.

Acknowledgments
This work was financed in part by Norwegian Research Council Grant 308733. The work of X. Hu is partially supported by the National Science Foundation under Grant DMS-2208267.

References

Aavatsmark, I. (2002). An introduction to multi-point flux approximations for quadrilateral grids. *Computational Geosciences*, *6*(3/4), 405–432. https://doi.org/10.1023/A:1021291114475

Berre, I., Boon, W. M., Flemisch, B., Fumagalli, A., Gläser, D., Keilegavlen, E., et al. (2021). Verification benchmarks for single-phase flow in three-dimensional fractured porous media. *Advances in Water Resources*, *147*, 103759. https://doi.org/10.1016/j.advwatres.2020.103759

Berre, I., Doster, F., & Keilegavlen, E. (2019). Flow in fractured porous media: A review of conceptual models and discretization approaches. *Transport in Porous Media*, *130*(1), 215–236. https://doi.org/10.1007/s11242-018-1171-6

Boon, W. M., Nordbotten, J. M., & Vatne, J. E. (2021). Functional analysis and exterior calculus on mixed-dimensional geometries. *Annali di Matematica Pura ed Applicata (1923-)*, *200*(2), 757–789. https://doi.org/10.1007/s10231-020-01013-1

Boon, W. M., Nordbotten, J. M., & Yotov, I. (2018). Robust discretization of flow in fractured porous media. *SIAM Journal on Numerical Analysis*, *56*(4), 2203–2233. https://doi.org/10.1137/17M1139102

Brezina, M., & Mandel, J. (2001). Convergence of algebraic multigrid based on smoothed aggregation. *Numerische Mathematik*, *88*(3), 559–579. https://doi.org/10.1007/s002110000226

Brezina, M., Vaněk, P., & Vassilevski, P. S. (2012). An improved convergence analysis of smoothed aggregation algebraic multigrid. *Numerical Linear Algebra with Applications*, *19*(3), 441–469. https://doi.org/10.1002/nla.775

Budiša, A., Boon, W. M., & Hu, X. (2020). Mixed-dimensional auxiliary space preconditioners. *SIAM Journal on Scientific Computing*, *42*(5), A3367–A3396. https://doi.org/10.1137/19M1292618

Budiša, A., & Hu, X. (2021). Block preconditioners for mixed-dimensional discretization of flow in fractured porous media. *Computational Geosciences*, *25*(2), 671–686. https://doi.org/10.1007/s10596-020-09984-z

Chan, T. F., & Wan, W.-L. (2000). Robust multigrid methods for nonsmooth coefficient elliptic linear systems. *Journal of Computational and Applied Mathematics*, *123*(1–2), 323–352. https://doi.org/10.1016/S0377-0427(00)00411-8

Erhel, J., de Dreuzy, J.-R., & Poirriez, B. (2009). Flow simulation in three-dimensional discrete fracture networks. *SIAM Journal on Scientific Computing*, *31*(4), 2688–2705. https://doi.org/10.1137/080729244

Ferronato, M., Franceschini, A., Janna, C., Castelletto, N., & Tchelepi, H. A. (2019). A general preconditioning framework for coupled multiphysics problems with application to contact- and poro-mechanics. *Journal of Computational Physics*, *398*, 108887. https://doi.org/10.1016/j.jcp.2019.108887

Flemisch, B., Berre, I., Boon, W., Fumagalli, A., Schwenck, N., Scotti, A., et al. (2018). Benchmarks for single-phase flow in fractured porous media. *Advances in Water Resources*, *111*, 239–258. https://doi.org/10.1016/j.advwatres.2017.10.036

Franceschini, A., Castelletto, N., & Ferronato, M. (2019). Block preconditioning for fault/fracture mechanics saddle-point problems. *Computer Methods in Applied Mechanics and Engineering*, *344*, 376–401. https://doi.org/10.1016/j.cma.2018.09.039

Frih, N., Martin, V., Roberts, J. E., & Saâda, A. (2012). Modeling fractures as interfaces with nonmatching grids. *Computational Geosciences*, *16*(4), 1043–1060. https://doi.org/10.1007/s10596-012-9302-6

