Week 15: Monday, Nov 30

Logistics

1. The in-class final exam will involve simple programming and basic theory. There will be a practice final so you get some sense of the types of problems I might ask.

2. The final homework is due Dec 11.

3. The plan for the next two lectures is to finish off iterative methods and to give an overview of multigrid methods. Friday's lecture will be a (unifying, I hope) review of the entire semester.

Convergence of CG

In exact arithmetic, the basic conjugate gradient algorithm computes approximate solutions to $Ax = b$ by minimizing $\|e_k\|_A = \|x_k - x\|$ where $x_k \in \mathcal{K}_k(A, b)$. Because x_k is an element of a Krylov subspace, we can write

$$x_k = p_{k-1}(A)b$$

where p_{k-1} is a polynomial of degree $k - 1$. The error is then

$$e_k = p_{k-1}(A)b - A^{-1}b = \hat{p}_k(A)e_0,$$

where $\hat{p}_k(z) = 1 - zp_{k-1}(z)$ and $e_0 = x_0 - A^{-1}b = A^{-1}b$. That is, the error at step k corresponds to choosing $\hat{p}_k(x)$ such that $\hat{p}(0) = 1$ to minimize $\|e_k\|_A$. Using the decomposition $A = Q\Lambda Q^T$, we have

$$\begin{aligned}
\|e_k\|_A^2 &= (\hat{p}_k(A)e_0)^T A(\hat{p}(A)e_0) \\
&= e_0^T Q\hat{p}_k(\Lambda)\Lambda\hat{p}_k(\Lambda)Q^T e_0 \\
&= \|\hat{p}(\Lambda)\tilde{e}_0\|_\Lambda \\
&\leq \|\hat{p}(\Lambda)\|_2^2 \, \|\tilde{e}_0\|_\Lambda^2 = \max_j \hat{p}(\lambda_j)^2 \, \|e_0\|_A^2,
\end{aligned}$$

where $\tilde{e}_0 = Q^T e_0$. Thus, we can obtain a bound by finding a family of polynomials with constant coefficient 1 that are small on a set containing the spectrum of A. This is exactly the same tactic we used when we looked at the convergence of the Lanczos iteration, and a similar argument involving Chebyshev polynomials leads us to the following theorem.

Theorem 1. *After k steps of conjugate gradient,*

$$\|e_k\|_A \leq 2\|e_0\|_A \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \right)^k,$$

where $\kappa = \kappa_2(A)$.

As was the case with the Lanczos iteration, though, a general-purpose convergence theorem for conjugate gradients is of limited usefulness because it makes no particular assumptions about the spectrum of A. In many practical problems, the convergence of the method is irregular, and depends on the distribution of eigenvalues of A, on the details of the right hand side vector b, and on the vagaries of floating point arithmetic.

On the other hand, consider the case when the eigenvalues of A are arranged in tight clusters. If there are k clusters, then we can find a degree k polynomial such that $\hat{p}_k(\lambda_j)$ is small on every cluster, and the conjugate gradient method may show very rapid convergence. For some problems (e.g. discretizations of second-kind integral equations), the matrix A does indeed have tight clusters of eigenvalues; in other cases, we can find a preconditioner such that the eigenvalues of the preconditioned problem are clustered.

GMRES

In the case where A is a nonsymmetric matrix, it is possible to solve $Ax = b$ by applying CG to the normal equations $A^T Ax = A^T b$. Unfortunately, the condition number of $A^T A$ is the square of the condition number of A, and consequently convergence of the CGNE (conjugate gradient on normal equations) method may be slow. A frequently-used alternative is the Generalized Minimal Residual (GMRES) method, which selects from each Krylov subspace $\mathcal{K}_k(A, b)$ an approximate solution x_k that minimizes $\|r_k\|^2 = \|b - Ax_k\|^2$.

