Week 12: Monday, Nov 5

Why iterative methods?

For many large linear systems, linear least squares problems, and eigenvalue problems, it is impractically expensive to use the sorts of $O(n^3)$ methods that we have described so far. Even sparse direct methods may be too expensive, depending on graph structure; as we mentioned earlier in the semester, these methods tend to work well for 2D PDE discretizations, but they scale more poorly for 3D problems, for example. Thus, we turn to *iterative* methods that produce a sequence of ever better guesses to the true solution to a problem, with some less-exorbitant cost per step. However, the potential speed advantages to using an iterative method come at a price: devising fast iterative methods for linear systems often requires a much deeper understanding of the underlying problem structure. Thus, while there are some good frameworks for various types of iterative solvers, they tend not to be as robust to "black box" uses as the direct solvers in LAPACK.

Stationary iterations

The oldest and simplest iterations for solving linear systems are *stationary* iterations. These iterations have largely been supplanted by more sophisticated methods (such as Krylov subspace methods), but they are still a useful building block.

Stationary iterations are so named because the solution to a linear system is expressed as finding the stationary point (fixed point) of some fixed-point iteration

$$x^{(k+1)} = F(x^{(k)}).$$

As is usually the case with fixed point iterations – linear or nonlinear – the simplest way to establish convergence is generally to establish that the mapping is a contraction, i.e.

$$\|F(x) - F(y)\| \leq \alpha \|x - y\|, \quad \alpha < 1.$$

The constant α then establishes the rate of convergence.

If we are solving a linear equation $Ax = b$, it generally makes sense to choose a fixed point iteration where the mapping F is affine. We can write any such fixed point iteration via a *splitting* of the matrix A, i.e. by writing $A = M - K$ with M nonsingular. Then we can rewrite $Ax = b$ in the form

$$Mx = Kx + b,$$

or, equivalently,

$$x = x + M^{-1}(b - Ax).$$

The fixed point iteration is then

$$x^{(k+1)} = M^{-1}(Kx^{(k)} + b) = x^{(k)} + M^{-1}(b - Ax^{(k)}).$$

If we define $R = M^{-1}K$, and $c = M^{-1}b$, we can write the iteration as

$$x^{(k+1)} = Rx^{(k)} + c;$$

and the error $e^{(k)} = x^{(k)} - x$ obeys the iteration

$$e^{(k+1)} = Re^{(k)}.$$

A sufficient condition for convergence is then that $\|R\| < 1$ in some operator norm. The actual necessary and sufficient condition is that $\rho(R) < 1$, where the spectral radius $\rho(R)$ is defined as $\max |\lambda|$ over eigenvalues λ of R.

Richardson iteration

Perhaps the simplest stationary iteration is *Richardson iteration*, in which M is chosen to be proportional to the identity:

$$x_{k+1} = x_k + \omega(b - Ax_k).$$

The iteration matrix in this case is simply $R = I - \omega A$. As long as all the eigenvalues of A have positive real part, Richardson iteration with a small enough ω will eventually converge – but that convergence may take a very long time.

In the case that A is symmetric and positive definite, the eigenvalues of $R = I - \omega A$ are $1 - \omega\lambda$, where the λ are the eigenvalues of A. Since in this case R is symmetric, $\|R\|_2$ is the largest singular value (largest eigenvalue magnitude)

$$\|R\|_2 = \max(|1 - \omega\lambda_{\max}|, |1 - \omega\lambda_{\min}|).$$

The rate of convergence is optimal when

$$|1-\omega\lambda_{\max}| = |1-\omega\lambda_{\min}|,$$

which occurs when $\omega = 2/(\lambda_{\max}+\lambda_{\min})$. In this case, the rate of convergence in the 2-norm is determined by

$$\|R\|_2 = 1 - \frac{2\lambda_{\min}}{\lambda_{\max}+\lambda_{\min}} = 1 - \frac{2}{\kappa(A)+1}.$$

Thus, if A is ill-conditioned, the iteration may be painfully slow.

