STUDY OF BDDC

WENTAO YANG *

1. Problem Statement. This study focuses on the balancing domain decomposition with constrains(BDDC) method originally proposed by Dohrmann[1] and it use [2] as a reference. BDDC aims to serve as a preconditioner for iterative solutions of sparse linear systems(often symmetric positive definite) arising from finite element formulation of the discretization of elliptic partial differential equations. BDDC belongs to the class of domain decomposition methods, in particular, the class of iterative substructuring methods with non-overlapping subdomains.

In general, domain decomposition methods try to partition meshes into subdomains that are independent except in the boundaries, therefore the problems can be solved in parallel. Though BDDC follows the idea, it has been modified and improved substantially with many years of research and development in this field. Currently, BDDC and FETI-DP(Finite Element Tearing and Interconnecting Dual Primal) are the two most advanced methods in their own subclasses of domain decomposition methods. Though there exist Adaptive BDDC, which attempts to enhance the robustness in heterogeneous coefcients in the model elliptic problems, and multilevel BDDC, which applies BDDC recursively to solve large problem. The focus of this study remains with the original formulation of BDDC.

After decomposition into subdomains, the interior degrees of freedom inside each subdomains are eliminated through the process called *static condensation*, which converts the original stiffness matrix into *Schur complement* matrix with the condition number greatly reduced. To further improve the condition number and overcome the irregularities of problems, global coarse grid correction, which provides global information exchange, and local Neumann-Neumann preconditioners, which solve Neumann problems in each subdomain on both sides of interface, are added into the method, forming the method what we called balancing domain decomposition(BDD). To address the failure of BDD in certain problems, namely, the fourth order problems such as plates, BDDC were proposed(along with FETI-DP), which reduces the coupling of degrees of freedom that are in different substructures and share different boundaries of the coarse problem and adds another set of constraints in the boundaries to ensure the continuity of degrees of freedom.

After illustrating the background and the motivation behind the algorithm, a pseudo code for BDDC is given below:

1. Given partitioned mesh $\Omega_i, i = 1, 2, 3...N$ and corresponding restriction operator $R_i, i = 1, 2, 3...N$ and stiffness matrix $A_i, i = 1, 2, 3...N$

2. We define W^Γ as the dofs in the boundaries of global domain. In each subdomain A_i, we have ${A_i}^{II}$, ${A_i}^{I\Gamma}$, ${A_i}^{\Gamma\Gamma}$ the stiffness matrix between interior dofs, between interior dofs and boundary dofs, between boundary dofs, respectively. We also define ${W_i}^I$, ${W_i}^\Gamma$ as W^Γ restricted to each subdomain.

3. In each subdomain boundary i, we define a set of dofs as coarse dofs that are required to be continuous across the interface(typically in the corner, may be

*wentaoy2@illinois.edu

chosen randomly or with specific restriction operators) and denote them as ${W_i}^{\Pi}$. We also define ${A_i}^{I\Pi}$, ${A_i}^{\Gamma\Pi}$, ${A_i}^{\Pi\Pi}$, in similar fashion as step 2. We define ${R_i}^{\Gamma\Pi}$ as the restriction from W^{Γ} to W^{Π} and ${R_i}^{\Pi}$ as the restriction from W^{Π} to ${W_i}^{\Pi}$ where W^{Π} is the sum of ${W_i}^{\Pi}$ injected into W^{Γ}.

4. We define $\hat{W}^{\Gamma}$ to be all degrees of freedom on the interface of subdomains that are also continuous across the interface, and in each subdomain i define ${W_i}^{\Delta}$ to be all degrees of freedom on the interface that are zeros in ${W_i}^{\Pi}$. We define R^{Γ} to be the restriction from $\hat{W}^{\Gamma}$ to ${W_i}^{\Gamma}$. We then define $R^{\Gamma\Delta}$ and ${R_i}^{\Delta}$ in the same fashion as step 3. Also, ${A_i}^{\Delta\Delta}$, ${A_i}^{\Delta I}$ can be computed from restriction in similar fashion as step 2,3.

