Numerical Methods II, Courant Institute, Spring 2024
http://www.math.nyu.edu/faculty/goodman/teaching/NumericalMethodsII2024.html
Jonathan Goodman, February, 2024

Assignment 4, Multigrid

The *multigrid* method is a way to solve large systems of equations that arise as discretizations of elliptic PDE. They are based on combining two (or more) kinds of "moves" (a term from MCMC that seems applicable here). *Relaxation*, also called *smoothing* when used in multibrid, is a *local* "move" like Gauss Seidel or Jacobi iteration. *Coarse grid correction* is a "move" that solves a version of the problem on a coarse grid. The local smoothing move reduces high frequency components of the error. The coarse grid correction reduces low frequency components. Alternating between smoothing and coarse grid correction gives a method that reduces all components. "Multigrid convergence" means that each full multigrid iteration (iterations are often called *cycles* to indicate that it cycles through the two moves) reduce the error norm by a fixed factor (say, .5) no matter how fine the grid is. If there are N grid points in all, then multigrid convergence would solve the equation system in total work proportional to N. By contrast, Gauss Seidel alone on an $n \times n$ mesh takes $O(n^2)$ iterations to reduce the error by a factor of 2, and each iteration takes $O(n^2)$ work. The number of grid points is $N = n^2$ so the work to get a factor of 2 reduction in the error is $O(N)$ instead of $O(1)$ for multigrid. Multigrid makes possible very high resolution massive grid computations that would be impractical using simple iteration methods, even when accelerated using conjugate gradients.

We will solve the discrete Laplace equation on a square $n \times n$ mesh with Dirichlet boundary conditions and the five point discrete Laplace operator. The discrete equations are

$$U_{jk} = \tfrac{1}{4} \left[U_{j,k+1} + U_{j,k-1} + U_{j+1,k} + U_{j-1,k} \right] + f_{jk} \; .$$

These equations should hold for j and k in the range $1, 2, \cdots, n$. Dirichlet boundary conditions are implemented by setting $U_{jk} = 0$ for j or k equal to 0 or $n+1$ (ghost cells). If V is any grid function, the *residual*, r, is defined by

$$V_{jk} = \tfrac{1}{4} \left[V_{j,k+1} + V_{j,k-1} + V_{j+1,k} + V_{j-1,k} \right] + f_{jk} + r_{jk} \; . \tag{1}$$

The *correction*, W, is the grid function that satisfies

$$W_{jk} = \tfrac{1}{4} \left[W_{j,k+1} + W_{j,k-1} + W_{j+1,k} + W_{j-1,k} \right] - r_{jk} \; .$$

The exact solution U is found by adding the correction to V (check this):

$$U = V + W \; .$$

The *coarse grid correction* is an approximation to W that is found using a coarser mesh.

Multigrid algorithms require communication between fine and coarse grids. Here is one of the ways this can happen. Suppose the fine grid n is odd, so $n = 2m + 1$. The coarse grid is the subset of fine grid points with even indices. This makes the coarse grid $m \times m$, which has about four times fewer grid points. Multigrid convention uses h to represent the fine grid (think of $h = \Delta x$) and $2h = H$ for the coarse grid. We write U^h for an $n \times n$ grid of values (padded with ghost cells, to make an $(n+2) \times (n+2)$ array in the computer). We write U^{2h} for an $m \times m$ grid vector. The *prolongation* operator (also called *interpolation*) is I^h_{2h} ("Interpolation" from the $2h$ to the h grid). Prolongation is $U^h = I^h_{2h} U^{2h}$. We do this using piecewise linear interpolation. Let j and k be in the range from 1 to m on the coarse grid. The (j, k) grid point on the coarse grid is in the same place as the $(2j, 2k)$ grid point on the fine grid. Therefore, we take

$$U^h_{2j,2k} = U^{2h}_{jk} \ .$$

We fill in in the pure x and y directions by linear interpolation, so

$$\begin{aligned} U^h_{2j-1,2k} &= \frac{1}{2} \left(U^{2h}_{jk} + U^{2h}_{j-1,k} \right) \\ U^h_{2j,2k-1} &= \frac{1}{2} \left(U^{2h}_{jk} + U^{2h}_{j,k-1} \right) \ . \end{aligned}$$

