Implementation and Tests of Multigrid Preconditioned Conjugate Gradient Method

Yuexia Luna Lin
Rycroft Group, SEAS, Harvard University
y_lin@g.harvard.edu

May 15, 2018

1 Introduction

Multigrid is an iterative method particularly suited for solving large sparse linear systems. It applies smoothers such as Jacobi and Gauss-Seidel methods on a hierarchy of grids of a range of spatial resolutions. This is combined with procedures called restrictions and interpolations to transfer data across the hierarchy of grids.

When solving general elliptic PDEs with variable coefficients, such as

$$\nabla(c(x)\nabla u(x)) = f(x) \tag{1}$$

if the variations in $c(x)$ have large discontinuities, according to [1], parts of the multigrid method need to be changed to ensure efficiency. For example, interpolation and discretized derivatives should be modified to avoid applying the operators across discontinuities.

Here we explore an alternative way of applying multigrid to solve Eq. 1 that uses multigrid as a preconditioner for the conjugate gradient (CG) method. By applying multigrid as a preconditioner, we can effectively cluster the eigenvalues and move them away from zero, therefore lower the condition number of the coefficient matrix [3]. In doing so, we achieve faster convergence than using either multigrid or conjugate gradient alone.

In the following sections, we will first show that using symmetric lexicographic Gauss-Seidel in a multigrid solve is compatible with the requirements of a preconditioner for conjugate gradient method. Next we will briefly review the source term, boundary conditions, and the numerical scheme used in solving the test problem Eq. 1. Finally, we present sample solutions and compare number of iterations and time needed by multigrid alone and multigrid preconditioned conjugate gradient (MPCG).

1.1 Preconditioned CG

In the report, we will denote vectors by lower case letters, matrices by bolded upper case letters, scalars by greek letters.

In precondition CG, instead of solving $\mathbf{A}x = b$, we solve the equation

$$\mathbf{MA}x = \mathbf{M}b \tag{2}$$

We will show that if matrix $\mathbf{M}$ is symmetric and positive definite (SPD), even though $\mathbf{MA}$ is not necessarily SPD, we can rewrite Eq. 2 such that the matrix multiply x on the LHS is SPD [2].

First we can decompose $\mathbf{M} = \mathbf{E}\mathbf{E}^\mathsf{T}$. Suppose v is an eigenvector of the matrix $\mathbf{MA}$ with eigenvalue λ. We then manipulate the equation as follows,

$$\mathbf{MA}v = \lambda v = \mathbf{E}\mathbf{E}^\mathsf{T}\mathbf{A}v = \mathbf{E}\mathbf{E}^\mathsf{T}\mathbf{A}(\mathbf{E}\mathbf{E}^{-1})v = \mathbf{E}(\mathbf{E}^\mathsf{T}\mathbf{A}\mathbf{E})\mathbf{E}^{-1}v$$

Apply $\mathbf{E}^{-1}$ to the left most ad right most equation, we have

$$\begin{aligned}\mathbf{E}^{-1}\mathbf{MA}v &= \lambda\mathbf{E}^{-1}v \\ &= \mathbf{E}^{-1}\mathbf{E}(\mathbf{E}^\mathsf{T}\mathbf{A}\mathbf{E})\mathbf{E}^{-1}v = (\mathbf{E}^\mathsf{T}\mathbf{A}\mathbf{E})\mathbf{E}^{-1}v\end{aligned}$$

Hence, $\mathbf{E}^\mathsf{T}\mathbf{A}\mathbf{E}$ has the same eigenvalues as $\mathbf{MA}$, with eigenvectors $\mathbf{E}^{-1}v$. We can rewrite Eq. 2 as

$$(\mathbf{E}^\mathsf{T}\mathbf{A}\mathbf{E})\hat{x} = \mathbf{E}^{-1}\mathbf{E}\mathbf{E}^\mathsf{T}b = \mathbf{E}^\mathsf{T}v \tag{3}$$

where $\hat{x} = \mathbf{E}^{-1}x$. Now we again have a SPD matrix, $\mathbf{E}^\mathsf{T}\mathbf{A}\mathbf{E}$, on the LHS, and we can apply conjugate gradient to Eq. 3. However, we would like to not have to compute $\mathbf{E}$. By a change of variable, we can go back to using $\mathbf{M}$ directly. For the sake of brevity, the derivation is omitted.