5. Compute the Shur complements of coarse grid correction in each subdomain by $S_i^{\Pi\Pi} = A_i^{\Pi\Pi} - \begin{bmatrix} A_i^{\Pi I} & A_i^{\Pi\Delta} \end{bmatrix} \begin{bmatrix} A_i^{II} & A_i^{\Delta I T} \\ A_i^{\Delta I} & A_i^{\Delta\Delta} \end{bmatrix}^{-1} \begin{bmatrix} A_i^{\Pi I T} \\ A_i^{\Pi\Delta T} \end{bmatrix}$

6. Compute the solutions for subdomain correction in each subdomain i by $\Psi_i = \begin{bmatrix} \Psi_i^{\Delta} \\ R_i^{\Pi} \end{bmatrix} = \begin{bmatrix} -\begin{bmatrix} 0 & R_i^{\Delta} \end{bmatrix} \begin{bmatrix} A_i^{II} & A_i^{\Delta I T} \\ A_i^{\Delta I} & A_i^{\Delta\Delta} \end{bmatrix}^{-1} \begin{bmatrix} A_i^{\Pi I T} \\ A_i^{\Pi\Delta T} \end{bmatrix} R_i^{\Pi} \\ R_i^{\Pi} \end{bmatrix}$ (Note that the other formulation intends to solve a primal problem with Lagrange multiplier.)

7. Form $\Psi = \begin{bmatrix} \Psi_1 \\ \Psi_2 \\ \dots \\ \Psi_N \end{bmatrix}$ and $S^{\Pi\Pi} = \sum_{i=1}^{N} {R_i}^{\Pi T} S_i^{\Pi\Pi} R_i^{\Pi}$

8. Form Schur complement operator in each subdomain i as S_i^{Δ} such that given any u_i, $\begin{bmatrix} A_i^{II} & A_i^{\Delta I T} \\ A_i^{\Delta I} & A_i^{\Delta\Delta} \end{bmatrix} \begin{bmatrix} {u_i}^I \\ {u_i}^{\Delta} \end{bmatrix} = \begin{bmatrix} 0 \\ S_i^{\Delta} {u_i}^{\Delta} \end{bmatrix}$ hold true. Compute S^{Δ} as a direct sum of all S_i^{Δ} and compute its inverse.

9. Introduce positive scaling factor $\delta_i(x)$ for any node x on the interface of each subdomain that satisfies $\sum_{j \in N_x} \delta_i(x) = 1$ where N_x is the number of subregions that contain x. (See [1] for better chracterization for this)

10. Apply $\delta_i(x)$ to each node in ${R_i}^{\Delta}$ and ${R_i}^{\Gamma}$ to get ${R_i}^{D\Delta}$ and ${R_i}^{D\Gamma}$, respectively.

11. Compute $M_{BDDC}^{-1} = {R^{D\Gamma}}^T ({R^{\Gamma\Delta}}^T S^{\Delta -1} R^{\Gamma\Delta} + \Psi S^{\Pi\Pi -1} \Psi^T) R^{D\Gamma}$

Finally, current theory[2] suggests that the condition number of BDDC is bounded by the product of the tolerance and a constant that only depends on the number of faces per subdomain, the number of edges per face and the number of subdomains sharing an edge.

2. Approach. To study the practical performance of the method. We use the implementation of BDDC in the NGSolve library[3] and implement the following numerical experiments. The first problem we study is a Poisson problem defined in a cube with the following variational form with homogeneous dirichlet conditions defined on the left and bottom surfaces,

$find : u \in H^1_{0,d}, s.t \int_{\Omega} \nabla u \nabla v = \int_{\Omega} f v, \forall v \in H^1_{0,d}$

