Lesson 8

CSPP58001 Numerical Methods:
TA: Kyle Gerard Felker

March 4, 2013

1 Full Multigrid Algorithm (FMA)

Returning to our diffusion equation, we consider the steady-state solution ($\frac{\partial u}{\partial t} = 0$) which satisfies this relationship

$$\nabla^2 u(x,y) = -\rho(x,y) \tag{1}$$

where ρ is the known time-independent source term. We are going to explore this equation thoroughly today, so that you will be well equipped to handle the full transient solution for your final homework project.

1.1 Discretization

Without a time step, there is no coefficient C, so our discretization is a bit simpler.

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = \Delta x^2 \rho(i,j) \tag{2}$$

We could use our normal discretization and solver to time step until we reach a steady state solution, but if we don't really care about what happens along the way, we can get there faster by just solving this system. This is the Poisson equation, whereas the Laplace equation is $\nabla^2 u(x,y) = 0$.

We will now set up grids of different granularities to solve our problem, as we did in the two level multigrid case in the previous lecture. Figure **??** is a graphical representation of these increasingly coarser grids. C is a good language for storing these grids, thanks to the ability to handle "jagged" arrays of different lengths in each dimension.

Here are the steps (and their corresponding source code files in the Dropbox):

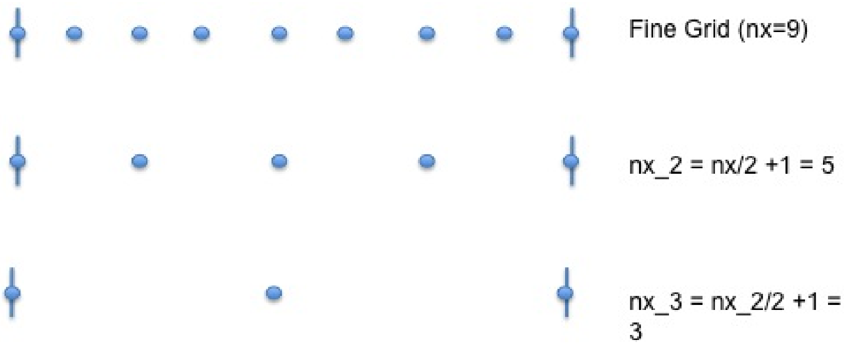

Figure 1: The multiple grids of multigrid for a 1D problem. The dots represent mesh points. Notice that nx= $2^n + 1$.

- Coarsen RHS (i.e. $\rho(i, j)$) from finest to coarsest grid (Restriction step). $\boxed{\text{rstr.c}}$
- Solve problem directly on coarse grid (Solve step). $\boxed{\text{smlslv.c}}$
- Interpolate solution one grid level finer (Prolongation step). $\boxed{\text{addint.c}}$
- Do Gauss-Seidel sweeps to reduce high-wavenumber error component (Sweep step). $\boxed{\text{relax.c}}$

Figure **??** illustrates two methods of organizing these steps. We will now illustrate each of these algorithms as they are implemented in our code.

$\boxed{\text{rstrct.c}}$

For our restriction operator, we use simple half-weighting. The boundary points must stay the same as you transition to coarser grids, and the coarser interior points are governed by

$$u_{coarse}(i) = \frac{1}{4}u(i-1) + \frac{1}{2}u(i) + \frac{1}{4}u(i+1) \tag{3}$$

$\boxed{\text{smlslv.c}}$

This part of the source code depends on your PDE. For example, on a 5x5 coarse grid, where the edges are unchanging boundary mesh points, we take the center (2,2) point's solution as

$$u(2,2) = \frac{-\Delta x^2}{4}\rho(2,2) \tag{4}$$

$\boxed{\text{addint.c}}$

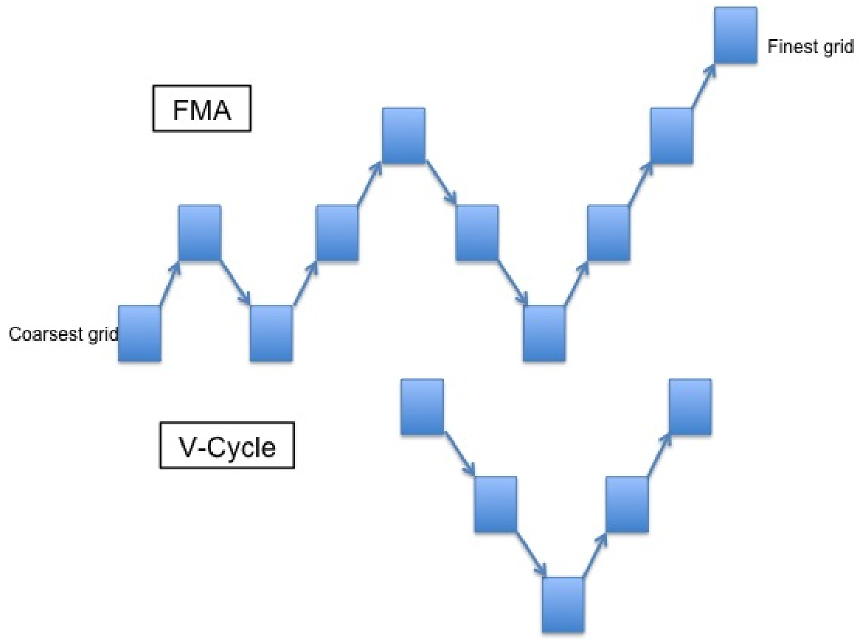

Figure 2: Two types of multigrid cycles for solving your system. The up arrows represent interpolation and the down arrows represent restriction.

This file is composed almost entirely of interp.c Interpolation is the inverse of restriction– it takes an array on the coarse grid and represents it on a grid one level finer. In 2D, we use bilinear interpolation. First, we do direct copies of the interior mesh points which exist on both the fine and coarse grids. Second, we go over the odd columns and fill out fine points vertically as $\frac{1}{2}$ each vertical neighbor. Third, we loop over even columns and fill the remaining fine points horizontally in the same way.

$\boxed{\text{relax.c}}$

We will loop over our temperature data in the standard Gauss-Seidel method:

$$u_{i,j} = \frac{1}{4}[-\Delta x^2 \rho(i,j) + u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}]$$

However, we make one improvement. We implement Red-Black ordering, where we update half of the mesh points in one sweep and do the rest in a second sweep in a checkerboard pattern. That way, the second sweep "red points" only depend on previously updated "black points".

1.2 Residual operation on FMA

Consider the residual defined as $r = -Ae = Ax - b$ where x is some guess at solution and $e = \tilde{x} - x$ is the error (exact solution minus guess). We can formulate the FMA steps as operations on this error

1. Guess x and compute $-r = b - Ax$
2. Restrict r to coarser grid
3. Solve $A^{2H}e^{2h} = r^{2h}$
4. Interpolate $e^{2h} \to e^h$
5. Add error to previous guess