The task now is to show using symmetric lexicographic Gauss-Seidel in multigrid results in a SPD matrix $\mathbf{M}$.

1.2 Symmetric Gauss-Seidel

To use Gauss-Seidel to solve the equation

$$\mathbf{A}x = b$$

we split the SPD matrix $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{D} - \mathbf{L} - \mathbf{U}$$

where $\mathbf{D}$ is the diagonal matrix with diagonal elements from $\mathbf{A}$; $-\mathbf{L}$ and $-\mathbf{U}$ are the strictly lower and upper matrices of $\mathbf{A}$. A forward sweep of Gauss-Seidel is the following iterative procedure

$$
\begin{aligned}
(\mathbf{D}-\mathbf{L})x &= \mathbf{U}x + b \\
x^{n+1} &= (\mathbf{D}-\mathbf{L})^{-1}\mathbf{U}x^n + (\mathbf{D}-\mathbf{L})^{-1}b
\end{aligned}
$$

Similarly, a backward sweep is

$$
x^{n+1} = (\mathbf{D}-\mathbf{U})^{-1}\mathbf{L}x^n + (\mathbf{D}-\mathbf{U})^{-1}b
$$

A lexicographic order Gauss-Seidel sweep (SLGS) is a forward sweep, followed by a backward sweep, i.e.

$$
\begin{aligned}
x^{n+1} &= (\mathbf{D}-\mathbf{U})^{-1}\mathbf{L}\left((\mathbf{D}-\mathbf{L})^{-1}\mathbf{U}x^n + (\mathbf{D}-\mathbf{L})^{-1}b\right) + (\mathbf{D}-\mathbf{U})^{-1}b \\
&= (\mathbf{D}-\mathbf{U})^{-1}\mathbf{L}(\mathbf{D}-\mathbf{L})^{-1}\mathbf{U}x^n + (\mathbf{D}-\mathbf{U})^{-1}\left(\mathbf{L}(\mathbf{D}-\mathbf{L})^{-1}+\mathbf{I}\right)b
\end{aligned}
$$

Simplify the second term in the above equation, we have

$$
\begin{aligned}
&(\mathbf{D}-\mathbf{U})^{-1}\left(\mathbf{L}(\mathbf{D}-\mathbf{L})^{-1}+\mathbf{I}\right)b \\
&= (\mathbf{D}-\mathbf{U})^{-1}\left(\mathbf{L}(\mathbf{D}-\mathbf{L})^{-1} + (\mathbf{D}-\mathbf{L})(\mathbf{D}-\mathbf{L})^{-1}\right)b \\
&= (\mathbf{D}-\mathbf{U})^{-1}(\mathbf{L}+\mathbf{D}-\mathbf{L})(\mathbf{D}-\mathbf{L})^{-1}b \\
&= (\mathbf{D}-\mathbf{U})^{-1}\mathbf{D}(\mathbf{D}-\mathbf{L})^{-1}b
\end{aligned}
$$

Hence,

$$
x^{n+1} = (\mathbf{D}-\mathbf{U})^{-1}\mathbf{L}(\mathbf{D}-\mathbf{L})^{-1}\mathbf{U}x^n + (\mathbf{D}-\mathbf{U})^{-1}\mathbf{D}(\mathbf{D}-\mathbf{L})^{-1}b
$$

Now if we consider a generic splitting of matrix $\mathbf{A} = \mathbf{P} - \mathbf{Q}$, an iterative method can be expressed as

$$
x^{n+1} = \mathbf{P}^{-1}\mathbf{Q}x^n + \mathbf{P}^{-1}b
$$

Let's now derive what the expressions are for $\mathbf{P}$ and $\mathbf{Q}$ in SLGS. We observe that

$$
\mathbf{P}^{-1} = (\mathbf{D}-\mathbf{U})^{-1}\mathbf{D}(\mathbf{D}-\mathbf{L})^{-1}
$$

