Lecture 20

Elliptic PDEs

- Elliptic equations: relaxation methods
- Successive over-relaxation.
- Cyclic reduction
- Multigrid methods.

Relaxation methods:

We ended last class discussing relaxation methods. The idea was to start with an elliptic equation and turn it into a diffusive equation,

$$\mathcal{L}u = \rho \qquad\qquad \frac{\partial u}{\partial t} = \mathcal{L}u - \rho$$

As the solution of the parabolic equation reaches its asymptotic value, the time derivatives vanish and one is effectively solving the elliptic equation we started with!

Let us consider as example the Poisson equation,

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} - \rho$$

If we tried an FTCS scheme,

$$u_{j,l}^{n+1} = u_{j,l}^{n} + \frac{\Delta t}{\Delta^2}\left(u_{j+1,l}^{n} + u_{j-1,l}^{n} + u_{j,l+1}^{n} + u_{j,l-1}^{n} - 4u_{j,l}^{n}\right) - \rho_{j,l}\Delta t$$

Taking the largest possible timestep, we get

$$u_{j,l}^{n+1} = \frac{1}{4}\left(u_{j+1,l}^{n} + u_{j-1,l}^{n} + u_{j,l+1}^{n} + u_{j,l-1}^{n}\right) - \frac{\Delta^2}{4}\rho_{j,l}$$

And this was Jacobi's method.

We also discussed the Gauss-Seidel method,

$$u_{j,l}^{n+1} = \frac{1}{4}\left(u_{j+1,l}^{n} + u_{j-1,l}^{n+1} + u_{j,l+1}^{n} + u_{j,l-1}^{n+1}\right) - \frac{\Delta^2}{4}\rho_{j,l}$$

Which used the already updated points in every "sweep".

To analyze a bit the rates of convergence (a full discussion exceeds the scope of this course) we will analyze the matrix equations resulting from both methods.

Let us switch notation and call u=x. The methods correspond to solving

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{b}$$

Let us decompose the matrix A into a diagonal, upper triangular and lower triangular matrix,

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

Jacobi's method then gives,

$$\mathbf{D} \cdot \mathbf{x}^{(r)} = -(\mathbf{L} + \mathbf{U}) \cdot \mathbf{x}^{(r-1)} + \mathbf{b}$$

Whereas the Gauss-Seidel method gives,

$$(\mathbf{L} + \mathbf{D}) \cdot \mathbf{x}^{(r)} = -\mathbf{U} \cdot \mathbf{x}^{(r-1)} + \mathbf{b}$$

In Jacobi's method, the eigenvalues of the matrix $D^{-1}(L+U)$ determine the rate of convergence. In particular the rate is dominated by the largest eigenvalue called "spectral radius" of the operator.

The number r of iterations needed to reduce the error by a factor 10^{-p} is therefore approximately,

$$r \approx \frac{p \ln 10}{(-\ln \rho_s)}$$

A detailed analysis shows that for Jacobi, for the wave equation with Dirichlet boundary conditions in all sides gives (J is # of mesh points),

$$\rho_s \simeq 1 - \frac{\pi^2}{2J^2} \qquad r \simeq \frac{2pJ^2 \ln 10}{\pi^2} \simeq \frac{1}{2} pJ^2$$

And Gauss-Seidel,

$$\rho_s \simeq 1 - \frac{\pi^2}{J^2} \quad r \simeq \frac{pJ^2 \ln 10}{\pi^2} \simeq \frac{1}{4} pJ^2$$

Since J is typically 100x100, the methods are impractical.

Let us now consider a simple modification that can make these methods useful.

