Parallel solver for $\boldsymbol{H}(\mathrm{div})$ problems using hybridization and AMG[‡]

Chak S. Lee[*] and Panayot S. Vassilevski[†]

January 19, 2016

Abstract

In this paper, a scalable parallel solver is proposed for $\boldsymbol{H}(\mathrm{div})$ problems discretized by arbitrary order finite elements on general unstructured meshes. The solver is based on hybridization and algebraic multigrid (AMG). Unlike some previously studied $\boldsymbol{H}(\mathrm{div})$ solvers, the hybridization solver does not require discrete curl and gradient operators as additional input from the user. Instead, only some element information is needed in the construction of the solver. The hybridization results in a H^1-equivalent symmetric positive definite system, which is then rescaled and solved by AMG solvers designed for H^1 problems. Weak and strong scaling of the method are examined through several numerical tests. Our numerical results show that the proposed solver provides a promising alternative to ADS, a state-of-the-art solver [12], for $\boldsymbol{H}(\mathrm{div})$ problems. In fact, it outperforms ADS for higher order elements.

1 Introduction

This paper is concerned with the $\boldsymbol{H}(\mathrm{div})$ bilinear form acting on vector functions $\boldsymbol{u}$, $\boldsymbol{v}$:

$$a(\boldsymbol{u},\boldsymbol{v}) = \int_\Omega \alpha\, \nabla\cdot\boldsymbol{u}\, \nabla\cdot\boldsymbol{v} + \beta\, \boldsymbol{u}\cdot\boldsymbol{v}\, d\boldsymbol{x}. \tag{1}$$

Here $\alpha,\ \beta \in L^\infty(\Omega)$ are some positive heterogeneous coefficients, and Ω is a simply-connected polygonal domain in $\mathbb{R}^d$, $d = 2, 3$. Discrete problems associated with $a(\cdot,\cdot)$ arise in many applications, such as first order least squares formulation of second order elliptic problems [4], preconditioning of mixed finite element methods [3], Reissner-Mindlin plates [2] and the Brinkman equations [13]. Let A be the linear system obtained from discretization of $a(\cdot,\cdot)$ by some $\boldsymbol{H}(\mathrm{div})$-conforming finite element of arbitrary order on a general unstructured mesh. Our goal is to design a scalable parallel solver for A.

It is well known that finding efficient iterative solvers for A is not trivial because of the "near-null space" of A. The currently available scalable parallel solvers include the auxiliary space divergence solver (ADS) [12] in the *hypre* library and PCBDDC [14] in the PETSc library. The former relies on the regular HX-decomposition for $\boldsymbol{H}(\mathrm{div})$ functions proposed in [10]. The setup of ADS is quite involved and require additional input from the user, namely, some discrete gradient and discrete curl operators. On the other hand, PCBDDC is based on the Balancing Domain Decomposition by Constraint algorithm [8]. Its construction requires that the local discrete systems are assembled at

[*]Department of Mathematics, Texas A&M University, cslee@math.tamu.edu

[†]Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, P.O. Box 808, L-561, Livermore, CA 94551, U.S.A., panayot@llnl.gov

[‡]This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

subdomain level. To accommodate high contrast and jumps in the coefficients, the primal space in PCBDDC is adaptively enriched by solving some generalized eigenvalue problems.

In this paper, we propose an alternative way to solve systems with A. Our approach is based on traditional hybridization technique used in the mixed finite element method ([3]), thus reducing the problem to a smaller problem for the respective Lagrange multipliers that are involved in the hybridization. The reduced problem is symmetric positive definite, and as is well-known, is H^1-equivalent. Thus, in principle, one may apply any scalable AMG solver that is suitable for H^1 problems. Unlike ADS, the hybridization approach does not require additional information from the user. Instead, it requires that the original problem is given in unassembled element-based form.

One main issue that has to be addressed is the choice of the basis of the Lagrange multiplier space. In general, the reduced problem contains the constant function in its near null-space. However, if the basis for the Lagrange multipliers is not properly scaled (i.e., does not provide partition of unity), the coefficient vector of the constant functions is not a constant multiple of the vector of ones. The latter is a main assumption in the design of AMG for H^1-equivalent problems. We resolve this problem in an algebraic way by constructing a diagonal matrix which we use to rescale the reduced system such that the constant vector is the near-null space of the rescaled matrix, so that the respective AMG is correctly designed.

The proposed hybridization with diagonal rescaling is implemented in a parallel code and its scalability is tested in comparison with the state-of-the-art ADS solver. The results demonstrate that the new solver provides a competitive alternative to ADS; it outperforms ADS very clearly for higher order elements.