Jacobi iteration

Jacobi iteration is usually introduced by talking about "sweeping" through the variables and updating each one based on the assumption that the other variables are correct. Component by component, we have

$$a_{ii}x_i^{(k+1)} + \sum_{j\neq i} a_{ij}x_j^{(k)} = b_i,$$

Alternately, we can think of Jacobi's iteration as taking $M = D$ to be the diagonal part of A. The iteration matrix in this case is

$$R = I - D^{-1}A.$$

If A is strictly row diagonally dominant, then $\|R\|_\infty < 1$, and the iteration converges.

When we discuss multigrid, we will also use as a building block the *damped Jacobi* iteration where $M = \omega^{-1}D$ for some $\omega < 1$.

Gauss-Seidel iteration

For the Jacobi iteration, we think of using equation j to update variable x_j under the assumption that the old values for all neighboring variables are correct. For the Gauss-Seidel iteration, we think of updating $x_1, x_2, \ldots$, and at each step using the most recent values of all the other variables for updates. That is, we update according to

$$\sum_{j\leq i} a_{ij}x_j^{(k+1)} + \sum_{j>i} a_{ij}x_j^{(k+1)} = b_i.$$

If we write $A = D - \tilde{L} - \tilde{U} = D(I - L - U)$ where $-\tilde{L}$ and $-\tilde{U}$ are the strictly lower and upper triangular parts of A, then Gauss-Seidel corresponds to using $M = D(I - L)$, and the iteration operator is

$$R = (I - L)^{-1}U$$

In the case of strict row diagonal dominance, $\|R\|_\infty < 1$; in fact, if R_{GS} and R_J are the iteration operators for Gauss-Seidel and Jacobi, then for strictly row diagonally dominant A we have

$$\|R_{GS}\|_\infty \leq \|R_J\|_\infty < 1.$$

We can also show that Gauss-Seidel converges in the symmetric positive definite case. Because the analysis technique will be relevant to some later discussions, we will take a moment to describe the argument. If A is symmetric positive definite, then the solution of $Ax = b$ is also the unique minimum of the quadratic function

$$\phi(x) = \frac{1}{2}x^T Ax - x^T b$$

Now suppose that $\hat{x}$ is an approximate solution, and we want to get a better solution of the form $\hat{x}' = \hat{x} + \alpha e_i$. Note that

$$\phi(\hat{x} + \alpha e_i) = \phi(\hat{x}) + \left(\frac{1}{2}\alpha^2 a_{ii} + \alpha e_i^T (Ax - b)\right),$$

which we can minimize by choosing

$$a_{ii}\alpha = e_i^T (b - Ax).$$

This exactly corresponds to the update

$$a_{ii}x^{(new)} + \sum_{j \neq i} a_{ij}x^{(prev)} = b_i.$$

Thus, Gauss-Seidel iteration can be seen as a coordinate-descent minimization algorithm with exact line searches. Furthermore, note that if $Ax = b$ and $\hat{x} = x + e$ is an approximation, then

$$\phi(\hat{x}) = \phi(x) + \frac{1}{2}e^T Ae,$$

so

$$\phi(\hat{x}) - \phi(x) = \|e\|_A^2/2,$$

where $\|e\|_A$ is the error in the "energy norm" induced by the positive definite matrix A. So in this case, the Gauss-Seidel iteration is monotonically convergent in the norm associated with A.

In many practical cases, even those that are not strictly diagonally dominant or symmetric and positive definite, Gauss-Seidel converges somewhat faster than Jacobi. But proving this requires knowing something about the structure of the problem. Outside of strictly row-diagonally dominant A, there are examples where Jacobi converges and Gauss-Seidel does not, and vice-versa.

Successive over-relaxation

If the Gauss-Seidel iteration gives us a good update, perhaps going even further in the Gauss-Seidel direction would give an even better update. This is the idea behind *SOR* (successive over-relaxation);

$$x^{(k+1)} = (1-\omega)x^{(k)} + \omega x_{GS}^{(k+1)}$$

The case where $\omega < 1$ is called *under-relaxation*; the case $\omega > 1$ is *over-relaxation*.