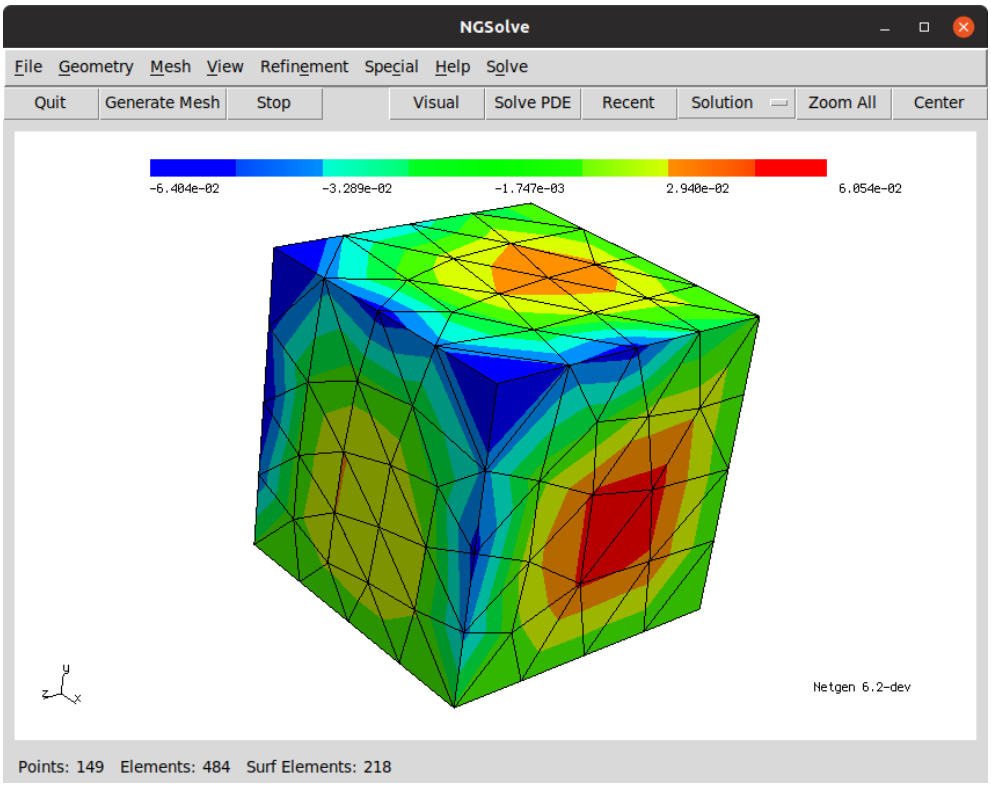


Fig. 1.1: Cube with Poisson

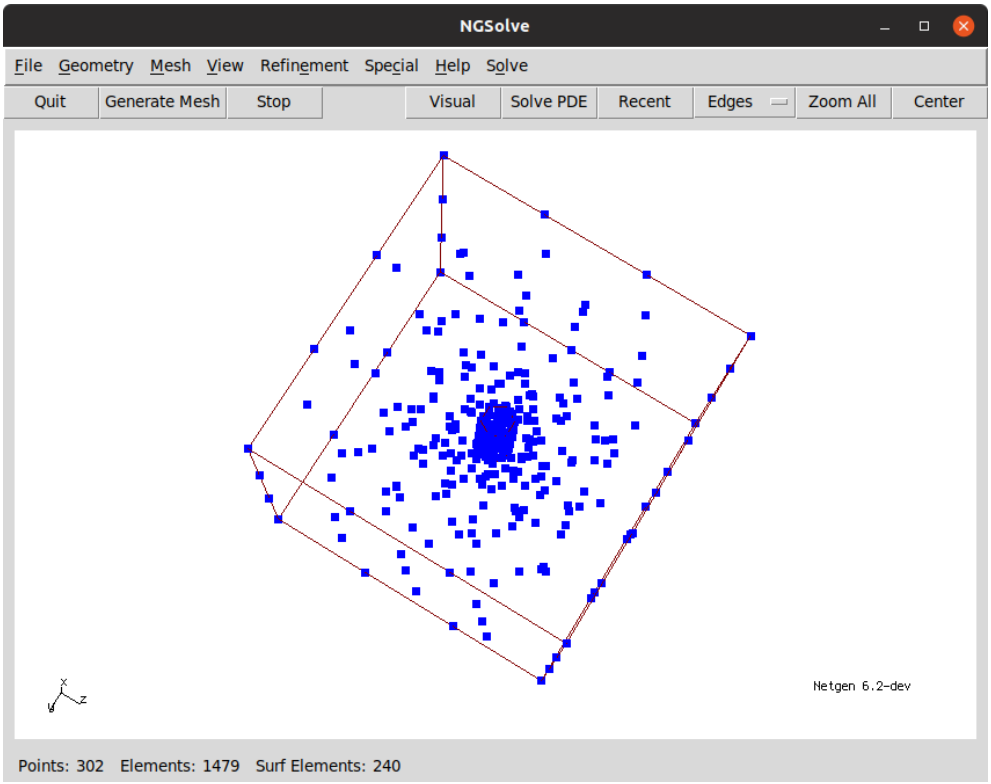


Fig. 2.1: Magnet in a Cube

To show the convergence of BDDC compared with other preconditioners, we test BDDC, Jacobi, Block-Jacobi, Symmetric-Gauss-Seidel and Geometric Multigrid as preconditioners for Conjugate-Gradient solver in this problem and we increase the order of polynomial of the finite element space and measure the iteration required to reach tolerance. Since for finite element space of order one and two, BDDC solves the problem in one iteration, we therefore plot the absolute error in each iteration of these preconditioners when the order is three. Then, we try adaptive refinement for multilevel multigrid and BDDC as specified by [3]. Figure 1.1 shows the solution on the cube mesh.

