UE2 dist-linalg project: Geometric Multigrid in 2D

Abhishek Purandare   Rached Chaaben

January 2023

https://git.sophia.mines-paristech.fr/abhishek.purandare/ue2-dist-linalg-multigrid-2d.git

1 Problem Statement

Given

Solving the following 2D Poisson's equation (isotropic),

$$-\Delta u(x,y) + \sigma u(x,y) = f(x,y) \; in \, \Omega \quad \sigma \geq 0$$

$$\Delta u(x,y) = 0 \; in \, \delta\Omega$$

The approach is taken on a uniform grid with a unit distance h between each grid point i.e $\Delta x = \Delta y$. The 2D Laplace operator for the above equation is,

$$B = \begin{bmatrix} 4+\sigma h^2 & -1 & \\ -1 & 4+\sigma h^2 & -1 \\ & -1 & 4+\sigma h^2 \end{bmatrix} \quad and \quad I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

We obtain the following system $Au = f$ from the discritization of the PDE,

$$\frac{1}{h^2} \begin{bmatrix} B & -I & & & \\ -I & B & -I & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -I \\ & & & -I & B \end{bmatrix} \cdot \begin{bmatrix} u_{11} \\ u_{12} \\ \vdots \\ u_{1n} \\ \vdots \\ u_{nn} \end{bmatrix} = \begin{bmatrix} f_{11} \\ f_{12} \\ \vdots \\ f_{1n} \\ \vdots \\ f_{nn} \end{bmatrix}$$

where n is the number of inner points over a single dimension of the grid, and $N = (n+1)^2$

Geometric Multigrid

Instead of solving the linear system by iterative methods that may take long time to converge due to the low frequency components of the error, Multigrid tends to apply the discritization process in a hierarchical way where it recursively coarsens the fine grid system to a point where solving the system directly on the coarse grid is negligible. By coarsening, low frequency components act like high frequency, and thus it becomes easier to converge.

Let $Au = f$ be the linear system with an exact solution u. Let $\bar{u}$ be the approximate solution such that the error is $e = u - \bar{u}$. Multiplying that by A we get, $Ae = Au - A\bar{u}$ i.e $Ae = f - A\bar{u}$. The quantity $f - A\bar{u} = r$ is a residual. Thus, if we solve a linear system, $Ae = r$, we obtain the exact solution, $u = \bar{u} + e$. The error is also known as the correction, as we are correcting the approximate solution to get a better result. For a linear system, solving this residual equation $Ae = r$ is equivalent to solving the original system. By transferring the fine grid to a coarse grid, we can obtain this residual equation relatively faster rather than solving the system iteratively or directly.

2 Implementation

Restriction and Prolongation

Let h be the unit for fine grid points and $H = 2h$ be the unit for coarse grid points. Then, the restriction operator I_h^H, transfers the fine grid to a coarse grid whereas the prolongation operator I_H^h does the interpolation from coarse to fine. For this approach, we chose Galerkin operator which is defined on a fine grid matrix A_h as

$$A_H = I_h^H A_h I_H^h \quad where\ I_H^h = \frac{1}{c}(I_h^H)^T$$

To first build the restriction operator in 2D, we extend 1D grid restriction operator to 2D by applying Kronecker product of 1D restriction on itself. We used different weighting schemes when performing the coarsening. Figure 1, shows the restriction matrix, and how the weights are distributed in the neighborhood of a grid point. Prolongation operator is built as a transpose of the restriction operator.

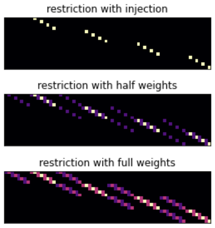

Figure 1: $I_h^H N = 64$

Relaxation

Relaxation applies smoothing by solving the system for few iterations using methods like JOR, and SOR. This serves as an initial guess to speed up the convergence of the coarse grid solution, and also removes the high frequency components of the error. Relaxation takes place as a pre-smoothing and post-smoothing for coarse grid solvers. To test the convergence for different sine waves we set $\sigma = 0$, and kept rhs null. From figure 2(b), we can see that Jacobi method has a hard time converging when frequencies are low. Therefore, solving the system iteratively may not converge or it may take longer. Both smoothers remained intact when converting the problem from 1D to 2D. In a 2D setting, the number of unknowns goes from $(N-1)$ to $(N-1)^2$. This adds another complexity because these iterative algorithms are bounded by $O(N^4)$. As for the ω, JOR with $\omega = 0.8$, and SOR with $\omega = 1.5$ worked best. So, we chose those values for further benchmarks.