The rest of the paper is organized as follows. In section 2, we give a detailed description of the hybridization technique. The properties of the hybridized system are discussed in section 3. After that, we present in section 4 several challenging numerical examples to illustrate the performance of the method comparing it with ADS. Lastly, we give some concluding remarks in section 5.

2 Hybridization

We consider the variational problem associated with the bilinear form (1): find $\boldsymbol{u} \in \boldsymbol{H}_0(\mathrm{div};\Omega)$ such that

$$a(\boldsymbol{u},\boldsymbol{v}) = (\boldsymbol{f},\boldsymbol{v}), \qquad \forall\, \boldsymbol{v} \in \boldsymbol{H}_0(\mathrm{div};\Omega). \tag{2}$$

Here, $\boldsymbol{f}$ is a given function in $\left(L^2(\Omega)\right)^d$ and $(\cdot,\cdot)$ is the usual L^2 inner product in Ω. Our following discussion is based on discretization of the variational problem (2) by Raviart-Thomas elements of arbitrary order. We note that other $\boldsymbol{H}(\mathrm{div})$-conforming finite elements can also be considered. Let $\mathcal{T}_h$ be a general unstructured mesh on Ω. The space of Raviart-Thomas elements of order $k \geq 0$ on $\mathcal{T}_h$ will be denoted by RT_k. For instance, if $\mathcal{T}_h$ is a simplicial mesh, then RT_k is defined to be

$$RT_k = \Big\{\, \boldsymbol{v}_h \in \boldsymbol{H}_0(\mathrm{div};\Omega) \;\Big|\; \boldsymbol{v}_h|_\tau \in \left(P_k(\tau)\right)^d + \boldsymbol{x}P_k(\tau) \quad \forall \tau \in \mathcal{T}_h \,\Big\},$$

where $P_k(\tau)$ denotes the set of polynomials of degree at most k on τ. For definitions of RT_k on rectangular/cubic meshes, see for example [3]. Discretization of (2) by RT_k elements results in a linear system of equations

$$Au = f. \tag{3}$$

We are going to formulate an equivalent problem such that the modified problem can be solved more efficiently. We note that RT_k basis functions are either associated with degrees of freedom (dofs) in the interior of elements, on boundary faces, or interior faces of a conforming finite element mesh. Those associated with dofs in the interior of elements or on boundary faces are supported in only one element, while those associated with dofs on interior faces are supported in two elements. In hybridization, the RT_k basis functions that are associated with dofs on interior faces are split

into two pieces, each supported in one and only one element. In practice, the splitting can be done by making use of the element-to-dofs relation table to identify the shared dofs between any pair of neighboring elements. This relation table can be constructed during the discretization. The space of Raviart-Thomas element after the splitting will be denoted by $\widehat{RT}_k$. If we discretize $a(\cdot,\cdot)$ with the basis functions in $\widehat{RT}_k$, the resulting system will have a block diagonal matrix $\widehat{A}$. Next, we need to enforce the continuity of the split basis functions in some way such that the solution of the modified system coincides with the original problem. Suppose a RT_k basis function $\boldsymbol{\phi}$ is split into $\widehat{\boldsymbol{\phi}}_1$ and $\widehat{\boldsymbol{\phi}}_2$. The simplest way is to use Lagrange multiplier space to make the coefficient vectors of the test functions from both sides of an interior interface to be the same. If we set such constraints for all the split basis functions, we obtain a constraint matrix C.

Remark 2.1 *There are other ways to enforce continuity of $\widehat{RT}_k$. For example, when constructing the constraint matrix C, one can also use the normal traces λ of the original RT_k basis functions as Lagrange multipliers, see [6].*

The modified problem after introducing the Lagrange multipliers takes the saddle–point form

$$\begin{bmatrix} \widehat{A} & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} \widehat{u} \\ \lambda \end{bmatrix} = \begin{bmatrix} \widehat{f} \\ 0 \end{bmatrix}. \tag{4}$$

Here, $\widehat{u}$ is the coefficient vector of $\widehat{\boldsymbol{u}}_h$. The saddle point problem (4) can be reduced

$$S\lambda = g, \tag{5}$$

where $S = C\widehat{A}^{-1}C^T$ and $g = C\widehat{A}^{-1}\widehat{f}$. The Schur complement S and the new right hand side g can be explicitly formed very efficiently because $\widehat{A}$ is block diagonal. In fact, the inversion of $\widehat{A}$ is embarrassingly parallel. Here, each local block of $\widehat{A}$ is invertible, so $\widehat{A}^{-1}$ is well-defined. We will show in the next section that S is actually a s.p.d. system of the Lagrange multipliers, and that it can be solved efficiently by existing parallel linear solvers. After solving for λ, $\widehat{u}$ can be computed by back substitution $\widehat{u} = \widehat{A}^{-1}(\widehat{f} - C^T\lambda)$. Noticing that the back substitution involves only an action of $\widehat{A}^{-1}$ (already available in the computation of S) and some matrix-vector multiplications, which are inexpensive (local) and scalable computations.