Simplify, we have,

$$
\begin{aligned}
\mathbf{P} &= \left[(\mathbf{D}-\mathbf{U})^{-1}\mathbf{D}(\mathbf{D}-\mathbf{L})^{-1}\right]^{-1} \\
&= (\mathbf{D}-\mathbf{L})\mathbf{D}^{-1}(\mathbf{D}-\mathbf{U}) \\
&= (\mathbf{D}-\mathbf{L})(\mathbf{I}-\mathbf{D}^{-1}\mathbf{U}) \\
&= \mathbf{D}-\mathbf{L}-\mathbf{U}+\mathbf{L}\mathbf{D}^{-1}\mathbf{U} \\
&= \mathbf{A}+\mathbf{L}\mathbf{D}^{-1}\mathbf{U}
\end{aligned}
$$

Note that $\mathbf{P}$ is symmetric, because $(\mathbf{D}-\mathbf{L})^{\mathsf{T}} = (\mathbf{D}-\mathbf{U})$.

This means

$$\mathbf{Q} = \mathbf{P} - \mathbf{A} = \mathbf{L}\mathbf{D}^{-1}\mathbf{U}$$

$\mathbf{Q}$ is also symmetric.

1.3 Multigrid with symmetric lexicographic ordering Gauss-Seidel (SLGS)

Following the notations in [3], we denote the linear equation at the l^{th} level as

$$\mathbf{A}_l x_l = b_l$$

Consider a two-grid system, where the finer grid is smoothed using SLGS, while the coarser grid is solved exactly. Smoothing is applied both before and after solving the equation on the coarser grid. We start from an initial guess of a zero vector. Restriction operator is defined as matrix $\mathbf{R}$, and interpolation is defined as the transpose of restriction, i.e. $\mathbf{T} = \mathbf{R}^{\mathsf{T}}$.

Applying this two grid system in one v-cycle results in the following matrix

$$\mathbf{M} = \mathbf{H}^m\mathbf{J} + \mathbf{J} + \mathbf{H}^m\mathbf{T}\mathbf{A}_{l-1}^{-1}\mathbf{R}(\mathbf{I} - \mathbf{A}_l\mathbf{J}) \tag{4}$$

where

$$\mathbf{H} = \mathbf{P}^{-1}\mathbf{Q}; \quad \mathbf{J} = \sum_{i=0}^{m-1} \mathbf{H}^i\mathbf{P}^{-1}$$

According to [3], we have the following theorems,

Theorem 1.
Assume matrix $\mathbf{A}_{l-1}^{-1}$ is symmetric and positive definite. If matrix $\mathbf{P}$ is symmetric, then matrix $\mathbf{M}$ in Eq. 4 is symmetric, positive definite.
Theorem 2.
If assumptions of Theorem 1 are satisfied, all multigrid methods with n cycles and m smoothing iterations with $m, n \geq 1$ satisfy conditions of a preconditioner of the conjugate gradient method.

We have shown that $\mathbf{P}$ for SLGS is symmetric, hence we have also shown that multigrid with SLGS can be used a preconditioner for CG.

2 Test problems and results

2.1 Model Poisson problem

We test MPCG on the 3D Poisson equation

$$\nabla(c(x)\nabla u(x)) = f(x) \qquad \text{in } \Omega = [0,1]^3 \tag{5}$$

• $c(x)$ is the diffusion constant, which can be uniform or vary spatially.

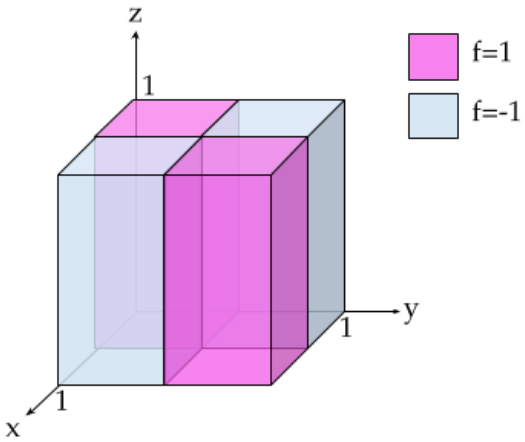
• The source term $f(x)$ is the same for all test cases, and it is described in Fig. 1a.