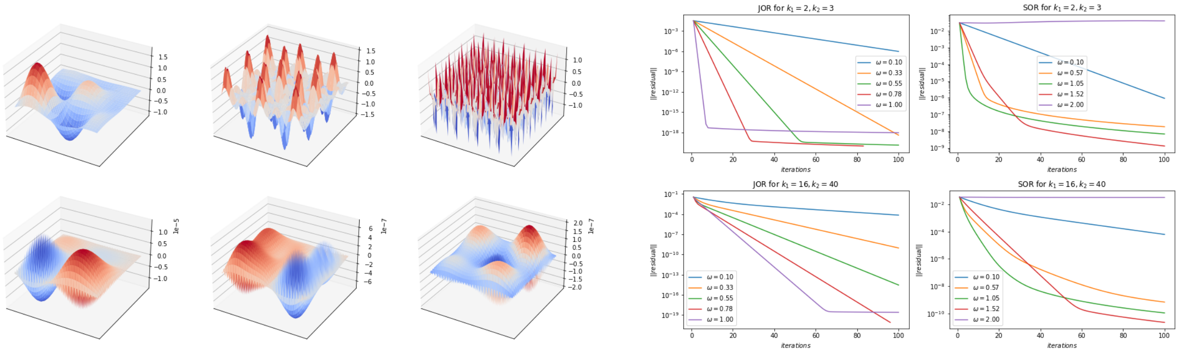

Figure 2: Relaxation over $N = 32$, $u(x,y) = \frac{1}{2}(sin(k_1 x\pi) + sin(k_2 x\pi)).(sin(k_1 y\pi) + sin(k_2 y\pi))$

Multigrid cycles

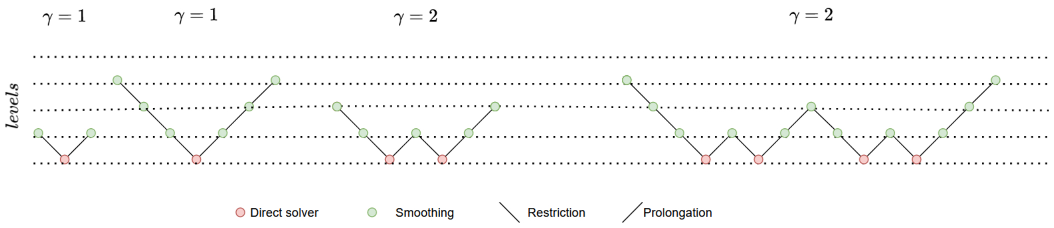

Figure 3: Two grid, V-cycle, and W-cycle representations

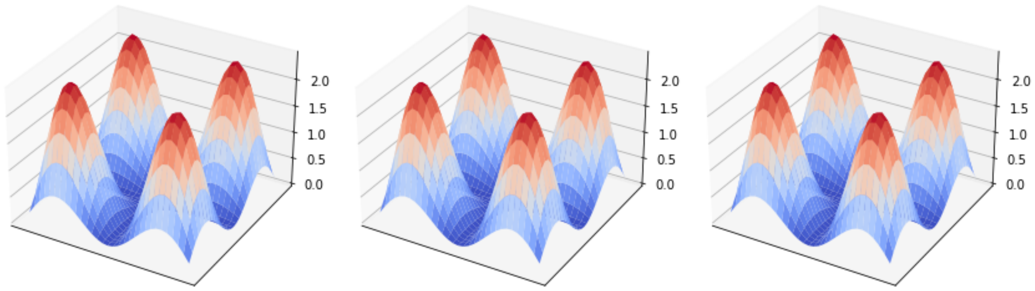

Figure 4: Different weighted schemes showed no major visual differences

We first implemented a two grid cycle algorithm as a foundation for multigrid for different levels l, and γ values. The general procedure was creating a 2D laplacian matrix A_h, restriction matrix R, and prolongation matrix P at each grid level. We recursively call the same method for solving $Ae = r$, and then add the interpolated corrections to our approximate solution. As for the exact solution, we tried both direct, and iterative solvers. There was no big difference. So, we chose a direct solver for the coarse linear system.

