Multigrid For Solving Elliptic Differential Equations

Brian KYANJO   Prof. Grady WRIGHT

Department of Mathematics
Boise State University

May 4, 2021

Outline

Introduction

Many problems that arise from physical applications give us a natural feel to the multigrid methods. These methods have been applied directly to non-linear problems, and many re- searchers have used them to perform different studies on a variety of problems (Brandt, 1977).

In the study we concentrated on the poison equation (1).

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y) \tag{1}$$

Introduction

Many problems that arise from physical applications give us a natural feel to the multigrid methods. These methods have been applied directly to non-linear problems, and many re- searchers have used them to perform different studies on a variety of problems (Brandt, 1977).
In the study we concentrated on the poison equation (1).

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y) \tag{1}$$

Introduction

- To examine why the method works (Trottenberg et al., 2000).
- To use the method to solve elliptic partial differential equations (PDEs)
- To apply Fourier Analysis to the two-grid operator.
- To experiment with different smothers e.g. red-black and Gauss-Seidel to see how errors are smoothed.
- To compare the method with Discrete Sine Transform (DST) and the Sparse Gaussian Elimination (SGE) solver in solving Elliptic Differential Equations , i.e, look at the computational time take to solve the problem in question.
- To validate the method results with a DST solver.

Formulation

Multigrid (MG) is one of the fastest method used in solving elliptic PDEs and it has been applied in combination with most of the discretization techniques.

The main idea behind multigrid is to switch between a coarser and finer grid to estimate the remaining smoothed error.

This is a good approach because:

- Its cheap iterating on a coarser grid than further continuing on the original grid.
- The convergence rate of most components of the error is greatly improved on shifting them on the coarser grid.

Formulation

Multigrid (MG) is one of the fastest method used in solving elliptic PDEs and it has been applied in combination with most of the discretization techniques.
The main idea behind multigrid is to switch between a coarser and finer grid to estimate the remaining smoothed error.
This is a good approach because:

- Its cheap iterating on a coarser grid than further continuing on the original grid.
- The convergence rate of most components of the error is greatly improved on shifting them on the coarser grid.

Formulation

Multigrid (MG) is one of the fastest method used in solving elliptic PDEs and it has been applied in combination with most of the discretization techniques.
The main idea behind multigrid is to switch between a coarser and finer grid to estimate the remaining smoothed error.
This is a good approach because:

- Its cheap iterating on a coarser grid than further continuing on the original grid.
- The convergence rate of most components of the error is greatly improved on shifting them on the coarser grid.

MG algorithm

The basic recursive multigrid algorithm on level l is illustrated below, with S, an iteration operator, K is a tridiagonal matrix;

```
**Multigrid method**   MGM($l, u, f$)
  if $l = 0$ then
    $u := K_0^{-1} f$  // exact solve on coarsest level
  else
    $u := S_l^{\nu_1}(u, f)$;  //presmoothing step
    $d := I_l^{l-1}(K_l u - f)$;  //restriction of the defect
    $v := 0$;  //starting value
    for $j := 1, \dots, \gamma$, do
      MGM($l-1, v, d$)  // solve $K_{l-1} v_{l-1} = d_{l-1}$ by applying $\gamma$ step of MGM($l-1, v, d$)
    end for
    $u := u - I_{l-1}^{l} v$  //update on level $l$
    $u := S_l^{\nu_2}(u, f)$  //postsmoothing step
  end if
```

Figure: MG algorithm

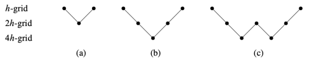
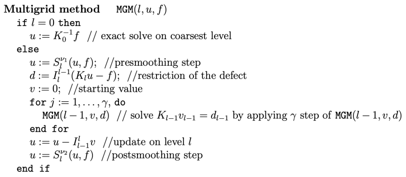


Figure: (a) and (b) respectively are One V-cycle with two and three levels. (c) One W-cycle with three levels (LeVeque, 2007).