3 Discussion

The hybridization approach described in the previous section can be summarized as follows

1. Split the RT_k basis to obtain $\widehat{A}$ and $\widehat{f}$.
2. Compute $\widehat{A}^{-1}$ and form $S = C\widehat{A}^{-1}C^T$ and $g = C\widehat{A}^{-1}\widehat{f}$.
3. Solve the system $S\lambda = g$.
4. Recover $\widehat{u}$ by back substitution.

As explained in section 2, step 2 and 4 are scalable (inexpensive local) computations. In contrast, step 3 involves the main computational cost. Thus, it is important that we can solve S efficiently. In this section, we describe some properties of S. First, we show that S is related to some hybridized mixed discretization of the second order differential operator $-\nabla\cdot(\beta^{-1}\nabla) + \alpha^{-1}I$ (acting on scalar functions). We note that the differential problem associated with (2) is

$$-\nabla(\alpha\nabla\cdot\boldsymbol{u}) + \beta\boldsymbol{u} = \boldsymbol{f} \tag{6}$$

with homogeneous Dirichlet boundary condition $\boldsymbol{u}\cdot\boldsymbol{n}=0$. The latter operator acts on vector-functions. We now make the following connection between these two operators. If we introduce an additional variable $p=\alpha\nabla\cdot\boldsymbol{u}$, then (6) becomes the following first order system (for $\boldsymbol{u}$ and p)

$$\begin{aligned}\beta\boldsymbol{u}-\nabla p&=\boldsymbol{f},\\ \nabla\cdot\boldsymbol{u}-\alpha^{-1}p&=0.\end{aligned}\tag{7}$$

It is noteworthy to note that the structure of (7) is the same as the mixed formulation of the differential operator $-\nabla\cdot(\beta^{-1}\nabla)+\alpha^{-1}I$. So we can apply a hybridized mixed discretization [6, 7] for $-\nabla\cdot(\beta^{-1}\nabla)+\alpha^{-1}I$ to discretize (7). To apply the the hybridized mixed discretization, we note that the weak form of (7) is to find $(\boldsymbol{u},p)\in\boldsymbol{H}_0(\mathrm{div};\Omega)\times L^2(\Omega)$ such that

$$\begin{aligned}(\beta\boldsymbol{u},\boldsymbol{v})+(p,\nabla\cdot\boldsymbol{v})&=(\boldsymbol{f},\boldsymbol{v}) &&\forall\,\boldsymbol{v}\in\boldsymbol{H}_0(\mathrm{div};\Omega)\\ (\nabla\cdot\boldsymbol{u},q)-(\alpha^{-1}p,q)&=0 &&\forall\,q\in L^2(\Omega).\end{aligned}\tag{8}$$

Let $W_h^k\subset L^2(\Omega)$ be a space of piecewise polynomials such that RT_k and W_h^k form a stable pair for the mixed discretization of (8). For instance, for simplicial meshes, we can take

$$W_h^k=\Big\{\ q\in\boldsymbol{L}^2(\Omega)\ \Big|\ q|_\tau\in P_k(\tau)\quad\forall\tau\in\mathcal{T}_h\ \Big\}.$$

If (8) is discretized by the pair $\widehat{RT}_k$-W_h^k and the continuity of $\widehat{RT}_k$ is enforced by the constraint matrix C as described in section 2, we get a 3 by 3 block system of equations of the form

$$\begin{bmatrix}\widehat{M} & \widehat{B}^T & C^T\\ \widehat{B} & -W & 0\\ C & 0 & 0\end{bmatrix}\begin{bmatrix}\widehat{u}\\ p\\ \lambda\end{bmatrix}=\begin{bmatrix}\widehat{f}\\ 0\\ 0\end{bmatrix}.\tag{9}$$

