These notes draw heavily on *Numerical Recipes*, a valuable resource for entry-level understanding of numerical methods relevant to physics.

Boundary value problems have a somewhat different set of solutions than initial value PDEs. However, we will see that in fact solving a boundary value problem can sometimes be written as an initial value problem too.

1. Fourier Methods

Boundary value problems with constant coefficients are often amenable to solution through Fourier methods. The simplest example is the Poisson equation:

$$\nabla^2 q = \rho \tag{1}$$

What is the Fourier transform of ∇^2?

You can also think of the problem as:

$$q = \int \mathrm{d}^3x' \frac{\rho}{|x-x|'} \tag{2}$$

where $1/x$ is the Greens Function solution for a delta function source. This is the same as: $q = \rho * G$.

What is the finite difference form of the Poisson equation (in one dimension)?

Now we can write down the discrete Fourier transform of q and ρ:

$$\begin{aligned} q_j &= \frac{1}{J}\sum_{n=0}^{J-1} \tilde{q}_n e^{-2\pi i j n/J} \\ \rho_j &= \frac{1}{J}\sum_{n=0}^{J-1} \tilde{\rho}_n e^{-2\pi i j n/J} \end{aligned} \tag{3}$$

If we plug in we find:

$$\begin{aligned} \frac{1}{J}\sum_{n=0}^{J-1} \tilde{q}_n \left[e^{-2\pi i (j+1) n/J} - 2e^{-2\pi i j n/J} + e^{-2\pi i (j-1) n/J}\right] &= \Delta^2 \frac{1}{J}\sum_{n=0}^{J-1} \tilde{\rho}_n e^{-2\pi i j n/J} \\ \frac{1}{J}\sum_{n=0}^{J-1} \tilde{q}_n e^{-2\pi i j n} \left[e^{-2\pi i n/J} - 2 + e^{2\pi i n/J}\right] &= \Delta^2 \frac{1}{J}\sum_{n=0}^{J-1} \tilde{\rho}_n e^{-2\pi i j n/J} \end{aligned} \tag{4}$$

This needs to be true for any choice of j. That means that each individual term has to be the same. You can also see that this arises from the uniqueness of the Fourier transform. For the functions on the left and right to be equal, the two Fourier transforms must be equal:

$$\tilde{q}_n \left[e^{-2\pi i n/J} - 2 + e^{2\pi i n/J}\right] = \Delta^2 \tilde{\rho}_n$$

$$\begin{aligned} 2\tilde{q}_n \left[\cos\left(2\pi n/J\right) - 1\right] &= \tilde{\rho}_n \\ \tilde{q}_n &= \frac{\Delta^2 \tilde{\rho}_n}{2\left[\cos\left(2\pi n/J\right) - 1\right]} \end{aligned} \tag{5}$$

Can you explain the relationship between this expression and the division by k^2 in the continuous case?

Your colleagues will or have already shown an example of using this method!

2. Relaxation Methods

Another class of methods for solving boundary value problems is that of relaxation methods.

The basic idea is that if you have a problem of this form:

$$\mathcal{L}q = \rho \tag{6}$$

where $\mathcal{L}$ is an elliptic operator such as ∇^2, you can choose to start from a random guess for q and solve the diffusion equation:

$$\frac{\partial q}{\partial t} = \mathcal{L}q - \rho \tag{7}$$

As $t \to \infty$, q will evolve to a static solution, which therefore satisfies the original equation.

If you solve this with FTCS, for the case of Poisson's equation, it results in:

$$q_j^{n+1} = q_j^n + \frac{\Delta t}{\Delta x^2}\left(q_{j+1}^n - 2q_j^n + q_{j-1}^n\right) - \rho_j \Delta t \tag{8}$$

We need to take $\Delta t/\Delta x^2 < 1/2$ for stability as we found before. This is called the *Jacobi method*.

It turns out you can also just do this "in-place," meaning that you can just use your current round of updates:

$$q_j^{n+1} = q_j^n + \frac{\Delta t}{\Delta x^2}\left(q_{j+1}^n - 2q_j^n + q_{j-1}^{n+1}\right) - \rho_j \Delta t \tag{9}$$

This method, called *Gauss-Seidel*, converges . . . a little bit faster.

Either method converges to a solution in a number of iterations that scales as J. In N-dimensions, it will scale as J^N. This is J^2 or J^{2N} operations respectively. For large problems this is clearly problematic. Do not bother with these methods for problems greater than 100×100.

However, we can look closer to get a more practical method that sees quite a bit of use. The solution $\vec{q}$ can be written as the solution of the linear system:

$$\mathbf{A} \cdot \vec{q} = \vec{b} \tag{10}$$

and we can split $\mathbf{A}$ as follows:

$$\mathbf{A} = \mathbf{D} + \mathbf{L} + \mathbf{U} \tag{11}$$

where $\mathbf{D}$ is the diagonal, $\mathbf{L}$ is the lower part of the matrix, and $\mathbf{U}$ is the upper part.