• Dirichlet, Neumann, and periodic boundary condition are implemented. However, results for homogeneous Neumann and periodic boundary conditions are reported.

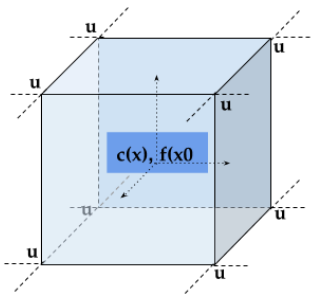
• Finite element formulation is used to discretize the equation. We use piecewise trilinear basis functions to represent $u(x)$, piecewise constant basis function for $f(x)$ and $c(x)$.

• Discretized values of u_{ijk} are at the cell corners, source f_{ijk} and diffusion constant c_{ijk} are at the cell centers. See Fig. 1b for arrangement of variable on the grid.

• A multigrid solve with one v-cycle using SLGS is used as preconditioner for MPCG. Both multigrid and MPCG algorithms are terminated when the sum of the squares of the residuals reaches the tolerance of 5×10^{-24}.



(a) Source term for 3D Poisson test problem.



(b) Variable arrangement on the grid.

Figure 1: Schematics of source term for the Poisson problem (left) and variable arrangement (right).

2.2 Results

In this subsection, all of the tests are run on two MPI processes. Distributing work on multiple MPI processes leads to slight differences in the solutions, when compared to those computed on a single MPI process. This is because after domain decomposition, carrying out SLGS sweeps in parallel in each of the subdomains changes whether or not updated values are available to a given grid point in a sweep. It has not been observed that slight differences due parallelization change the general convergence behavior of either method.

1. **Uniform** $c(x)$

   We run tests on grid sizes 51^3, 101^3, and 151^3. Fig. 2 shows cross-sections at $z = 0.5$ of example solutions, with different boundary conditions. Note that since we need to run one multigrid solve to initialize the MPCG iterations, the total number of v-cycles in MPCG is one more than the number of iterations.

   (a) **Homogeneous Neumann BC**

   Number of iterations and timing results are tabulated in Table 1.

   Since uniform diffusion constant is a simple case for both methods, we observe only a small difference between the two methods. MPCG typically takes 1-2 fewer v-cycles than multigrid alone.

N	MG v-cycles	MPCG iterations	MG time (s)	MPCG time (s)
51	9	6	0.374	0.307
101	8	6	1.97	1.79
151	8	5	6.59	5.13

Table 1: Tests with homogeneous Neumann BC and uniform $c(x)$.

   (b) **Periodic BC**

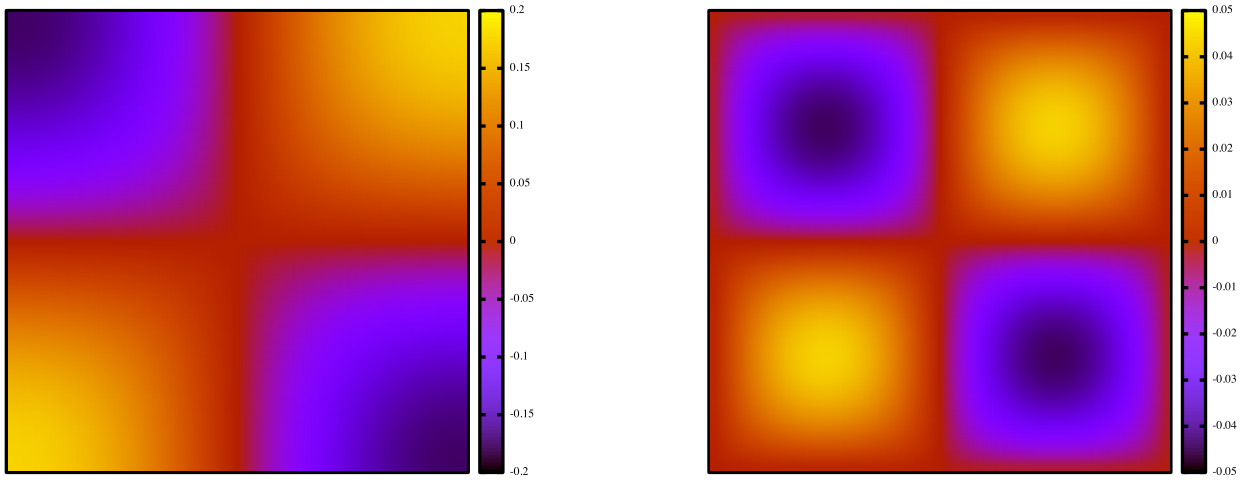
   Results are tabulated in Table 2.