The diagonal fill-in uses values at the 4 nearest coarse grid points

$$U^h_{2j-1,2k-1} = \frac{1}{4} \left(U^{2h}_{jk} + U^{2h}_{j-1,k} + U^{2h}_{j,k-1} + U^{2h}_{j-1,k-1} \right) \ .$$

These equations define the prolongation operator $U^h = I^h_{2h} U^{2h}$. As a matrix, I^h_{2h} is $n^2 \times m^2$ (n^2 rows and m^2 columns). The *restriction* operator produces a course grid vector from a fine grid vector. It goes from h to $2h$ and is written I^{2h}_h. We will use a restriction operator called "full weighting" (because it's better than an earlier idea). This is

$$I^{2h}_h = \tfrac{1}{4} \left(I^h_{2h} \right)^T \ .$$

This is a matrix with m^2 rows and n^2 columns. The prefactor is so that all ones on the fine grid lead to all ones on the coarse grid. Figures **??** and **??** illustrate the prolongation and restriction operators. The restriction operator takes the coarse grid value to be a weighted sum of the 9 nearest fine grid values. The prolongation operator distributes each coarse grid value to the same 9 fine grid points. A fine grid value is the sum of the contributions from its coarse neighbors. There can be 1, 2 or 4 coarse grid values contributing to a given find grid value, depending on the location of the fine grid point. These operations

The multigrid algorithm is *recursive* in that it uses a procedure that calls itself. This makes code structure more important. There should be separate procedures (subroutines, functions) for a Gauss Seidel iteration (one "sweep"), and for coarse-to-fine and fine-to-coarse grid transfers. These need to have clear interfaces and understandings about their arguments (sizes, presence of ghost

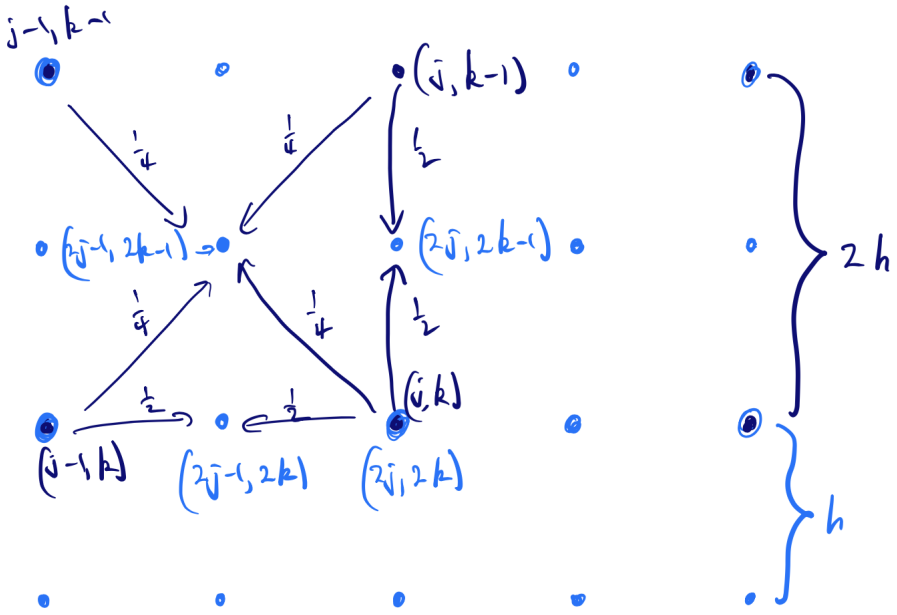

Figure 1: Prolongation operator with weights. Fine grid points are blue and coarse points are black. Every black point is also blue, which is why black points have blue circles around them.