By running the Arnoldi process, we can compute a matrix Q_k whose columns form an orthonormal basis for $\mathcal{K}_k(A, b)$, so that $x_k = Q_k y_k$. If we start the Arnoldi process with $q_1 = b/\|b\|$, we have

$$r_k = b - AQ_k y_k = \|b\|q_1 - Q_{k+1}\tilde{H}_{k+1} y_k,$$

where $\tilde{H}_{k+1} \in \mathbb{R}^{(k+1)\times k}$ is the leading part of a Hessenberg matrix. Therefore, y_k minimizes $\left\| \, \|b\|e_1 - \tilde{H}_{k+1} y_k \, \right\|_2$. Because $\tilde{H}_{k+1}$ is upper Hessenberg, we can apply a QR factorization to $\tilde{H}_{k+1}$ and solve the resulting linear system for y_k in $O(k^2)$ time.

In practice, the cost of storing and computing with the Arnoldi basis Q_k becomes prohibitively expensive once k gets too large. Therefore, GMRES is usually performed with restarting. That is, after some number of steps — ten or twenty, perhaps — one computes the best approximate solution $\hat{x}$ and restarts the procedure to solve the residual equation $A(x - \hat{x}) = r$, where $r = b - A\hat{x}$

Though preconditioned conjugate gradients and GMRES with restarts are among the most popular Krylov subspace methods, there are many others. A good place to read more is the Templates book.

The 1D model problem

As we noted above, it's difficult to say many useful things about the convergence of iterative methods without looking at a concrete problem. Therefore, we will finish the class by introducing multigrid methods for a very specific model problem: a discretization of the Poisson equation in one dimension.

The continuous version of our model problem is a one-dimensional Poisson equation with homogeneous Dirichlet boundary conditions:

$$\begin{aligned} -\frac{d^2u}{dx^2} &= f \text{ for } x \in (0,1) \\ u(0) &= 0 \\ u(1) &= 0 \end{aligned}$$

Let $x_j = j/(n+1)$ for $j = 0, 1, \ldots, n$ be a set of mesh points. We can approximate the second derivative of u at a point by a finite difference method:

$$-\frac{d^2u}{dx^2}(x_j) \approx \frac{-u(x_{j-1}) + 2u(x_j) - u(x_{j+1})}{h^2}$$

where $h = 1/(n+1)$ is the mesh spacing. If we replace the second derivative in the Poisson equation with this finite-difference approximation, we have a scheme for computing $u_j \approx u(x_j)$:

$$\begin{aligned} -u_{j-1} + 2u_j - u_{j-1} &= hf_j \text{ for } 1 \leq j \leq n \\ u_0 &= 0 \\ u_n &= 0 \end{aligned}$$

We can write this approximation as a matrix equation $Tu = h^2 f$, where

$$T = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

Part of what makes this simple Poisson discretization so appealing as a model problem is that we can compute the eigenvalues and eigenvectors directly. This is because solving the $(T - \lambda)\psi = 0$ is equivalent to considering the constant coefficient difference equation

$$\psi_{k+1} - (2 - \lambda)\psi_k + \psi_{k-1} = 0$$

subject to the boundary conditions $\psi_0 = \psi_{n+1} = 0$. Recall that we encountered something very like this difference equation in Lecture 33, when we described the Chebyshev polynomials. Solutions to this difference equation must have the form

$$\psi_k = \alpha \xi^k + \beta \bar{\xi}^k,$$

where ξ and $\bar{\xi}$ are the roots of the characteristic polynomial $p(z) = z^2 - (2 - \lambda)z + 1$. For $0 \leq \lambda \leq 4$, these roots form a complex conjugate pair, each with unit magnitude; that is, we can write $\xi = \exp(i\theta)$ for some θ, and so

$$\xi^k = \exp(ik\theta) = \cos(k\theta) + i \sin(k\theta).$$

Thus, any solution to the difference equation must have the form

$$\psi_k = \gamma \cos(k\theta) + \mu \sin(k\theta).$$

Plugging in the boundary conditions, we find that $\gamma = 0$, and $\theta = l\pi/(n + 1)$ for some l. Thus, the normalized eigenvectors of T are z_j with entries

$$z_j(k) = \sqrt{\frac{2}{n+1}} \sin\left(\frac{jk\pi}{n+1}\right),$$

and the corresponding eigenvalues are

$$\lambda_j = 2\left(1 - \cos\frac{\pi j}{n+1}\right).$$

In the next lecture, we shall take up the task of analyzing some iterative methods via the eigendecomposition $T = Z\Lambda Z^T$.