Successive over-relaxation:

The idea consists in viewing the Gauss-Seidel formula in terms of the residues in each iteration. Start from,

$$(\mathbf{L}+\mathbf{D})\cdot\mathbf{x}^{(r)} = -\mathbf{U}\cdot\mathbf{x}^{(r-1)} + \mathbf{b}$$

And add and subtraxt $x^{(r-1)}$ on the RHS,

$$\mathbf{x}^{(r)} = \mathbf{x}^{(r-1)} - (\mathbf{L}+\mathbf{D})^{-1}\cdot\underbrace{[(\mathbf{L}+\mathbf{D}+\mathbf{U})\cdot\mathbf{x}^{(r-1)} - \mathbf{b}]}_{\text{Residue}}$$

$$\mathbf{x}^{(r)} = \mathbf{x}^{(r-1)} - (\mathbf{L}+\mathbf{D})^{-1}\cdot\xi^{(r-1)}$$

Now overcorrect, $\mathbf{x}^{(r)} = \mathbf{x}^{(r-1)} - \omega(\mathbf{L}+\mathbf{D})^{-1}\cdot\xi^{(r-1)}$

Where ω is the over-relaxation parameter.

It can be shown that for $0<\omega<2$ the method is convergent and that

$$\omega = \frac{2}{1+\sqrt{1-\rho^2_{\text{Jacobi}}}}$$ Is the optimal value.

Also that, $$\rho_{\mathrm{SOR}} = \left(\frac{\rho_{\mathrm{Jacobi}}}{1 + \sqrt{1 - \rho_{\mathrm{Jacobi}}^2}} \right)^2$$

So for the Poisson equation, $$\omega \simeq \frac{2}{1 + \pi/J}$$

$$\rho_{\mathrm{SOR}} \simeq 1 - \frac{2\pi}{J} \qquad \text{for large} \quad J$$

And therefore the method is of practical applicability,

$$r \simeq \frac{pJ \ln 10}{2\pi} \simeq \frac{1}{3} pJ$$

The catch consists in the choice of ω. Such tremendous improvements in convergence ratios are only achieved for a narrow window of ω's.

What does one do in practice? One possibility is to associate the problem one is considering with one for which ω is known analytically (for instance by taking as constants coefficients that are not).

If one is solving several times the same problem with slightly different parameters, one can use empirical knowledge from previous runs.

The matrix notation we used is good for theoretical analyses, but for practical implementation it is better to write things out. For a generic elliptic problem we will have equations like,

$$a_{j,l}u_{j+1,l} + b_{j,l}u_{j-1,l} + c_{j,l}u_{j,l+1} + d_{j,l}u_{j,l-1} + e_{j,l}u_{j,l} = f_{j,l}$$

For instance in our model problem we had a=b=c=d=1, e=-4.
The iterative procedure is defined as,

$$u^*_{j,l} = \frac{1}{e_{j,l}}\left(f_{j,l} - a_{j,l}u_{j+1,l} - b_{j,l}u_{j-1,l} - c_{j,l}u_{j,l+1} - d_{j,l}u_{j,l-1}\right)$$

And the updated value is a weighted average,

$$u^{\text{new}}_{j,l} = \omega u^*_{j,l} + (1-\omega)u^{\text{old}}_{j,l}$$

To calculate it we evaluate the residual,

$$\xi_{j,l} = a_{j,l}u_{j+1,l} + b_{j,l}u_{j-1,l} + c_{j,l}u_{j,l+1} + d_{j,l}u_{j,l-1} + e_{j,l}u_{j,l} - f_{j,l}$$

And the successive overrelaxation algorithm is

$$u_{j,l}^{\text{new}} = u_{j,l}^{\text{old}} - \omega \frac{\xi_{j,l}}{e_{j,l}}$$

This is easy to program and the norm of the residual can be used as a criterion of when to stop.

Another practical issue is how to sweep the mesh points. One can simply go down rows and columns. But one can also observe that if one divides the mesh into “even” and “odd” meshes like the red and black squares of a checkerboard, odd points depend only on even points and vice versa. So one could sweep, say, all the odd points updating the even ones or the other way around.