Geuzaine, C., & Remacle, J.-F. (2009). Gmsh: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. *International Journal for Numerical Methods in Engineering*, *79*(11), 1309–1331. https://doi.org/10.1002/nme.2579

Gläser, D., Schneider, M., Flemisch, B., & Helmig, R. (2022). Comparison of cell-and vertex-centered finite-volume schemes for flow in fractured porous media. *Journal of Computational Physics*, *448*, 110715. https://doi.org/10.1016/j.jcp.2021.110715

Graham, I. G., & Hagger, M. (1999). Unstructured additive Schwarz–conjugate gradient method for elliptic problems with highly discontinuous coefficients. *SIAM Journal on Scientific Computing*, *20*(6), 2041–2066. https://doi.org/10.1137/S1064827596305593

Greer, S., Hyman, J., & O'Malley, D. (2022). A comparison of linear solvers for resolving flow in three-dimensional discrete fracture networks. *Water Resources Research*, *58*(4), WR031188. https://doi.org/10.1029/2021wr031188

Hu, X., Adler, J. H., & Zikatanov, L. T. (2022). HAZmath: A simple finite element, graph, and solver library.

Hu, X., & Vassilevski, P. S. (2019). Modifying AMG coarse spaces with weak approximation property to exhibit approximation in energy norm. *SIAM Journal on Matrix Analysis and Applications*, *40*(3), 1131–1152. https://doi.org/10.1137/18M1165190

Hyman, J. D., Karra, S., Makedonska, N., Gable, C. W., Painter, S. L., & Viswanathan, H. S. (2015). dfnWorks: A discrete fracture network framework for modeling subsurface flow and transport. *Computers and Geosciences*, *84*, 10–19. https://doi.org/10.1016/j.cageo.2015.08.001

Karimi-Fard, M., Durlofsky, L. J., & Aziz, K. (2004). An efficient discrete-fracture model applicable for general-purpose reservoir simulators. *SPE Journal*, *9*(02), 227–236. https://doi.org/10.2118/88812-PA

Keilegavlen, E., Berge, R., Fumagalli, A., Starnoni, M., Stefansson, I., Varela, J., & Berre, I. (2021). PorePy: An open-source software for simulation of multiphysics processes in fractured porous media. *Computational Geosciences*, *25*(1), 243–265. https://doi.org/10.1007/s10596-020-10002-5

Keilegavlen, E., & Hu, X. (2022). Runscripts for linear solvers for mixed-dimensional flow problems. https://doi.org/10.5281/zenodo.6593919

Mandel, J., & Brezina, M. (1996). Balancing domain decomposition for problems with large jumps in coefficients. *Mathematics of Computation*, *65*(216), 1387–1401. https://doi.org/10.1090/s0025-5718-96-00757-0

Nepomnyaschikh, S. (1999). Preconditioning operators for elliptic problems with bad parameters. In *Eleventh international conference on domain decomposition methods* (pp. 82–88).

Nordbotten, J. M., Boon, W. M., Fumagalli, A., & Keilegavlen, E. (2019). Unified approach to discretization of flow in fractured porous media. *Computational Geosciences*, *23*(2), 225–237. https://doi.org/10.1007/s10596-018-9778-9

Oswald, P. (1999). On the robustness of the BPX-preconditioner with respect to jumps in the coefficients. *Mathematics of Computation*, *68*(226), 633–650. https://doi.org/10.1090/s0025-5718-99-01041-8

Sandve, T. H., Berre, I., & Nordbotten, J. M. (2012). An efficient multi-point flux approximation method for discrete fracture–matrix simulations. *Journal of Computational Physics*, *231*(9), 3784–3800. https://doi.org/10.1016/j.jcp.2012.01.023

Trottenberg, U., Oosterlee, C. W., & Schuller, A. (2000). *Multigrid*. Elsevier.

Vaněk, P., Mandel, J., & Brezina, M. (1996). Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems. *Computing*, *56*(3), 179–196. https://doi.org/10.1007/BF02238511

White, J. A., Castelletto, N., & Tchelepi, H. A. (2016). Block-partitioned solvers for coupled poromechanics: A unified framework. *Computer Methods in Applied Mechanics and Engineering*, *303*, 55–74. https://doi.org/10.1016/j.cma.2016.01.008