MG algorithm

The basic recursive multigrid algorithm on level l is illustrated below, with S, an iteration operator, K is a tridiagonal matrix;



```
Multigrid method   MGM($l, u, f$)
  if $l = 0$ then
    $u := K_0^{-1} f$  // exact solve on coarsest level
  else
    $u := S_l^{\nu_1}(u, f)$;  //presmoothing step
    $d := I_l^{l-1}(K_l u - f)$;  //restriction of the defect
    $v := 0$;  //starting value
    for $j := 1, \ldots, \gamma$, do
      MGM($l-1, v, d$)  // solve $K_{l-1} v_{l-1} = d_{l-1}$ by applying $\gamma$ step of MGM($l-1, v, d$)
    end for
    $u := u - I_{l-1}^{l} v$  //update on level $l$
    $u := S_l^{\nu_2}(u, f)$  //postsmoothing step
  end if
```

Figure: MG algorithm

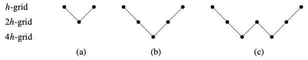


Figure: (a) and (b) respectively are One V-cycle with two and three levels. (c) One W-cycle with three levels (LeVeque, 2007).

Accuracy and Computational cost

MG solves elliptic PDEs to a given accuracy in a number of operations proportional to the number of unknowns i.e., it scales linearly with the number of discrete nodes used.

The computational cost associated with using multigrid method as a solver is O(N), however there is an exception to O(N), since the W-cycle multigrid uses O(Nlog(N) time to solve a 1D problem.

Accuracy and Computational cost

MG solves elliptic PDEs to a given accuracy in a number of operations proportional to the number of unknowns i.e., it scales linearly with the number of discrete nodes used.
The computational cost associated with using multigrid method as a solver is O(N), however there is an exception to O(N), since the W-cycle multigrid uses O(Nlog(N) time to solve a 1D problem.

Smothers Used.

In MG algorithm smothers act as the central components that determine the performance of the algorithm by reducing high frequency errors.

Damped Jacobi Damped Jacobi described in equation (2) is Jacobi method is with a relaxation parameter ω. It converges faster for high frequency.

$$u^{(m+1)} = (I - \omega D^{-1}A)u^{(m)} + \omega D^{-1}b, \qquad \omega \in (0,1] \tag{2}$$

Red-Black Gauss Seidel: This is basically the red-black ordering of grid points.

Smothers Used.

In MG algorithm smothers act as the central components that determine the performance of the algorithm by reducing high frequency errors.
Damped Jacobi Damped Jacobi described in equation (2) is Jacobi method is with a relaxation parameter ω. It converges faster for high frequency.

$$u^{(m+1)} = (I - \omega D^{-1}A)u^{(m)} + \omega D^{-1}b, \qquad \omega \in (0, 1] \tag{2}$$

Red-Black Gauss Seidel: This is basically the red-black ordering of grid points.

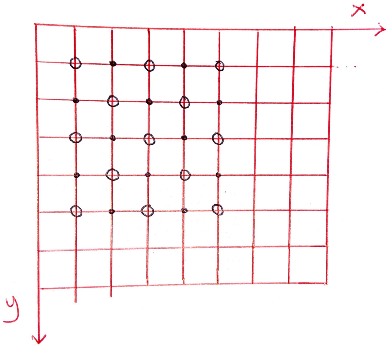
Smothers Used.

In MG algorithm smothers act as the central components that determine the performance of the algorithm by reducing high frequency errors.
Damped Jacobi Damped Jacobi described in equation (2) is Jacobi method is with a relaxation parameter ω. It converges faster for high frequency.

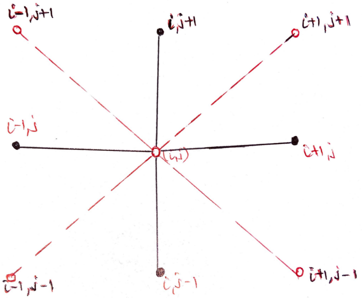
$$u^{(m+1)} = (I - \omega D^{-1}A)u^{(m)} + \omega D^{-1}b, \qquad \omega \in (0, 1] \tag{2}$$

Red-Black Gauss Seidel: This is basically the red-black ordering of grid points.

Cont.



(a)



(b)

Figure: (a) and (b) respectively show $2D$ grid and stencil for red-black with red (o) and black ($\cdot$) points

Fourier Analysis (FA) to the two-grid operator.

FA is used to chose individual multigrid components for different situations, and it was also applied in the calculation of convergence rates of MG methods while solving elliptic PDEs by using the exact eigenfunction of the discrete operator that is compatible with the boundary conditions (Kuo and Levy, 1989).

The discrete fine-grid solution u_h can be given by the linear combination of of Fourier components that generate the whole space of bounded infinite grid functions.