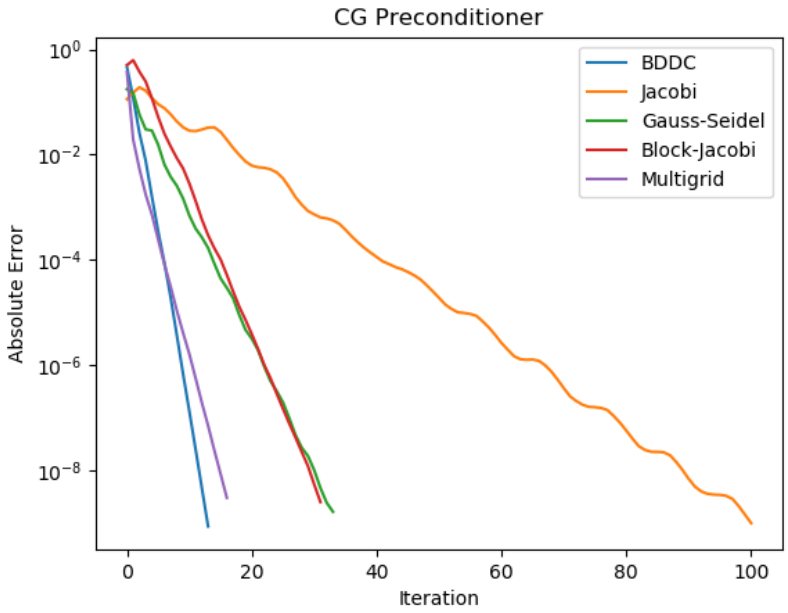
To show the performance of BDDC in solving more complex problem, the second problem we test is the well-known Maxwell equation in finite element setting with a magnet placed in the center of a cube. 2.1 shows the mesh and it is clearly to see that there are two partitions. Below is the formulation of the problem.

Given B as magnetic flux, magnetic field H, and magnetization M, We have $B = \rho(H + M), divB = 0, curlH = 0$ write it the weak form:

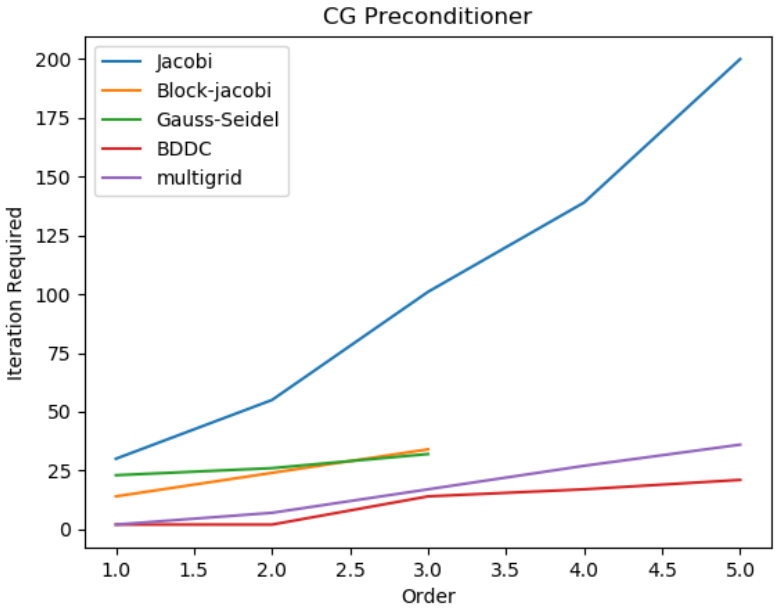
$find : A \in H(curl), B = curl(A), s.t \int \rho^{-1} curlAcurlv = \int Mcurlv, \forall v \in H(curl)$