As $\widehat{M}$ and W are weighted L^2 mass matrices of the spaces $\widehat{RT}_k$ and W_h^k respectively, they are invertible. Hence, the 2 by 2 block matrix $\begin{bmatrix}\widehat{M} & \widehat{B}^T\\ \widehat{B} & -W\end{bmatrix}$ is invertible, and (9) can be reduced to

$$\begin{bmatrix}C & 0\end{bmatrix}\begin{bmatrix}\widehat{M} & \widehat{B}^T\\ \widehat{B} & -W\end{bmatrix}^{-1}\begin{bmatrix}C^T\\ 0\end{bmatrix}\lambda=\begin{bmatrix}C & 0\end{bmatrix}\begin{bmatrix}\widehat{M} & \widehat{B}^T\\ \widehat{B} & -W\end{bmatrix}^{-1}\begin{bmatrix}\widehat{f}\\ 0\end{bmatrix}.\tag{10}$$

Since the $(1,1)$ block of $\begin{bmatrix}\widehat{M} & \widehat{B}^T\\ \widehat{B} & -W\end{bmatrix}^{-1}$ can be written as $(\widehat{M}+\widehat{B}^TW^{-1}\widehat{B})^{-1}$ and $\widehat{A}=\widehat{M}+\widehat{B}^TW^{-1}\widehat{B}$, the reduced problem (10) is in fact identical to (5). Therefore, the Schur complement S in (5) can be characterized by the hybridized mixed discretization for the differential operator $-\nabla\cdot(\beta^{-1}\nabla)+\alpha^{-1}I$.

Remark 3.1 *Actually the hybridized mixed discretization for $-\nabla\cdot(\beta^{-1}\nabla)+\alpha^{-1}I$ in [6, 7] gives rise to the reduced system $\widetilde{S}$ for the Lagrange multiplier λ where*

$$\widetilde{S}=C\Big(\widehat{M}^{-1}-\widehat{M}^{-1}\widehat{B}^T\big(\widehat{B}\widehat{M}^{-1}\widehat{B}^T+W\big)^{-1}\widehat{B}\widehat{M}^{-1}\Big)C^T.$$

However, since W is invertible, an application of the Sherman-Morrison-Woodbury formula implies that $\widetilde{S}=S$.

In [7], the authors proved that S is spectrally equivalent to the norm $|||\cdot|||$ on the space of Lagrange multipliers defined as

$$|||\lambda|||^2=\sum_{\tau\in\mathcal{T}_h}\frac{1}{|\partial\tau|}\|\lambda-m_\tau(\lambda)\|_{\partial\tau}^2$$

where $m_\tau(\lambda) = \frac{1}{|\partial\tau|}\int_{\partial\tau}\lambda\, ds$. More precisely, there are constants C_1 and C_2, depending only on the approximation order k, the coefficients α, β of the operator, and the shape regularity of $\mathcal{T}_h$ such that

$$C_1|||\lambda|||^2 \le \lambda^T S\lambda \le C_2|||\lambda|||^2 \qquad \forall\ \lambda.$$

Consequently, S is symmetric positive definite. Moreover, this shows that the near-null space of S is spanned by the constant functions, which is the main assumption to successfully apply solvers of AMG type. When solving with S, we opt for the parallel algebraic multigrid solver BoomerAMG from the *hypre* library [9].

The fact that the constant functions are in the near-null space of S is not sufficient to guarantee the efficiency of BoomerAMG. One of the factors affecting the success of BoomerAMG is that the constant coefficient vector $\mathbf{1}$ should be in the near-null space of S (as a matrix). However, this is not always the case. Indeed, depending on the choice of basis for the Lagrange multipliers space, the coefficient vector of a constant function is not necessarily a constant vector. To resolve this issue, we chose to rescale S by a diagonal matrix D such that the constant vector is now in the near-null space of D^TSD. To achieve this, we solve the homogeneous problem $Sd = 0$ by applying a few smoothing steps to a random initial guess. In our numerical experiments to be presented in the next section, we use 5 conjugate gradient (CG) iterations preconditioned by the Jacobi smoother in the computation of d, which is fairly inexpensive. Once d is computed, we set $D_{ii} = d_i$ (the ith entry of d). Noticing that $D\mathbf{1} = d$, so $\mathbf{1}$ is in the near-null space of D^TSD. We can then apply CG preconditioned by BoomerAMG constructed from D^TSD to efficiently solve the system

$$(D^TSD)\lambda_D = D^Tg.$$

Lastly, the original Lagrange multiplier λ is recovered simply by setting $\lambda = D\lambda_D$.

Another useful feature of S is that its size is less than or equal to the size of the original matrix A. This is because there is a one-to-one correspondence between Lagrange multipliers and Raviart-Thomas basis functions associated with interior faces. For higher order Raviart-Thomas elements, a portion of the basis functions are associated with interior of elements. These basis functions are supported in one element only, so they do not need Lagrange multipliers to enforce their continuity. Hence, for higher order approximations, the size of S is considerably smaller than the size of A. As a result, methods for solving with S is likely to be more efficient and faster than solving with A (using the state-of-the-art solvers such as ADS) which is confirmed by our experiments.