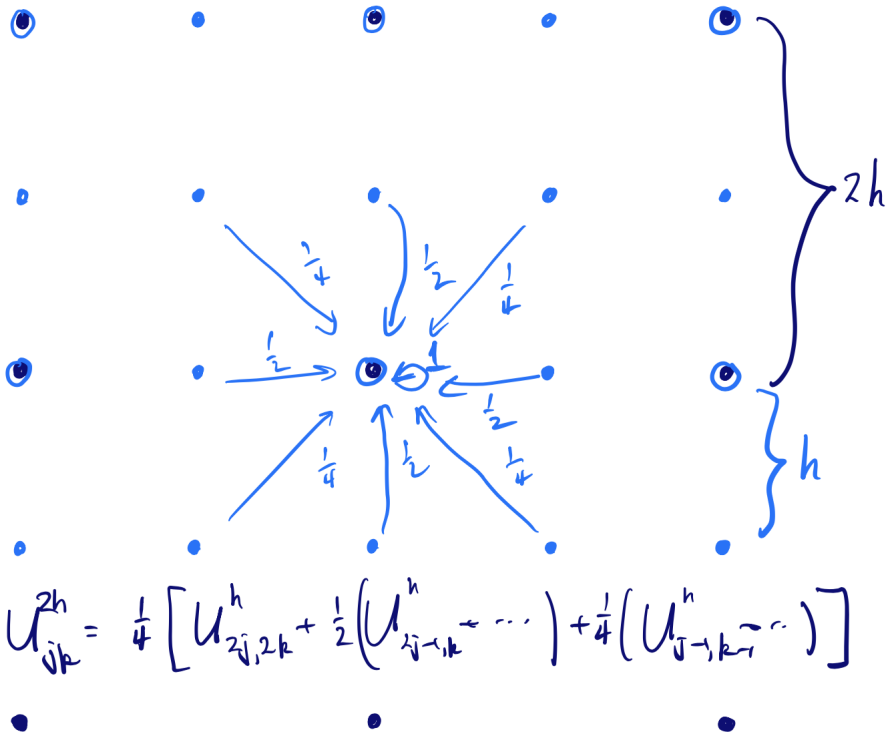

Figure 2: Restriction operator with weights. All the weights drawn add up to $1 + 4 \cdot \frac{1}{2} + 4 \cdot \frac{1}{4} = 4$. The overall prefactor $\frac{1}{4}$ makes it so that restricting a constant function on the fine grid gives the same constant on the coarse grid. Figure **??** has the fine and coarse grid points labelled.

cells, etc.)

Exercise 1.

1. Write a procedure that takes two grid functions U and f and does one Gauss Seidel sweep.

2. Write a procedure that computes and returns the residual defined by (**??**).

3. Write procedures to implement restriction and prolongation operators. Both find and coarse grids should have borders of ghost cells with zeros that may be accessed but should not be modified. To check that these are coded correctly, check that restrictions and prolongations of linear functions produce the same linear functions. Be careful that this (linear $\longrightarrow$ linear) property happens only in the interior but is spoiled at the boundary by ghost cell values equal to zero.

4. Start with $U^h = 0$ and $f = 1$ (the constant function) and then do t Gauss Seidel sweeps (call your Gauss Seidel procedure t times). Take that U^h and see how well it can be represented by an interpolation from the coarse grid by first restricting to the coarse grid and then interpolating back to the fine grid:
$$\widetilde{U}^h = I_{2h}^h I_h^{2h} U^h \ .$$
Also calculate r, the residual of U^h. Make one figure that contains plots of
$$e(t) = \left\| \widetilde{U}^h - U^h \right\| \ , \ \ \|r\|$$
as functions of t. Note that these functions may have different overall scales, so you may need different scalings of the y axis to see both curves clearly. The result should be that r decreases more slowly with t than $e(t)$. This shows that the error is smooth enough to be accurately represented on a coarse grid before it is small. This effect is more dramatic when n is larger. Take n as large as is practical (run time a few seconds) for this part. You will not need a large t.