   When running the test for $N = 101$, the number of iterations needed by either methods seem atypically high. Additional tests with $N = 91$ and $N = 111$ verify that $N = 101$ is in fact an atypical case.

   For $N = 101$, each v-cycle reduces the error less, compared to the other similar grid sizes. This could be due to odd sizes of the lower level grids for $N = 101$, and how they affect restrictions and interpolations. For $N = 101$, the lowest two levels have grid sizes 13 and 7, whereas for $N = 91$, the lowest two levels have grid sizes 12 and 6, and for $N = 111$, 14 and 7. This issue would be discussed further in the last section.

N	MG v-cycles	MPCG iterations	MG time (s)	MPCG time (s)
51	8	6	0.315	0.294
91	9	6	1.72	1.35
101	19	10	5.0	2.97
111	9	6	3.01	2.43
151	9	6	7.46	6.24

Table 2: Tests with periodic boundary condition and uniform $c(x)$.



(a) Homogeneous Neumann BC, $N = 51$.

(b) Periodic BC, $N = 51$.

Figure 2: Example solution to the Poisson problem, taken at a cross-section at $z = 0.5$.

2. **Varying** $c(x)$

We initialize a spherical region with radius $r = 0.25$, centered at $(0.5, 0.5, 0.5)$, with a uniform diffusion constant c_{in}, and initialize the region outside with another constant value c_{out}. We then vary the ratio between the densities inside and outside the spherical region, i.e. $\delta = \frac{c_{in}}{c_{out}}$.

To simplify, we fix grid size in the following tests to be 111^3. Fig. 3 shows cross-sections at $z = 0.5$ of the solutions for $\delta = 100$ with different boundary conditions.

(a) **Homogeneous Neumann BC**

Number of iterations and timing results are tabulated in Table 3.

As δ increases, the advantage of MPCG has over multigrid alone becomes apparent. For very sharp transitions in diffusion constant, e.g. $\delta = 100,\ 500$, MPCG takes only half as many iterations as multigrid. Similar results are observed for tests with periodic boundary condition.

δ	MG v-cycles	MPCG iterations	MG time (s)	MPCG time (s)
4	9	6	2.93	2.43
20	14	9	4.79	3.53
100	20	10	6.55	3.8
500	26	12	8.67	4.61

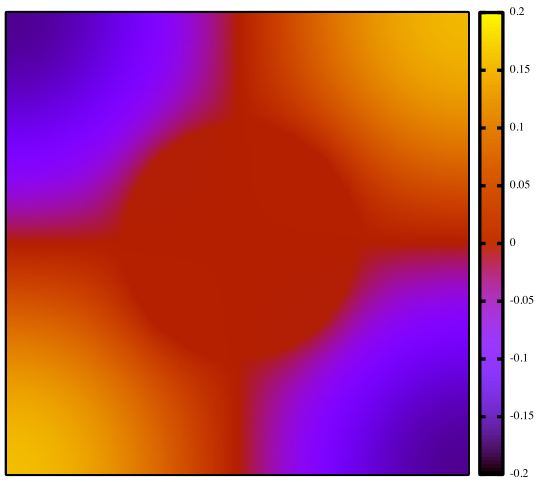
Table 3: Tests with homogeneous Neumann boundary condition and varying $c(x)$.