Algorithm 1 $Wcycle(A_h, b_h, u_h, N, L)$

```
Require: number of grid levels L $\geq 3$ and number of segments $N \mod 2^L = 0$
Ensure: $u_{final}$ solution of $A_h u_h = b_h$
  $R \leftarrow$ restriction operator
  $P \leftarrow$ prolongation operator
  $A_H \leftarrow R A_h P$                                  ▷ fine to coarse operation
  $u_0 \leftarrow u_h$                                      ▷ initial guess
  $u_h \leftarrow smoothing(A_h, b_h, u_0)$                 ▷ pre-smoothing
  $d_h \leftarrow b_h - A_h u_h$                            ▷ compute the defect
  $d_H \leftarrow R d_h$                                    ▷ restrict the defects
  if $L = 3$ then
    $e_H \leftarrow (0, 0, \ldots,)^T$
    $u_H \leftarrow twogrid(A_H, d_H, e_H, N/2)$            ▷ direct solver of two-grid
    $e_H \leftarrow twogrid(A_H, d_H, u_H, N/2)$
  else
    $u_H \leftarrow Wcycle(A_H, d_H, 0, N/2, L-1)$
    $e_H \leftarrow Wcycle(A_H, d_H, u_H, N/2, L-1)$
  end if
  $e_h \leftarrow P e_H$                                    ▷ prolongation of the error
  $u_h \leftarrow u_h + e_h$                                ▷ solution correction
  $u_{final} \leftarrow smoothing(A_h, b_h, u_h)$           ▷ post-smoothing
```

3 Results

We use the following parameters for benchmarking,

- pre-smoothing, and post-smoothing operators perform 5 iterations of SOR($\omega = 1.5$).
- For weighted schemes, we found no major differences between the final results (Figure 4). In some case, smoothing was less effective for half weighted scheme where some high frequency components were still visible in the solution. Therefore, We finalised on full weighted scheme as it yielded lower residual norms in general.
- Tested for σ in $\left[0\,, 10^6\right]$
- Kept right hand side, $f(x, y) = (sin(x\pi) + sin(3x\pi)).(sin(y\pi) + sin(3y\pi))$
- Generated all results for $N = 64$

We tried to compare the V-cycle and W-cycle with varying sigma and number of grid levels.
First, we computed the residual of the multigrid methods with varying sigma. We used a logarithmic scale from 0 to 10^6. As Figure 5 shows, both V-cycle and W-cycle have shown the same evolution of the residual with varying sigma. As we can see the best value is $\sigma = 10^5$. We tested this for different number of grid points, and found that the evolution, $u \to b$ holds when $\sigma \to \infty$.

Varying the number of grid levels did not affect neither execution time nor the residuals that nearly remains the same for different numbers of multigrid levels (from 3 to 5). Figure 6 shows the residual

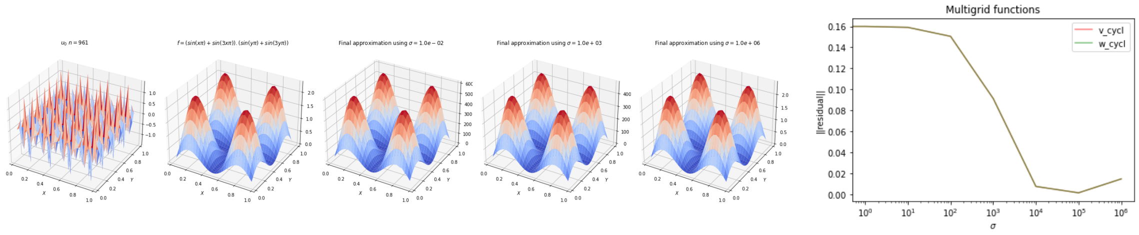

Figure 5: Evolution of the result over varying σ comparing with $f(x,y)$

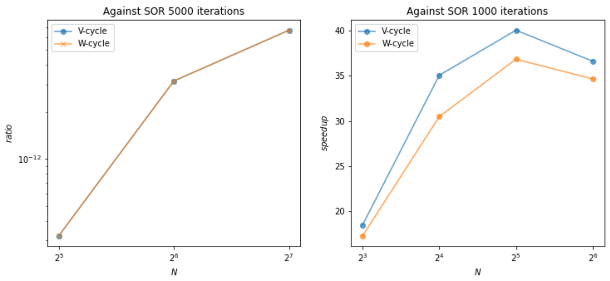

Figure 6: Residual ratio and Speedups obtained over solving the system using SOR for several iterations

ratios and speedups obtained over SOR($\omega = 1.5$) Vs V-cycle, and W-cycle. It proves that we can obtain reasonable residual ratios ($\approx 10^{-12}$) very quickly.