The *two grid* method involves only the coarse $m \times m$ grid with spacing $2h$ and the $n \times n$ grid with spacing h, and $n = 2m + 1$. One two-grid iteration is

- Start with a fine grid function U^h.
- Do p Gauss Seidel sweeps on the fine grid, giving a fine grid function V^h
- Compute the V^h residual on the fine grid, r^h.
- Restrict the fine grid residual to the coarse grid $r^{2h} = I_h^{2h} r^h$.
- Solve the discrete Laplace system on the coarse grid: $L^{2h} W^{2h} = -r^{2h}$.
- Interpolate the coarse grid correction to the fine grid $W^h = I_{2h}^h W^{2h}$.

- Add the coarse grid correction $\widetilde{U}^h = V^h + W^h$.
- $U^h \to \widetilde{U}^h$ is one two-grid iteration.

Exercise 2. Implement the two grid iteration. You may solve the coarse grid equations using Gauss Seidel iterations on the coarse grid. Make a plot of $\|r\|$ as a function of two-grid iteration number. Choose a relatively small p (such as $p = 2$ or $p = 1$ or $p = 4$) and note how the residual decreases. How does the rate of decrease depend on the mesh size n? (Answer: it should be independent of n, but this experiment will be very slow for large n, likely slower than just doing Gauss Seidel on the fine mesh.

The *multigrid method* is essentially the two grid method with the coarse grid solve replaced by some number of multi-grid iterations on the coarse grid. One multi-grid iteration consists of p fine grid Gauss Seidel sweeps followed by a coarse grid correction. To do a coarse grid correction, you first evaluate the residual on the fine grid and restrict it to the coarse grid, as in the two-grid method. The coarse grid correction is q multi-grid iterations on the coarse grid. The method with $q = 1$ is the "V-cycle". With $q = 2$ it is the "W-cycle". With $q = 3$ it is a "super W-cycle" (very rare). The coarsest mesh could be $n = 3$ or $n = 2$. On that you either do a direct solve, which involves a $9\times$ matrix if $n = 3$, or you do a lot of Gauss Seidel iterations (maybe 10).

Exercise 3. Show that the total work for one multi-grid iteration is $O(n^2)$, where n^2 is the number of mesh points on the finest grid. This is true for $q = 1, 2, 3$. The total work is $O(n^2 \log(n))$ for $q = 4$. Don't worry about $q > 4$. The basic fact is that the coarse grid has 4 times fewer points than the fine grid, so 3 coarse grid sweeps take $\frac{3}{4}$ of the work of one coarse grid sweep. Continuing, 9 sweeps on the coarse coarse grid take $\left(\frac{3}{4}\right)^2$ less work. You can sum over all the grids and get a finite answer (that doesn't grow with n) if $q \leq 3$.

Multi-grid convergence means getting a fixed factor reduction in the error or residual for an amount of work proportional to the number of fine grid points where the reduction factor is independent of the fine grid size. In practice, the reduction is not just fixed, but small, such as $\frac{1}{2}$ or less.

Figure]reffig:mg illustrates different multigrid cycles depending on parameters of the method.

- p – the number of pre-sweeps. The number of Gauss Seidel sweeps before the coarse grid correction.
- r – the number of post-sweeps. The number of Gauss Seidel iterations after the coarse grid correction.
- q – the number of multigrid cycles on the coarse grid per fine grid grid multigrid cycle. $q = 1$ is the V cycle. $q = 2$ is the W cycle.

Exercise 4. Implement the multi-grid method and see whether the residual (which can be measured in the solution process) goes down by a fixed factor independent of the mesh size. Experiment with parameters p and q to see how the convergence factor depends on them. Note that $p = 2$ is roughly twice as much work per iteration as $p = 1$, so try to make recommendations for p and q for solving the discrete Laplace equation on a big fine mesh as quickly as possible.