4 Numerical Examples

In this section, we present some numerical results regarding the performance of our hybridization AMG solver. All the experiments are performed on the cluster Sierra at the Lawrence Livermore National Laboratory. Sierra has a total of 1944 nodes (Intel Xeon EP X5660 clocked at 2.80 GHz), which are connected by InfiniBand QDR. Each node has 12 cores and equipped with 24 GB of memory.

In the solution process, the hybridized system with S is rescaled by the diagonal matrix D as described in the previous section. The rescaled system D^TSD is then solved by the CG method preconditioned with BoomerAMG (constructed from D^TSD) from the *hypre* library. As one of our goals is to compare the hybridization AMG solver with ADS, we present also the performance of ADS in all the examples. In order to have fair comparisons, the time to solution for the hybridization AMG solver includes the formation time of the Schur complement S, the computation time to construct the rescaling matrix D, the solve time for the problem with the modified matrix D^TSD by CG preconditioned by BoomerAMG, and the recovery time of the original unknown $\boldsymbol{u}$. The time to solution for ADS is simply the solve time for the original problem with A by the CG preconditioned by ADS. For the tables in the present section, # proc refers to the number of processors, while # iter refers to the number of PCG iterations.

4.1 Weak Scaling

We first test the weak scaling of the hybridization AMG solver. The problem setting is as follows. We will solve problem (3) obtained by RT_k discretization on uniform tetrahedral mesh in 3D. Starting from some initial tetrahedral mesh, we refine the mesh uniformly. The problem size will be increased by about 8 times after one such refinement. At the same time, the number of processors for solving the refined problem is increased 8 times so that the problem size per processor is kept roughly the same. Both the lowest order Raviart-Thomas elements RT_0 and a higher order elements, RT_2, are

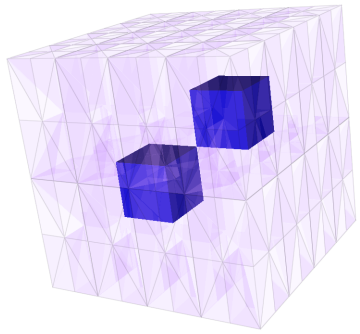


Figure 1: Initial mesh for the RT_2 weak scaling test case. Blue region indicates Ω_i.

considered. We solve a heterogeneous coefficient problem on the unit cube, i.e. $\Omega = [0,1]^3$. The boundary conditions are $\boldsymbol{u} \cdot \boldsymbol{n} = 0$ on $\partial\Omega$, and the source function $\boldsymbol{f}$ is the constant vector $[1,1,1]^T$. Let $\Omega_i = [\frac{1}{4}, \frac{1}{2}]^3 \cup [\frac{1}{2}, \frac{3}{4}]^3$. We will consider β being constant 1 throughout the domain, whereas

$$\alpha = \begin{cases} 1 & \text{in } \Omega \backslash \Omega_i \\ 10^p & \text{in } \Omega_i \end{cases}$$

and we choose p = -4, 0, or 4. For RT_2 test case, we first partition Ω into 8 x 8 x 4 parallelepipeds. The initial tetrahedral mesh in this case is then obtained by subdividing each parallelepiped into tetrahedrons, see figure 1. The initial mesh of the RT_0 test case is obtained by refining the initial mesh of the RT_2 test case 3 times. The PCG iterations are stopped when the l_2 norm of the residual is reduced by a factor of 10^{10}. The time to solution (in seconds) of both the hybridization AMG and ADS for the RT_0 case are shown in table 1. Additionally, we also report the number of PCG iterations in the brackets. We see that the number of iterations of the hybridization solver are very stable against problem size and the heterogeneity of α. The average time to solution of the hybridization approach is about 2 times faster than that of ADS. The solution time difference between the two solvers is more significant in the high order discretization case. This is due to the fact that size of the hybridized system S is much smaller than the size of the original system A. Indeed, in the case of RT_2, the average time to solution of the hybridization approach is about 8 times faster than that of ADS, see Table 2. In figure 2, we plot the solution time of both solvers where $p = 4$ in the definition of α. We can see that the hybridization solver has promising weak scaling up to 1536 processors.