We note that all the experiments above are in serial. To illustrate the BDDC's ability of parallelism, as this is the original purpose of Domain Decomposition method, we use the MPI implementation of NGSolve to solve a Poisson problem defined in a cube that is very similar to the first test. We use four processes as well as multi threading and compare Jacobi Preconditioner with BDDC Preconditioner. Because of the limitation of hardware, we could not show the scaling results, but it can be done easily.

3. Numerical Results. First, for all geometric multigrid results, we used direct-solve in the coarsest level and gauss-seidel in refined level. For the first test, Figure 3.1(a) shows the convergence for five preconditioners with order three with 2708 degrees of freedom, it is obvious to see that the performance of Multigrid and BDDC are better than other preconditioners. Figure 3.1(b) shows the convergence as the order of finite element space grows. We note that as the order grows from one to five, the number of degrees of freedom also grows like 149, 896, 2708, 6087, 11511, respectively. Also, Gauss-Seidel and Block-Jacobi no longer converge if the order is greater than 3 and BDDC becomes much better than multigrid as the order grows large. Adaptive Refinement is shown by 3.2 with level 3, interestingly, though multigrid and BDDC are similar when there is only one level, but with adaptive refinement, BDDC outperforms multigrid. Note that 3.2 only shows absolute error, while the numbers of degrees of freedom are 2708, 19464, 147395, respectively for 3 levels. Finally, we computed that, for this problem, the condition number of BDDC is 2.68923, while the condition number of Jacobi is 156.95125.



(a) iteration vs. error when order = 3



(b) order vs. iteration required to fully converge

Fig. 3.1: Poisson in Cube

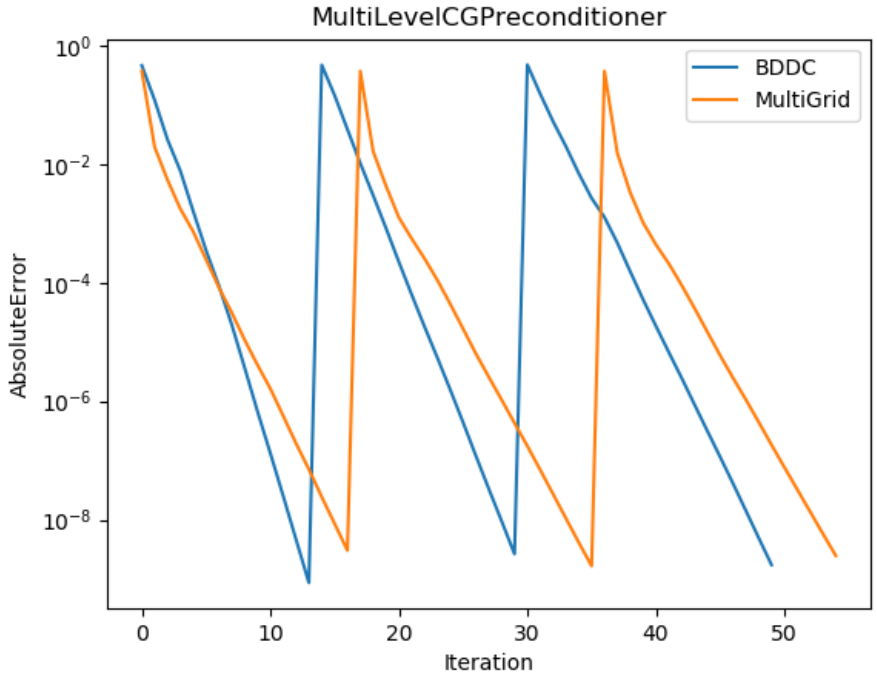


Fig. 3.2: Refinement

For the second test, the result is shown by Figure 3.3. There are 2 partitions, 400 points, 2264 elements and the total number of degrees of freedom is 21439. The condition number is 20.014 for BDDC, and 86269.207 for Jacobi, so the reuslt shown in the graph is expected. We note that in this problem multiigrid performs better than BDDC.