The Jacobi method can be written:

$$\mathbf{D} \cdot \vec{q}^{n+1} = -\left(\mathbf{L} + \mathbf{U}\right) \cdot \vec{x}^n + \vec{b} \tag{12}$$

The Gauss-Siedel method can be written:

$$\left(\mathbf{L} + \mathbf{D}\right) \cdot \vec{q}^{n+1} = -\mathbf{U} \cdot \vec{x}^n + \vec{b} \tag{13}$$

This latter method forms the basis for the idea of *successive overrelaxation*. Basically, you imagine inverting $\mathbf{L} + \mathbf{D}$ and you then write the same method as:

$$\vec{q}^{n+1} = \vec{q}^n - \left(\mathbf{L} + \mathbf{D}\right)^{-1} \cdot \left[\left(\mathbf{L} + \mathbf{D} + \mathbf{U}\right) \cdot \vec{q}^n - \vec{b}\right] \tag{14}$$

The latter factor is just the residual of the solution at step n, which we can denote $\vec{\xi}^n$. Then we can write Gauss-Seidel as:

$$\vec{q}^{n+1} = \vec{q}^n - \left(\mathbf{L} + \mathbf{D}\right)^{-1} \cdot \vec{\xi}^n \tag{15}$$

This seems like a complication to Gauss-Seidel, but the last step is to imagine instead iterating with:

$$\vec{q}^{n+1} = \vec{q}^n - \omega \left(\mathbf{L} + \mathbf{D}\right)^{-1} \cdot \vec{\xi}^n \tag{16}$$

The inversion necessary to take each step is a simple one; as we will see, simple enough that it just looks like a simple update.

It can be shown that the method is convergent for $0 < \omega < 2$, though typically you want $\omega > 1$, which typically (though not always) it will give faster convergence than Gauss-Seidel.

You can write the problem as:

$$a_j q_{j+1} + b_j q_{j-1} + c_j q_j = f_j \tag{17}$$

e.g. for Poisson $a_j = b_j = 1$ and $c_j = -2$.

$\left(\mathbf{L} + \mathbf{D}\right)^{-1} \cdot \vec{\xi}$ is just defined by solving:

$$\left(\mathbf{L} + \mathbf{D}\right) \cdot \tilde{\xi} = \vec{\xi}\omega \tag{18}$$

for $\tilde{\xi}$. This is simple, because it just involves the following steps:

$$\begin{aligned} \tilde{\xi}_0 &= \frac{\xi_0}{c_0} \\ \tilde{\xi}_1 &= \frac{1}{c_1}\left(\xi_1 - b_1 \tilde{\xi}_0\right) \\ \tilde{\xi}_2 &= \frac{1}{c_2}\left(\xi_2 - b_2 \tilde{\xi}_1\right) \end{aligned} \tag{19}$$

So let's imagine stepping through in j and keeping track of the current residual $\xi_j^{\rm curr}$ based on progress so far, and using the formula:

$$q_j^{\rm new} = q_j^{\rm old} - \frac{\xi_j^{\rm curr}}{c_j} \tag{20}$$

Then:

$$q_0^{\rm new} = q_0^{\rm old} - \frac{\xi_0}{c_0} \tag{21}$$

So far so good. This step is the same as the 0th component of:

$$\vec{q}^{\rm new} = \vec{q}^{\rm old} - \tilde{\xi} \tag{22}$$

When we move on to q_1, we have:

$$q_1^{\rm new} = q_1^{\rm old} - \frac{\xi_1^{\rm curr}}{c_1} \tag{23}$$

where we can write

$$\begin{aligned} \xi_1^{\rm curr} &= -\left(f_j - a_1 q_2^{\rm old} - b_1 q_0^{\rm new} - c_1 q_1^{\rm old}\right) \\ &= \xi_1 - b_1\left(q_1^{\rm old} - q_1^{\rm new}\right) \\ &= \xi_1 - b_1\tilde{\xi}_0 \end{aligned} \tag{24}$$

So:

$$\begin{aligned} q_1^{\rm new} &= q_1^{\rm old} - \frac{\xi_1^{\rm curr}}{c_1} \\ &= q_1^{\rm old} - \frac{1}{c_1}\left(\xi_1 - b_1\tilde{\xi}_0\right) \end{aligned} \tag{25}$$

and this too corresponds to the first component of:

$$\vec{q}^{\rm new} = \vec{q}^{\rm old} - \tilde{\xi} \tag{26}$$

Clearly the update in equation 20 can be scaled by ω.

Note that when you update, the updates to odd points don't affect other odd points, only even points, and vice-versa.

The entire trick to successful successive overrelaxation techniques is the selection of ω. It turns out that an optimal choice for the Poisson equation is:

$$\omega_o = \frac{2}{1 + \sqrt{1 - \rho_J^2}} \tag{27}$$

where:

$$\rho_J = \cos(\pi/J) \tag{28}$$

It further turns out that this choice only is optimal asymptotically. The best way to proceed is to perform odd and even updates separately, and change ω on each sweep:

$$\begin{aligned} \omega^0 &= 1 \\ \omega^{1/2} &= \frac{1}{1-\rho_J^2/2} \\ \omega^{n+1/2} &= \frac{1}{1-\rho_J^2\omega^n/2} \end{aligned} \tag{29}$$

which converges to the optimal choice but starts closer to underrelaxation.