Another practical matter is that the optimal ω may not be the best initial ω. That would lead to the accumulations of errors before the algorithm converges. This can be dealt with a technique known as Chebyshev acceleration. One starts with ω=1 and "even" and "odd" meshes and eventually converges to the optimal ω.

$$\omega^{(0)} = 1$$

$$\omega^{(1/2)} = 1/(1 - \rho^2_{\text{Jacobi}}/2)$$

$$\omega^{(n+1/2)} = 1/(1 - \rho^2_{\text{Jacobi}}\omega^{(n)}/4), \qquad n = 1/2, 1, ..., \infty$$

$$\omega^{(\infty)} \rightarrow \omega_{\text{optimal}}$$

```
      SUBROUTINE sor(a,b,c,d,e,f,u,jmax,rjac)
      INTEGER jmax,MAXITS
      DOUBLE PRECISION rjac,a(jmax,jmax),b(jmax,jmax),
     *     c(jmax,jmax),d(jmax,jmax),e(jmax,jmax),
     *     f(jmax,jmax),u(jmax,jmax),EPS
      PARAMETER (MAXITS=1000,EPS=1.d-5)
```

Successive overrelaxation solution of equation (19.5.25) with Chebyshev acceleration. a, b, c, d, e, and f are input as the coefficients of the equation, each dimensioned to the grid size JMAX × JMAX. u is input as the initial guess to the solution, usually zero, and returns with the final value. rjac is input as the spectral radius of the Jacobi iteration, or an estimate of it.

```
      INTEGER ipass,j,jsw,l,lsw,n
      DOUBLE PRECISION anorm,anormf,
     *     omega,resid          Double precision is a good idea for JMAX bigger than about 25.
      anormf=0.d0               Compute initial norm of residual and terminate iteration when
      do 12 j=2,jmax-1              norm has been reduced by a factor EPS.
        do 11 l=2,jmax-1
          anormf=anormf+abs(f(j,l))      Assumes initial u is zero.
        enddo 11
      enddo 12
      omega=1.d0
      do 16 n=1,MAXITS
        anorm=0.d0
        jsw=1
        do 15 ipass=1,2                  Odd-even ordering.
          lsw=jsw
          do 14 j=2,jmax-1
            do 13 l=lsw+1,jmax-1,2
              resid=a(j,l)*u(j+1,l)+b(j,l)*u(j-1,l)+
     *              c(j,l)*u(j,l+1)+d(j,l)*u(j,l-1)+
     *              e(j,l)*u(j,l)-f(j,l)
              anorm=anorm+abs(resid)
              u(j,l)=u(j,l)-omega*resid/e(j,l)
            enddo 13
            lsw=3-lsw
          enddo 14
          jsw=3-jsw
          if(n.eq.1.and.ipass.eq.1) then
            omega=1.d0/(1.d0-.5d0*rjac**2)
          else
            omega=1.d0/(1.d0-.25d0*rjac**2*omega)
          endif
        enddo 15
        if(anorm.lt.EPS*anormf)return
      enddo 16
      pause 'MAXITS exceeded in sor'
      END
```

Operator splitting and alternating direction implicit method

$$\frac{\partial u}{\partial t} = \nabla^2 u - \rho$$

$$\mathcal{L}_x u = 2u_{j,l} - u_{j+1,l} - u_{j-1,l}$$

$$\mathcal{L}_y u = 2u_{j,l} - u_{j,l+1} - u_{j,l-1}$$

Just as before, we take two half steps

$$\frac{u^{n+1/2} - u^n}{\Delta t/2} = -\frac{\mathcal{L}_x u^{n+1/2} + \mathcal{L}_y u^n}{\Delta^2} - \rho$$

$$\frac{u^{n+1} - u^{n+1/2}}{\Delta t/2} = -\frac{\mathcal{L}_x u^{n+1/2} + \mathcal{L}_y u^{n+1}}{\Delta^2} - \rho$$