For the last test, we show the final solution and the mesh by 3.4. There are 4 partitions, 1781 points, 10157 elemnst, and 35035 degrees of freedom. From 3.5, we see that each process of BDDC takes about 14 iterations to converge, while each process of Jacobi takes more than 100 iterations. Overall, BDDC is much better than Jacobi.

From all the experiments above, we see that the advantages of BDDC are that it has comparable performance as multigrid, it performs very well as the order of finite element space grows large, and it is one of the best methods in solving the target problems in parallel. The disadvantages are that it requires information of the mesh and geometry of the problem, it is complicated and expensive to form, and it is costly to apply compared to multigrid if there is no parallelism.

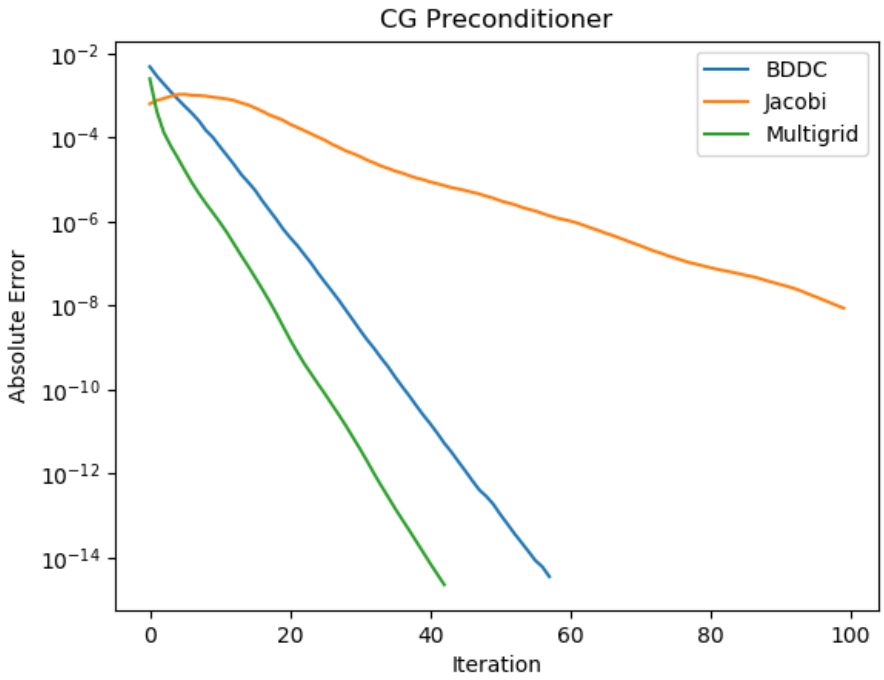


Fig. 3.3: Maxwell

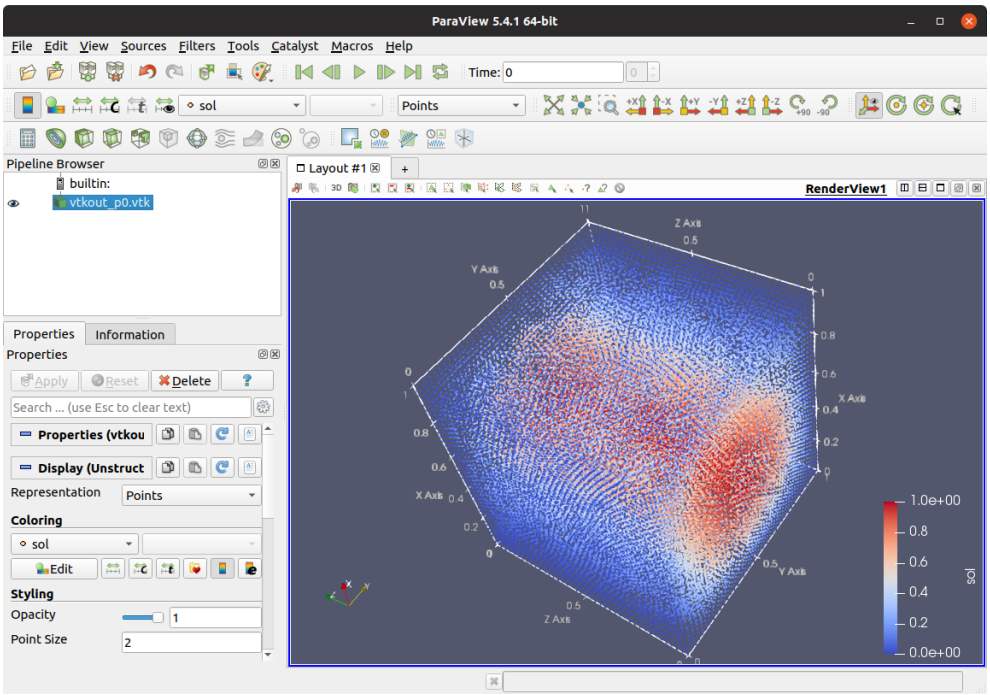


Fig. 3.4: Parallel

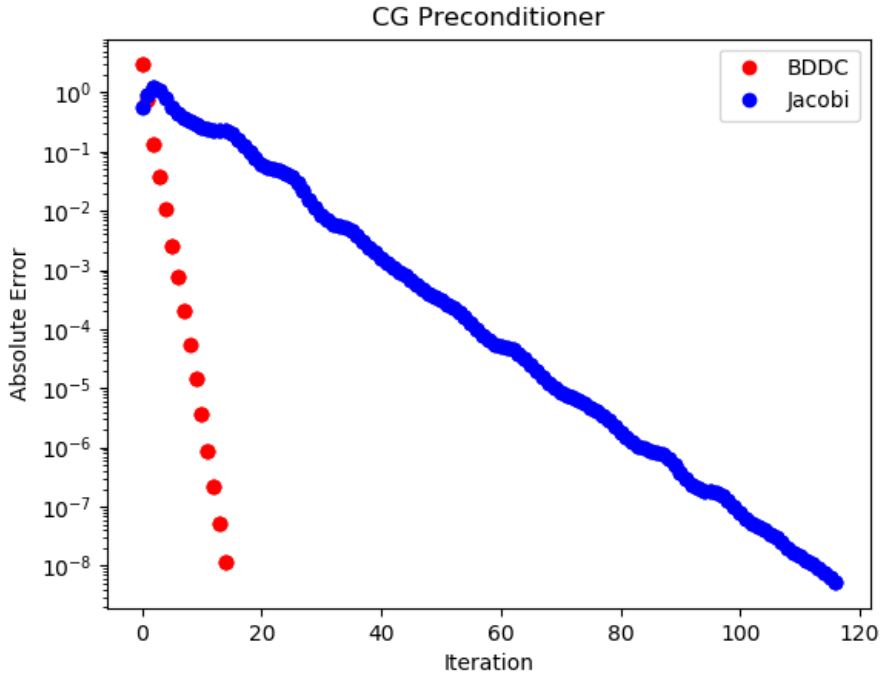


Fig. 3.5: Parallel

4. Conclusions. In this study, we review the algorithm to form the BDDC preconditioner and give the history and some motivations behind it. We then test three problems and compare it with four different preconditioners and show its good performance.

[1]Dohrmann CR (2003) A preconditioner for substructuring based on constrained energy minimization. SIAM J Sci Comput 25(1):246258

[2]Hyun Kim, Hyea Chung, Eric Junxian, Wang. (2018). BDDC and FETI-DP algorithms with a change of basis formulation on adaptive primal constraints. ETNA - Electronic Transactions on Numerical Analysis. 49. 64-80. 10.1553/etna$_v ol49s64$.

[3]https://ngsolve.org/