Multigrid in a Nutshell

David J. Appelhans

I. DISCRETIZATION: FROM A DIFFERENTIAL EQUATION TO A LINEAR ALGEBRA PROBLEM

Through numerical approximations, solving a differential equation on a computer amounts to having to solve a linear algebra equation for the unknown function values at grid points. For example, consider the very simple differential equation,

$$L\tilde{u} = f, \quad \text{in } \Omega \tag{1}$$

with zero dirichlet boundary conditions

$$\tilde{u}|_{\partial\Omega} = 0 \tag{2}$$

Where L is a differential operator and $\tilde{u}$ could be a function of one or more variables ($\tilde{u}(x, y, ...t)$) [1]. A simple case which I will use as an example is $L\tilde{u} = \frac{d^2\tilde{u}}{dx^2}$, with $\tilde{u}(x)$ a plain old 1D function. Of course we can solve this problem without the use of computers, but it is illustrative of how the process works.

An important point to note is the analytic solution to the differential equation, $\tilde{u}$, is defined everywhere in Ω and can have as many kinks and bends as it likes (it is assumed C^∞). However, a computer is an inherently discrete beast so it is no surprise then that a numerical solution consists of approximating $\tilde{u}$ by values defined only at specific points in Ω (finite difference) or by a finite number of basis functions (finite element and spectral methods). If we define u to be the numerical approximation to $\tilde{u}$, all of the numerical approximations have an error dependence corresponding to the how small of a finite dimensional space we are using to approximate the analytic solution $\tilde{u}$. This error arising from casting $\tilde{u}$ into a finite setting is called the discretization error.

For example numerically solving

$$\frac{d^2\tilde{u}}{dx^2} = f(x) \quad \text{in } (0, 1) \tag{3}$$

using a finite difference scheme would amount to the following. 1) **Define a grid:** let $N + 1$ be the total number of grid points were we will solve for the approximation. Then the distance between points is $h = 1/N$ and the points are $x_i = ih$ for $i = 0, 1, ...N$. **Use an approximation for the derivatives in terms of neighboring grid points.** By taylor expanding,

$$\tilde{u}(x \pm h) = \tilde{u}(x) \pm h\tilde{u}'(x) + h^2\tilde{u}''(x) \pm h^3\tilde{u}'''(x) + h^4\tilde{u}''''(x) + ... \tag{4}$$

and then adding,

$$\tilde{u}(x + h) + \tilde{u}(x - h) = 2\tilde{u}(x) + h^2\tilde{u}''(x) + 2h^4\tilde{u}''''(x) + ... \tag{5}$$

we can rearrange and divide by h^2 to find that

$$\tilde{u}''(x) = \frac{\tilde{u}(x - h) - 2\tilde{u}(x) + \tilde{u}(x + h)}{h^2} + 2h^2\tilde{u}''''(x) + ... \tag{6}$$

The approximation takes place when we drop the terms corresponding higher orders of h^2 and define

$$u''(x) := \frac{u(x - h) - 2u(x) + u(x + h)}{h^2}. \tag{7}$$

The discretization error is

$$\tilde{u} - u \approx 2h^2\tilde{u}, \tag{8}$$

and as $h \to 0$, $u \to \tilde{u}$. However, for finite h we can always expect error in our numerical solution proportional to h^2.

Applying our finite difference approximation to the example $\frac{d^2\tilde{u}}{dx^2} = f$ gives the following system of equations at each interior grid points,

$$\frac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1})}{h^2} = f(x_i) \quad \text{for each } i = 1, 2, ..N - 1 \tag{9}$$

We only care about the interior grid points because the boundary points are known from the dirichlet boundary conditions, $\tilde{u}(x_0) = 0, \tilde{u}(x_N) = 0$. Examination of Equation 9 leads to the realization that we have traded a differential equation where the unknown is a function $\tilde{u}(x)$, for a system of equations where the unknowns are the values of a function u at specific grid points. We traded our differential equation for a linear system of equations that are conveniently written using linear algebra,

$$\frac{1}{h^2}\begin{pmatrix} -2 & 1 & & \\ 1 & -2 & 1 & \\ & \ddots & \ddots & \ddots \\ & & 1 & 2 \end{pmatrix}\begin{pmatrix} u(x_1) \\ u(x_2) \\ \vdots \\ u(x_N) \end{pmatrix} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ f(x_N) \end{pmatrix} \tag{10}$$