$$(\mathbf{L}_x + r\mathbf{1}) \cdot \mathbf{u}^{n+1/2} = (r\mathbf{1} - \mathbf{L}_y) \cdot \mathbf{u}^n - \Delta^2 \rho$$

$$(\mathbf{L}_y + r\mathbf{1}) \cdot \mathbf{u}^{n+1} = (r\mathbf{1} - \mathbf{L}_x) \cdot \mathbf{u}^{n+1/2} - \Delta^2 \rho$$

$$r \equiv \frac{2\Delta^2}{\Delta t}$$

And the matrices on the left hand sides are tridiagonal,
so the systems are easy to solve.

Cyclic reduction

Given an equation like $$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + b(y)\frac{\partial u}{\partial y} + c(y)u = g(x,y)$$

It will lead to a linear system, $$\mathbf{u}_{j-1} + \mathbf{T}\cdot\mathbf{u}_j + \mathbf{u}_{j+1} = \mathbf{g}_j\Delta^2$$

Where we left the y derivatives implicit in vector form. The matrix has the form T=B-2, with B tridiagonal due to the y derivatives and the factor 2 coming from the x derivative. The CR consists in writing the above equation three times,

$$\mathbf{u}_{j-2} + \mathbf{T}\cdot\mathbf{u}_{j-1} + \mathbf{u}_j = \mathbf{g}_{j-1}\Delta^2$$
$$\mathbf{u}_{j-1} + \mathbf{T}\cdot\mathbf{u}_j + \mathbf{u}_{j+1} = \mathbf{g}_j\Delta^2$$
$$\mathbf{u}_j + \mathbf{T}\cdot\mathbf{u}_{j+1} + \mathbf{u}_{j+2} = \mathbf{g}_{j+1}\Delta^2$$

Matrix multiply the middle one by –T and add the three of them.

We get, $$\mathbf{u}_{j-2} + \mathbf{T}^{(1)} \cdot \mathbf{u}_j + \mathbf{u}_{j+2} = \mathbf{g}_j^{(1)} \Delta^2$$

Which is of the same form of the equation we started with but with,

$$\mathbf{T}^{(1)} = 2\mathbf{1} - \mathbf{T}^2$$

$$\mathbf{g}_j^{(1)} = \Delta^2(\mathbf{g}_{j-1} - \mathbf{T} \cdot \mathbf{g}_j + \mathbf{g}_{j+1})$$

So we have cut down the number of equations by a factor 2. This can be iterated. Assuming the grid has an even number of points for simplicity, we will end up with a single equation for $\mathbf{u}_{J/2}$. We solve that equation, and then go back a level and start reconstructing the other elements of the solution. We sort of "unwrap" the solution from the midpoint.

Multigrid methods:

These are the modern method of choice for solving elliptic PDEs.

The key idea of these methods is to consider grids *coarser* than the one we had in mind initially.

Remember the Jacobi method. It was essentially an averaging, or if you wish a "smoothing" of the solution. Each "relaxation sweep" therefore quickly filtered out any high frequency discrepancy between the approximate solution and the real one.

What made the method slow to converge are long-wavelength errors. But those can be taken care of with a coarser mesh!

Once one notices this point there is no need to stop at two levels of mesh, one can always coarsen out further. This is the multigrid idea.