4.2 Strong Scaling

In the second example, we investigate the strong scaling of the hybridization AMG solver. The problem considered in this section is the crooked pipe problem, see [12] for a detail description of the problem. The mesh for this problem is depicted in figure 3. The coefficient α and β are piecewise constants. More precisely, $(\alpha, \beta) = (1.641, 0.2)$ in the red region, and $(\alpha, \beta) = (0.00188, 2000)$ in

$\alpha = \{1, 10^p\}$, $\beta \equiv 1$		p		
# proc	Problem size	-4	0	4
Hybridization-BoomerAMG-CG				
3	200,704	0.97 (24)	0.96 (21)	0.93 (21)
24	1,589,248	1.15 (24)	1.15 (23)	1.16 (23)
192	12,648,448	1.45 (27)	1.48 (25)	1.43 (24)
1,536	100,925,440	3.31 (29)	3.03 (28)	3.03 (28)
ADS-CG				
3	200,704	2.68 (21)	1.74 (10)	1.79 (11)
24	1,589,248	4.04 (25)	3.53 (13)	3.54 (13)
192	12,648,448	7.10 (27)	5.73 (15)	5.61 (14)
1,536	100,925,440	8.30 (28)	6.28 (15)	6.51 (15)

Table 1: Time to solution in seconds: RT_0 on tetrahedral meshes, the corresponding number of PCG iterations are the reported in the brackets.

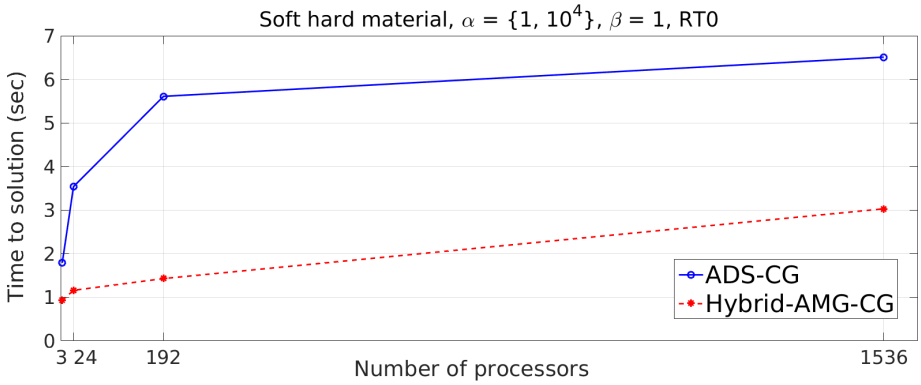
$\alpha = \{1, 10^p\}$, $\beta \equiv 1$		p		
# proc	Problem size	-4	0	4
Hybridization-BoomerAMG-CG				
3	38,400	0.30 (15)	0.31 (16)	0.31 (16)
24	301,056	0.48 (18)	0.50 (21)	0.48 (20)
192	2,383,872	0.75 (28)	0.89 (29)	0.77 (29)
1,536	18,972,672	1.97 (44)	1.95 (47)	2.10 (47)
ADS-CG				
3	38,400	4.85 (23)	3.55 (13)	3.80 (14)
24	301,056	7.24 (29)	5.47 (18)	5.73 (20)
192	2,383,872	11.56 (37)	8.89 (25)	9.56 (28)
1,536	18,972,672	24.28 (53)	16.51 (37)	16.37 (39)

Table 2: Time to solution in seconds: RT_2 on tetrahedral meshes, the corresponding number of PCG iterations are the reported in the brackets.

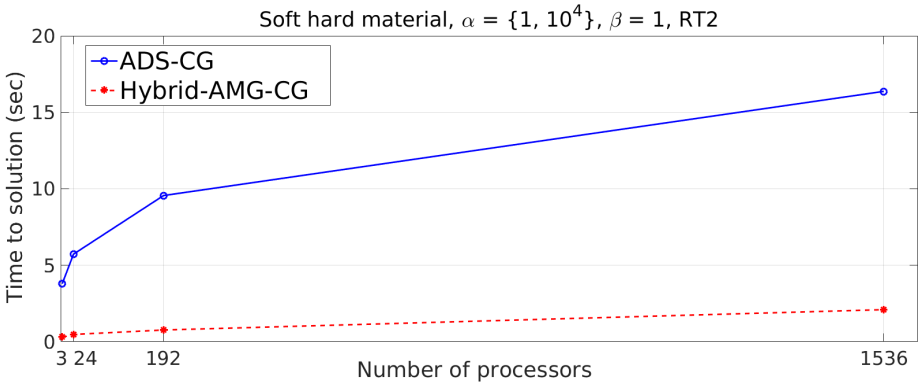
the blue region. The difficulties of this problem are the large jumps of coefficients and the highly stretched elements in the mesh (see figure 3). For this test, the problem discretized by RT_1. The size of A is 2,805,520, and we solve the problem using 4, 8, 16 ,32 and 64 processors. The PCG iteration is stopped when the l_2 norm of the residual is reduced by a factor of 10^{14}. The number of PCG iterations and time to solution are reported in Table 3, and we plot the speedup in figure 4. When measuring the speedup, solution time are corrected by the number of iterations.