This is a simple example of finite differences, but the lesson is the same in most other numerical schemes; the differential equation is discretized to form a linear algebra problem,

$$Au = f. \tag{11}$$

II. SO WHAT IS MULTIGRID?

The differential equation $L\tilde{u} = f$ has been discretized to give the linear algebra problem $Au = f$, where A is a matrix coming from the discretization. The numerical solution u is an approximation to the analytic solution $\tilde{u}$ and the difference between them is discretization error. But how do we go about solving the linear algebra problem $Au = f$?

One way is by guassian elimination, and this is known as a direct solve. A direct solve will give us the exact answer u to $Au = f$ to within computer roundoff error, (which is usually small). However they are slow.

The second approach is iterative solvers. Through repeated application of a simple operation, they achieve an approximation, v, to the numerical solution u. We can work hard to make v very close to u, but after a while the effort would be wasted because the real interest lies in getting close to $\tilde{u}$. The fact that v is only approximately equal to u is not a concern, because u is only approximately equal to the analytic solution $\tilde{u}$. v just needs to be close enough to u so that the dominant source of error is the discretization error not the solver error.

The example finite difference problem provides an easy example of an iterative solver. Recall we had

$$\frac{u(x_{i-1}) - 2u(x_i) + u(x_{i+1})}{h^2} = f(x_i) \quad \text{for each } i = 1, 2, ..N-1 \tag{12}$$

Solving for $u(x_i)$ in the above equation relates $u(x_i)$ to the average of neighboring points.

$$u(x_i) = \frac{1}{2}u(x_{i-1}) + u(x_{i+1}) - \frac{h^2}{2}f(x_i) \quad \text{for each } i = 1, 2, ..N-1 \tag{13}$$

As a starting point take a random guess for each of the $u(x_i)$, with $u(x_0) = 0$ and $u(x_N) = 0$ as prescribed by the boundary conditions. By successively updating each $u(x_i)$ in terms of neighboring values an approximation to the linear algebra problem is achieved. This is the Guass-Sidel iterative method.

Call the approximation v where the entries of v are the successive approximations of each $u(x_i)$. The error between the iterative solution v and the actual solution to u is $e = u - v$. The solution u to $Au = f$ is not in general known however so a computable measure of how close the approximation v satisfies the linear algebra problem $Au = f$ is given by the residual,

$$r = f - Av. \tag{14}$$

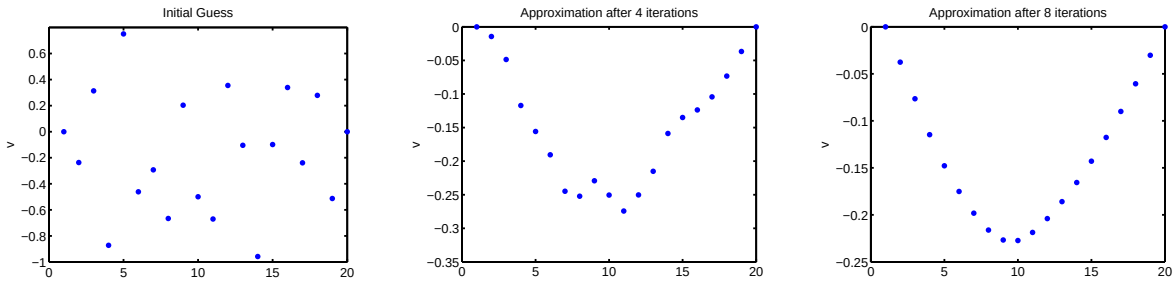
Simple mathematical manipulation gives what is called the error equation,