More in detail: suppose we are trying to solve a linear elliptic PDE

$\mathcal{L}u = f$ And we discretize it on a mesh of size h, $\mathcal{L}_h u_h = f_h$

And suppose that $\tilde{u}_h$ is an approximation to the solution. Then

$v_h = u_h - \tilde{u}_h$ is the error of the solution and $d_h = \mathcal{L}_h \tilde{u}_h - f_h$

is the defect or residual of the equation. Since the equation is linear,

$$\mathcal{L}_h v_h = -d_h$$

We now consider an approximation to the operator (for instance one step of iteration of Gauss-Seidel), and the approximate solution it generates,

$$\widehat{\mathcal{L}}_h \widehat{v}_h = -d_h$$

The approximate operator is “simpler” than the real one. If we were doing the Gauss-Seidel method the corrected answer would be,

$$\tilde{u}_h^{\text{new}} = \tilde{u}_h + \widehat{v}_h$$

But here we wish to consider a different approach. Consider an approximation in a coarser mesh of spacing H=2h. The residual equation would read,

$$\mathcal{L}_H v_H = -d_H$$

and would be considerably easier to solve, given the coarseness of the mesh.

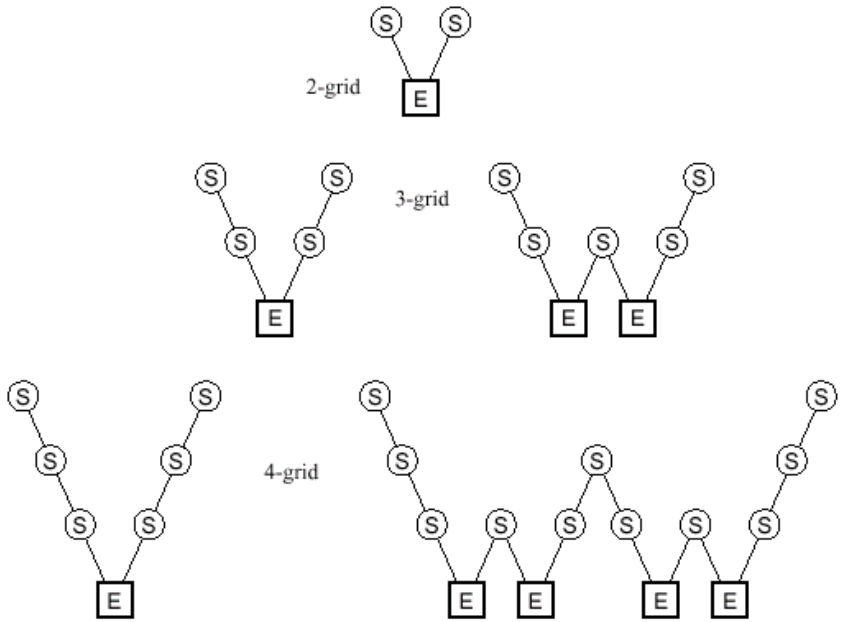
To get the residual on the coarse grid we need a "restriction" operator, also called fine-to-coarse operator,

$$d_H = \mathcal{R} d_h$$

We then generate an approximate solution $\widetilde{u}_H$ on the coarser mesh and use a "prolongation" or interpolation operator to bring the solution to the finer mesh,

$$\widetilde{v}_h = \mathcal{P}\widetilde{v}_H \qquad \widetilde{u}_h^{\text{new}} = \widetilde{u}_h + \widetilde{v}_h$$

In practice you might choose to do a few smoothings on the fine grid, move to the coarse grid and do a few smoothings there and so on. In the multigrid jargon this translates into the “V” and “W” cycles,



The prolongation and restriction operators can be described by their “symbol”, that is, what values they produce with a u that is non-zero at a point x,y. For the prolongation operator we just take linear interpolation,

$$\begin{bmatrix} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{bmatrix}$$

For the restriction operator one could simply take “1”. Sometimes it is required that it be the adjoint of the prolongation operator under the trivial inner product defined by summing functions across the grid,

$$\langle u_h | v_h \rangle_h \equiv h^2 \sum_{x,y} u_h(x,y) v_h(x,y)$$

Summary

- Jacobi and Gauss-Seidel converge too slowly.
- Overrelaxation can cure that problem.
- Operator splitting and cyclic reduction are alternatives to the Fourier method that do not involve relaxation.
- Multigrid is the modern way to solve the problems of Jacobi and Gauss-Seidel.