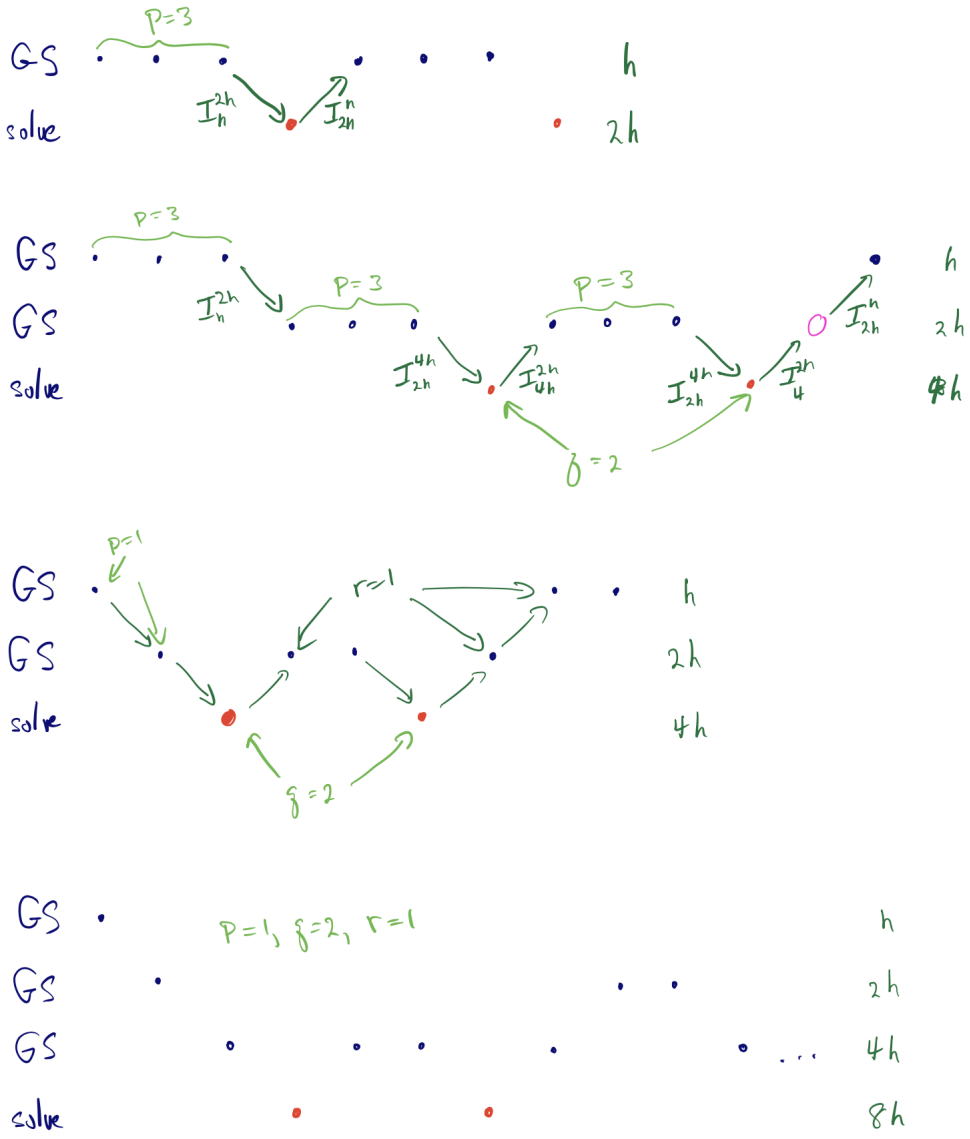

Figure 3: Illustrations of multigrid cycles. Black dots represent Gauss Seidel sweeps. Red dots represent direct solves. Dark green arrows represent grid transfer (restriction or prolongation) operations. The top is a two grid method with $p = 3$ Gauss Seidel followed by a coarse grid solve. Next is a three grid W-cycle ($q = 2$ with $p = 3$. After that is another W-cycle, but this time with one pre-sweep ($p = 1$) and one *post* sweep ($r = 1$). With the post-sweep, there is at least one Gauss Seidel sweep between the $4h \to 2h$ and $2h \to h$ prolongations. The bottom is maybe the most popular W-cycle (one pre and post sweep) illustrated with 4 grids.