$$A(u - v) = f - Av \tag{15}$$

$$Ae = r \tag{16}$$

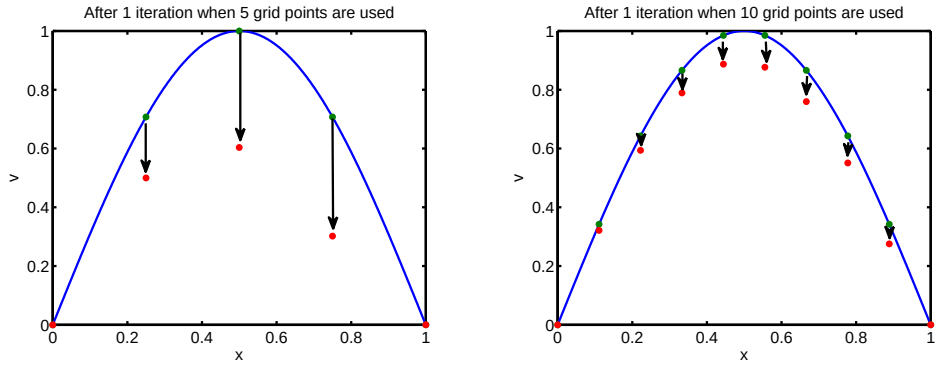
If we have an approximation v and we exactly know the error e, we can exactly know the true solution $u = e + v$. Thus given an approximation and solving the error equation is equivalent to solving the original linear algebra problem $Au = f$.

This is important because iterative solvers have the very special property of not only getting closer to the true solution, but doing so in a way that leads to the final error being smooth. For this reason they are sometimes called smoothers.



(a) (b) (c)

FIG. 1: Gauss Sidel iterations reduce the error closer to zero, but they also eliminate error in a way that is most effective against high frequency error.



(a) (b)

FIG. 2: Performing one Gauss-Sidel iteration on the smooth error $Sin(\pi x)$ with 5 grid points in (a) and ten grid points in (b). The coarser grid in (a) is able to reduce the error much more quickly than the fine grid in (b).

For example, consider when the right hand side is equal to zero, $Au = 0$. This is an especially easy case to analyze because the true solution is $u = 0$ and the error in the approximation is simply the negative of the approximation, $e = 0 - v$. Figure 1 shows a random initial guess after several Gauss-Sidel iterations. The error (and v itself in this case) are going to zero, but the error is also being made smoother.

Unfortunately for iterative solvers like the one shown above, once the error in the approximation is smooth, the averaging process will dramatically slow down in converging to the true solution.

The slowdown of iterative solvers faced with smooth error (or more generally "near nullspace modes") is the main reason for multigrid. Multigrid uses a coarser grid to solve for the error in the approximation once the iterative solver on the fine grid has stalled because of smooth error. The crucial reason this works is because smooth error is eliminated faster on coarser grids and smooth error inherently does not need a very fine grid to be accurately represented. Figure 2 shows how a smooth error that can stall an iterative solver on one grid level is much more easily handled on a grid with half as many points.

And this is the power of a multigrid V-cycle. It begins iterating on a guess for the solving $Au = f$. A couple of iterations is sufficient for the error to be smooth enough to be represented on a coarser grid. The residual is computed, $r = f - Av$ and the residual is transferred to a coarser grid, usually half as many points. The quantity of interest on this coarser grid is the error which can be approximated by iteratively solving $Ae = r$. After several iterations the residual of the approximation for this equation is computed and transferred to an even coarser grid. There the error in the error is computed iteratively. This process continues until the coarsest grid is reached at which point the correction to each of the approximation is applied to the grid above.

The matlab file gives a 1-D finite difference example, plotting the analytic solution $\tilde{u}$, linear algebra solution u, and the iterative approximation v at each grid level.

For a much more detailed explanation of introductory multigrid topics please see *A Multigrid Tutorial* by William

---

[1] $\tilde{u}$ could even be a vector function representing several unknowns, $\tilde{\mathbf{u}} = (\tilde{u}_1(x, y...t), \tilde{u}_2(x, y, ...t), ...)$, which is the case in solving the Navier-Stokes equations for the velocity and pressure of a fluid.