Fourier Analysis (FA) to the two-grid operator.

FA is used to chose individual multigrid components for different situations, and it was also applied in the calculation of convergence rates of MG methods while solving elliptic PDEs by using the exact eigenfunction of the discrete operator that is compatible with the boundary conditions (Kuo and Levy, 1989).
The discrete fine-grid solution u_h can be given by the linear combination of of Fourier components that generate the whole space of bounded infinite grid functions.

FA cont.

The error before and after the i^{th} 2-grid-cycle is given by equations (3) and (4) respectively.

$$e_h^{(i)} = u_h^{(i-1)} - u_h \tag{3}$$

$$e_h^{(i)} = u_h^{(i)} - u_h \tag{4}$$

So their corresponding two-grid cycle error transformation is given by equation (5);

$$e_h^{(i)} = M_h^{2h} e_h^{(i-1)} \tag{5}$$

with M_h^{2h} is the error-transformation operator given by

$$M_h^{2h} = S_h^{\nu_2} K_h^{2h} S_h^{\nu_1} \tag{6}$$

FA cont.

The error before and after the i^{th} 2-grid-cycle is given by equations (3) and (4) respectively.

$$e_h^{(i)} = u_h^{(i-1)} - u_h \tag{3}$$

$$e_h^{(i)} = u_h^{(i)} - u_h \tag{4}$$

So their corresponding two-grid cycle error transformation is given by equation (5);

$$e_h^{(i)} = M_h^{2h} e_h^{(i-1)} \tag{5}$$

with M_h^{2h} is the error-transformation operator given by

$$M_h^{2h} = S_h^{\nu_2} K_h^{2h} S_h^{\nu_1} \tag{6}$$

Other Solvers Used.

These solvers have been used in comparing and validating our results.
Discrete Sine Transform (DST): It uses a signal at a discrete domain, and at the two ends of the array different variants corresponding to almost different odd/even boundary conditions(Gupta and Rao, 1990).
Sparse Gaussian Elimination (SGE): This solver uses sparse matrix (only a small fraction of the matrix elements are non zeros) capabilities to form D2x and D2y sparse matrices (Brayton et al., 1970).

Other Solvers Used.

These solvers have been used in comparing and validating our results.
Discrete Sine Transform (DST): It uses a signal at a discrete domain, and at the two ends of the array different variants corresponding to almost different odd/even boundary conditions(Gupta and Rao, 1990).
Sparse Gaussian Elimination (SGE): This solver uses sparse matrix (only a small fraction of the matrix elements are non zeros) capabilities to form D2x and D2y sparse matrices (Brayton et al., 1970).

Case Study

The $2D$ elliptic problem was solved over the domain [a,b]x[a,b] with $a = 0$ and $b = 1$, grid spacing $dx = dy = h = \frac{b-a}{m+1}$, with $m = 2^{k-1}$

$$\nabla^2 u = f(x,y) \tag{7}$$

where

$$f(x,y) = 10\pi^2(1 + \cos(4\pi(x + 2y))) - 2 * \sin(2\pi(x + 2y))e^{sin(2\pi(x+2y))}$$

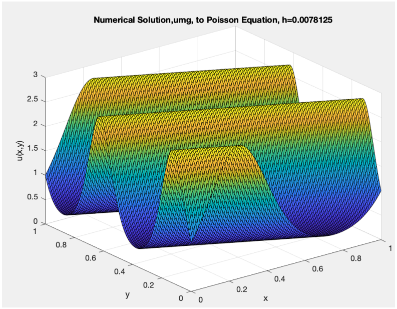
with Boundary conditions.

$$u(x,y) = g(x,y)$$

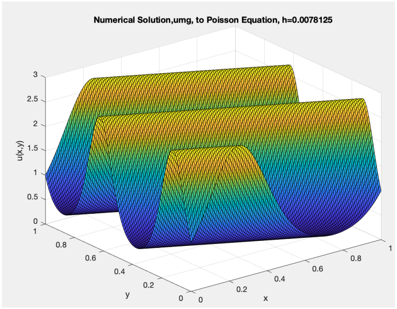
where

$$g(x,y) = e^{sin(2\pi(x+2y))}$$

Results



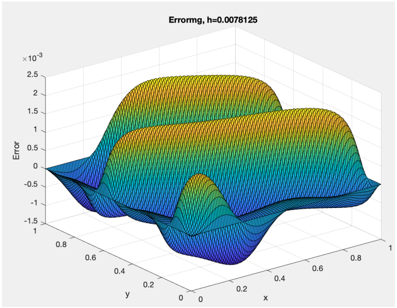
(a)