4 Anisotropic problem

Given

Solving the following 2D Poisson's equation,

$$-\frac{\partial^2 u(x,y)}{\partial x^2} - \epsilon \frac{\partial^2 u(x,y)}{\partial y^2} = f(x,y)$$

$$u(x,y) = 0 \; in \; \delta\Omega$$

The 2D Laplace operator for the above equation is,

$$B = \begin{bmatrix} 2(1+\epsilon) & -\epsilon & \\ -\epsilon & 2(1+\epsilon) & -\epsilon \\ & -\epsilon & 2(1+\epsilon) \end{bmatrix}$$

Results

Smoothing

To test the smoothing, we set the rhs to null and see how the convergence is for smoothing. Figure 7, shows that higher value for ϵ takes longer to converge. $\epsilon = 0$ did give the correct results for both the

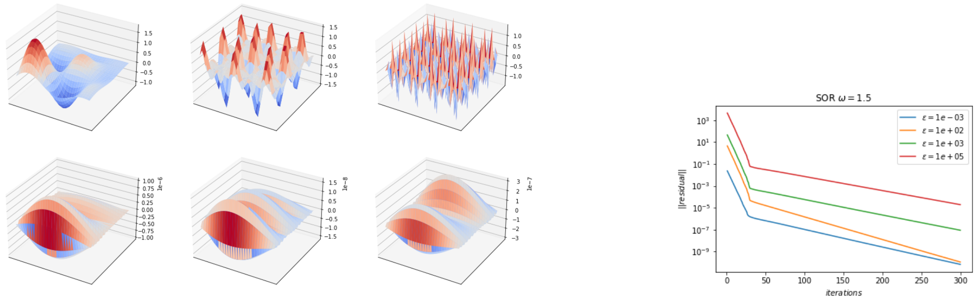

Figure 7: Solution for different frequencies with $f(x, y) = 0$, and $\epsilon = 10$

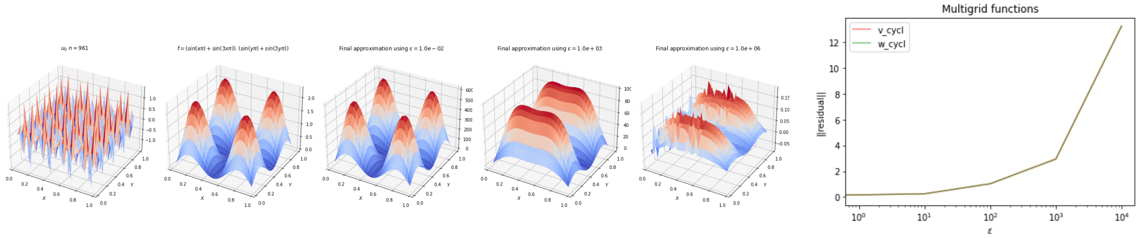

Figure 8: Evolution of the result over varying ϵ comparing with $f(x, y)$

V-cycle, and W-cycle.

Coarsening strategies

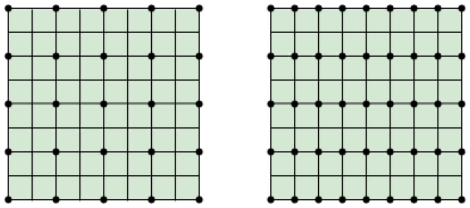

Figure 9: Standard and Semi-y coarsening

From the Figure 8, increasing ϵ affects the results drastically. It is clearly not able to converge to a good result. Unlike prior (isotropic) PDE, which is able to maintain the oscillation in both x and y axes, higher values for ϵ, almost loose that trend in one of the axes. In the figure, we can see frequency of x axis becomes low. We also confirmed that none of the weighted schemes were able to improve the solution.

It means only the coarsening strategy must be adjusted for the optimal solution. Currently, we were applying standard coarsening where the number of grid points were reduced in both directions of the grid. For this scenario, probably, we should have given less importance to y axis. We apply semi-y coarsening where we keep the grid points in x axis, and coarsen the grid points in y axis.

5 Conclusion

- Through this project we have implemented geometric multigrid methods and seen the capability of convergence to a good estimation of the RHS compared to iterative solvers.

- The anisotropic problem helped us to understand the impact of the coarsening strategy on solving the problem and thus for multigrid methods coarsening should be adapted to the problem.
- In terms of improvement, one important fact is that we are dealing with sparse matrices. So, it is optimal to use sparse representations such as CSR to reduce the computational cost associated with large sparse matrices. We decided to keep the dense matrices to see a speedups against iterative methods.