Both solvers exhibit good strong scaling. We note that in this example, the solution time of the hybridization AMG solver is much smaller than the ADS solver. The average solve time of the hybridization AMG solver is about 10 times smaller than that of ADS. In particular, the hybridization AMG solver with 4 processors is still 2 times faster than ADS with 64 processors. The difference in the computation time for this example is highly noticeable.

Lastly, we report the time spent on different components of the hybridization approach in Table 4. We observe that except solving with S (i.e. setup and PCG solve), the other components scale fairly well. Also, as we point out in section 3, solving with S is the most time consuming part of the hybridization AMG code. We remark that during the formation of S, we stored the inverses of local blocks of $\widehat{A}$. So when we recover u by back substitution, only matrix multiplication is needed. Hence, the recovery of u is extremely cheap and scalable.



(a) RT_0



(b) RT_2

Figure 2: Weak scaling comparisons between hybridization (red dotted line) and ADS (blue solid line)

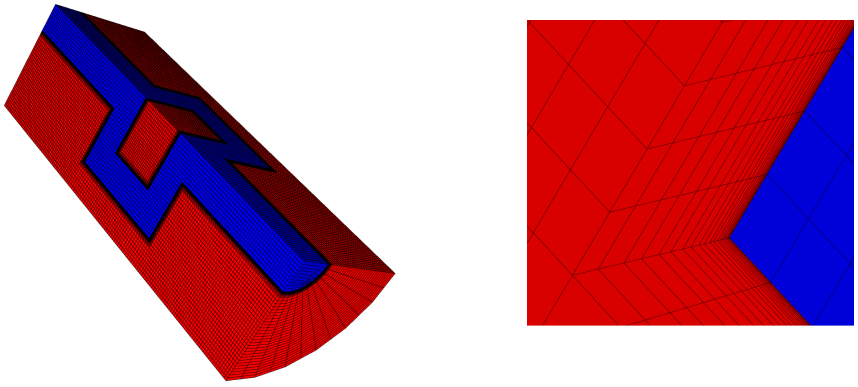


Figure 3: The mesh for the Crooked Pipe problem (left). A dense layer of highly stretched elements (right) has been added to the neighborhood of the material interface in the exterior subdomain in order to resolve the physical diffusion.

5 Concluding Remarks

In this paper, we present a parallel solver for $\boldsymbol{H}(\mathrm{div})$ problems on general unstructured mesh using hybridization and AMG. A symmetric positive definite system for the Lagrange multipliers is first

	Hybridization-BoomerAMG-CG		ADS-CG	
# proc	# iter	time to solution	# iter	time to solution
4	25	23.46	32	508.66
8	31	14.21	32	251.37
16	28	6.83	33	130.26
32	28	3.98	34	73.47
64	31	2.92	34	54.58

Table 3: Strong scaling test, original problem size: 2,805,520.

# proc	formation of S	computation of D	setup	PCG solve	recovery of u
4	7.55	0.22	3.87	11.72	0.092
8	3.95	0.11	2.29	7.81	0.046
16	1.84	0.057	1.4	3.52	0.022
32	1.11	0.034	0.83	2.01	0.012
64	0.68	0.027	0.52	1.7	0.006

Table 4: Timing of each component of the new solver.

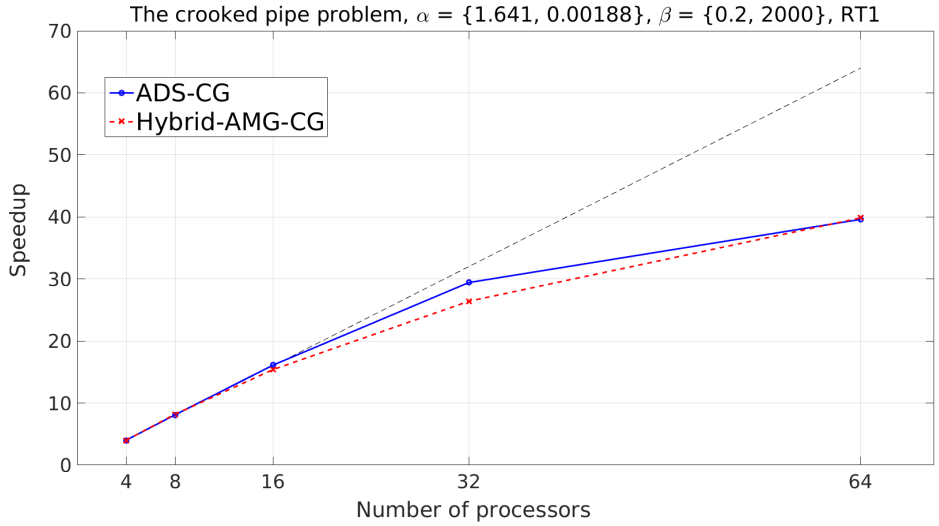


Figure 4: Strong scaling comparison between hybridization (red dotted line) and ADS (blue solid line). Black dotted line indicates perfect scaling