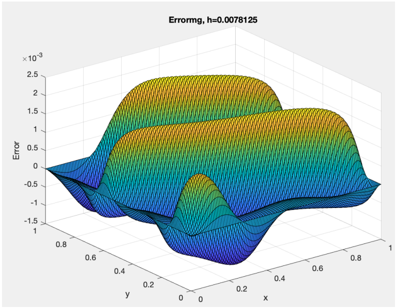
(b)

Figure: (a) and (b) respectively are solutions obtained using damped Jacobi and Red-black smothers.

Cont.



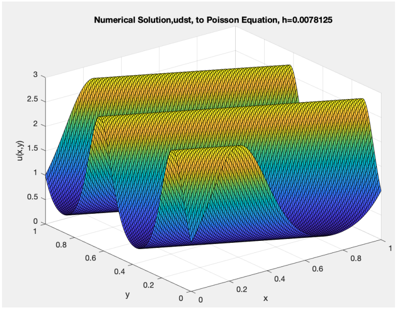
(a)



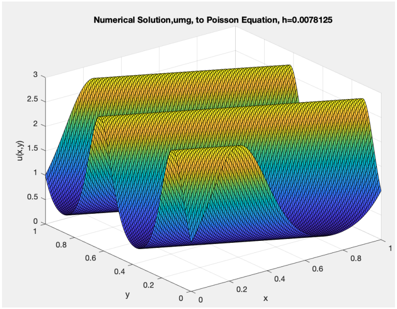
(b)

Figure: (a) and (b) respectively are errors obtained using damped jacobi and Red-black smothers.

Cont.



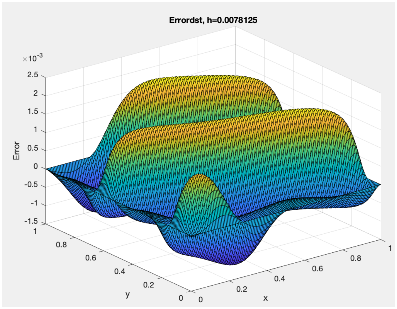
(a)



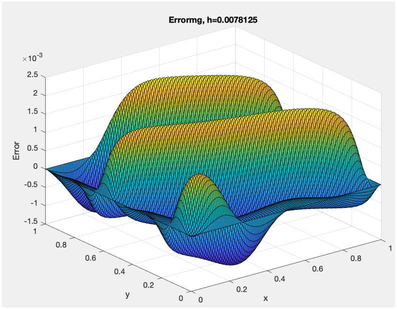
(b)

Figure: (a) and (b) respectively are solutions obtained using DST and Multigrid solver with damped Jacobi smother.

Cont.



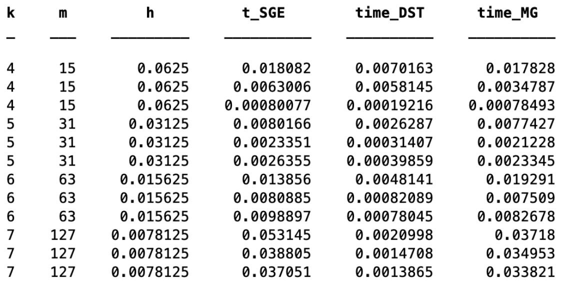
(a)



(b)

Figure: (a) and (b) respectively are errors obtained using DST and Multigrid solver with damped Jacobi smother.

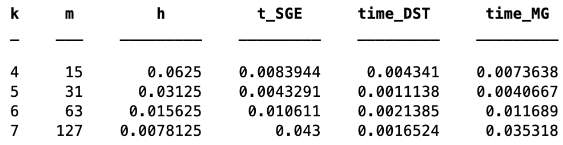
Cont.



```
k     m         h           t_SGE        time_DST       time_MG
_    ___    _________    __________    __________    __________

4     15       0.0625      0.018082     0.0070163      0.017828
4     15       0.0625     0.0063006     0.0058145     0.0034787
4     15       0.0625    0.00080077    0.00019216    0.00078493
5     31      0.03125     0.0080166     0.0026287     0.0077427
5     31      0.03125     0.0023351    0.00031407     0.0021228
5     31      0.03125     0.0026355    0.00039859     0.0023345
6     63     0.015625      0.013856     0.0048141      0.019291
6     63     0.015625     0.0080885    0.00082089      0.007509
6     63     0.015625     0.0098897    0.00078045     0.0082678
7    127    0.0078125      0.053145     0.0020998       0.03718
7    127    0.0078125      0.038805     0.0014708      0.034953
7    127    0.0078125      0.037051     0.0013865      0.033821
```