(b) **Periodic BC**

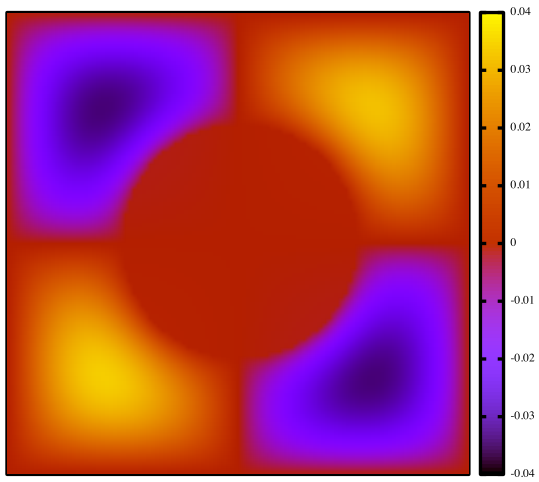
Results are tabulated in Table 4. Discussion see above.

δ	MG v-cycles	MPCG iterations	MG time (s)	MPCG time (s)
4	9	6	3.14	2.48
20	13	8	4.46	3.24
100	18	10	6.18	3.91
500	25	11	8.48	4.32

Table 4: Tests with periodic boundary condition and varying $c(x)$.



(a) Homogeneous Neumann BC, $N = 111$.



(b) Periodic BC, $N = 111$.

Figure 3: Example solution to the Poisson problem taken at a cross-section at $z = 0.5$. The diffusion constant $c_{in} = 100$ in the spherical region of radius 0.25, centered at $(0.5, 0.5, 0.5)$. Outside of that region, $c_{out} = 1$.

3 Conclusion and Observation

In conclusion, a multigrid preconditioned conjugate gradient method is implemented and tested on solving 3D Poisson equation with spatially varying diffusion constant, $c(x)$.

In some test problems, the discontinuities in the diffusion constant are quite large. The ratio of $c(x)$, i.e. δ, across a grid point is as high as 500. In these tests, using MPCG takes only half as many iterations as using multigrid alone.

Problems with even larger coefficient ratios, i.e. $\delta = 650, 800, 1000$, are also tested. For these problems, cases with periodic boundary condition converge. The number of iterations required by multigrid and MPCG is consistent with what we have observed before.

However, test cases with Neumann boundary condition do not converge, regardless of methods used. Using multigrid alone, we observe numerical instability; the residual grows to infinity after dozens of iterations. Using MPCG, the residual does not grow to infinity, but neither does it get as low as the tolerance. It decreases first, then remains approximately the same after dozens of iterations. I suspect the instability and failure to converge issues are due to multigrid not adapting interpolation and difference operators to accommodate the discontinuities.

We also observe an interesting case where a particular grid size, e.g. $N = 101$, takes an atypically greater number of iterations to converge compared to similar grid sizes. We note that the atypical case has two odd size grids at two bottom levels, while the other systems with similar grid sizes only have one odd size grid at the bottom most level.

To explore the effect of odd grid sizes, we run several other tests, with different types of boundary conditions and various system sizes. We find that test cases with certain system size do not converge at all, for instance grid size $N = 80$ with homogeneous Neumann boundary condition does not converge, instead the residual remains the same after the 4th iteration. This likely has something to do with the known issue in the multigrid solver with interpolations and restrictions when they are applied to grids with an odd grid size.

These issues require further, and careful investigation, however, due to time constraints, they were not further studied in this project, but will be explored and addressed in future work.

4 Code

The code for this project is in `esim/mpcg` directory. The `README` file contains descriptions on how to use the main routine in `mpcg_test.cc`.

References

[1] A. Brandt and O. E. Livne, *Multigrid Techniques: 1984 Guid with Applications to Fluid Dynamics*, SIAM, 1984.

[2] J. R. Shewchuk, *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, School of Computer Science, Carnegie Mellon University, 1994.

[3] Osamu Tatebe, *The Multigrid Preconditioned Conjugate Gradient Method,* The Sixth Copper Mountain Conference on Multigrid Methods, Part 2 p621-634, NASA, Langley Research Center, 1993.