formed by hybridization. Motivated by its relation to the hybridized mixed discretization for second order scalar elliptic problem, we solve this global system by AMG. A rescaling technique was implemented so that the rescaled system has the right scaling suitable for BoomerAMG to work appropriately. We perform several numerical tests on both structured and unstructured meshes in 3D. Our results show that the proposed solver has good weak and strong scalability, and that it is an attractive alternative for solving $\boldsymbol{H}(\mathrm{div})$ problems, especially for high order discretizations. Previously proposed solvers for the problem focus on the treatment of the kernel of the discrete divergence operator. In particular, some discrete curl and gradient operators need to be provided by the user in the construction. On the contrary, the solver proposed in the current paper does not require such additional input from the user. It does require the stiffness matrix to be given in unassembled element-matrix form, which is available during discretization. Although in this

paper we focus on finite element problems discretized by Raviart-Thomas elements, the proposed approach can be applied to other $\boldsymbol{H}(\mathrm{div})$ conforming discretizations like Brezzi-Douglas-Marini elements, Arnold-Boffi-Falk elements [1], or numerically upscaled problems [5, 11].

References

[1] Douglas N. Arnold, Daniele Boffi, and Richard S. Falk, *Quadrilateral H(div) finite elements*, SIAM Journal on Numerical Analysis, 42 (2005), pp. 2429–2451.

[2] Douglas N. Arnold, Richard S. Falk, and Ragnar Winther, *Preconditioning discrete approximations of the Reissner–Mindlin plate model*, RAIRO Modél. Math. Anal. Numér., 31 (1997), pp. 517–557.

[3] Franco Brezzi and Michel Fortin, *Mixed and Hybrid Finite Element Methods*, Springer-Verlag New York, Inc., New York, NY, USA, 1991.

[4] Z. Cai, R. Lazarov, T. A. Manteuffel, and S. F. McCormick, *First-order system least squares for second-order partial differential equations: Part I*, SIAM Journal on Numerical Analysis, 31 (1994), pp. 1785–1799.

[5] E. T. Chung, Y. Efendiev, and C. S. Lee, *Mixed generalized multiscale finite element methods and applications*, SIAM Multiscale Modeling and Simulation, 13 (2015), pp. 338–366.

[6] Bernardo Cockburn and Jayadeep Gopalakrishnan, *A characterization of hybridized mixed methods for second order elliptic problems*, SIAM Journal on Numerical Analysis, 42 (2004), pp. 283–301.

[7] Bernardo Cockburn and Jayadeep Gopalakrishnan, *Error analysis of variable degree mixed methods for elliptic problems via hybridization*, Mathematics of Computation, 74 (2005), p. 16531677.

[8] Clark R. Dohrmann, *A preconditioner for substructuring based on constrained energy minimization*, SIAM Journal on Scientific Computing, 25 (2003), pp. 246–258.

[9] Van Emden Henson and Ulrike Meier Yang, *BoomerAMG: A parallel algebraic multigrid solver and preconditioner*, Applied Numerical Mathematics, 41 (2002), pp. 155 – 177.

[10] Ralf Hiptmair and Jinchao Xu, *Nodal auxiliary space preconditioning in H(curl) and H(div) spaces*, SIAM Journal on Numerical Analysis, 45 (2007), pp. 2483–2509.

[11] D. Kalchev, C. S. Lee, U. Villa, Y. Efendiev, and P. S. Vassilevski, *Upscaling of mixed finite element discretization problems by the spectral AMGe method*, Submitted.

[12] Tzanio V. Kolev and Panayot S. Vassilevski, *Parallel auxiliary space AMG solver for H(div) problems*, SIAM Journal on Scientific Computing, 34 (2012), pp. A3079–A3098.

[13] Panayot S. Vassilevski and Umberto Villa, *A block-diagonal algebraic multigrid preconditioner for the Brinkman problem*, SIAM Journal on Scientific Computing, 35 (2013), pp. S3–S17.

[14] Stefano Zampini, *PCBDDC: a novel class of robust dual-primal preconditioners in PETSc*, submitted.