Figure: Wall time at each run

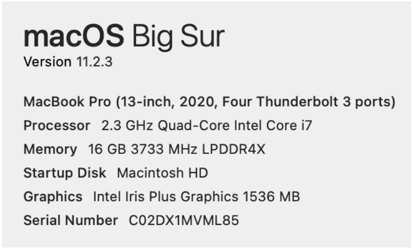


```
k     m         h          t_SGE       time_DST      time_MG
_    ___    _________    _________    _________    _________

4     15       0.0625    0.0083944     0.004341    0.0073638
5     31      0.03125    0.0043291    0.0011138    0.0040667
6     63     0.015625     0.010611    0.0021385     0.011689
7    127    0.0078125        0.043    0.0016524     0.035318
```

Figure: Mean wall time

Cont.


macOS Big Sur
Version 11.2.3

MacBook Pro (13-inch, 2020, Four Thunderbolt 3 ports)

Processor	2.3 GHz Quad-Core Intel Core i7
Memory	16 GB 3733 MHz LPDDR4X
Startup Disk	Macintosh HD
Graphics	Intel Iris Plus Graphics 1536 MB
Serial Number	C02DX1MVML85

Figure: Computer specifications

Conclusion

We used the Discrete Sine Transform (DST) to compare and validate our solution. And according to figures 6a, 6b, 7a, and 7b, we were able to validate our solution since the results depict the same solution and magnitude of the error (10^{-3}).

According to table 9, overall, DST wins in terms of cost, however the mean wall time for MG, is similar to the two methods we are validating with, hence the results obtained from using MG to solve the same problem are realistic.

In a case where the matrix of the original equation is symmetric positive definite (SPD), then a multgrid method with a forced reduced tolerance may be used as a preconditioner, however, the SPD constraint makes construction of the preconditioner complicated.

Conclusion

We used the Discrete Sine Transform (DST) to compare and validate our solution. And according to figures 6a, 6b, 7a, and 7b, we were able to validate our solution since the results depict the same solution and magnitude of the error (10^{-3}).
According to table 9, overall, DST wins in terms of cost, however the mean wall time for MG, is similar to the two methods we are validating with, hence the results obtained from using MG to solve the same problem are realistic.
In a case where the matrix of the original equation is symmetric positive definite (SPD), then a multgrid method with a forced reduced tolerance may be used as a preconditioner, however, the SPD constraint makes construction of the preconditioner complicated.

Conclusion

We used the Discrete Sine Transform (DST) to compare and validate our solution. And according to figures 6a, 6b, 7a, and 7b, we were able to validate our solution since the results depict the same solution and magnitude of the error (10^{-3}).
According to table 9, overall, DST wins in terms of cost, however the mean wall time for MG, is similar to the two methods we are validating with, hence the results obtained from using MG to solve the same problem are realistic.
In a case where the matrix of the original equation is symmetric positive definite (SPD), then a multgrid method with a forced reduced tolerance may be used as a preconditioner, however, the SPD constraint makes construction of the preconditioner complicated.

Thank you!

briankyanjo@u.boisestate.edu

References

Brandt, A. (1977). Multi-level adaptive solutions to boundary-value problems. *Mathematics of computation*, 31(138):333–390.

Brayton, R. K., Gustavson, F. G., and Willoughby, R. A. (1970). Some results on sparse matrices. *Mathematics of Computation*, 24(112):937–954.

Gupta, A. and Rao, K. R. (1990). A fast recursive algorithm for the discrete sine transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(3):553–557.

Kuo, C.-C. J. and Levy, B. C. (1989). Two-color fourier analysis of the multigrid method with red-black gauss-seidel smoothing. *Applied Mathematics and Computation*, 29(1):69–87.

LeVeque, R. J. (2007). *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM.

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