Week 6: Wednesday, Sep 30

Logistics

1. I will be out of town next Tuesday, October 6, and so will not have office hours on that day. Please make sure you start on the homework and ask your questions before then, since it is due on October 7!

2. We will finish talking about linear systems in the next couple lectures, and will begin on least squares problems, the QR decomposition, and the SVD. After that will be the midterm, and then eigenvalue problems.

Stability of Cholesky

Recall that if we run Gaussian elimination to compute $A = LU$, we have

$$\hat{L}\hat{U} = A + E,$$

where the backward error E satisfies the bound

$$\frac{\|E\|}{\|L\|\|U\|} = O(\epsilon_{\text{mach}}).$$

We run into problems in the unpivoted algorithm when $\|L\|\|U\| \gg \|A\|$.

For Cholesky factorization, we have a similar backward error bound:

$$\frac{\|E\|}{\|L\|\|L^T\|} = O(\epsilon_{\text{mach}}).$$

However, note that if $L = U\Sigma V^T$ is an SVD of the Cholesky factor, then

$$A = LL^T = U\Sigma^2 U^T,$$

so the singular values of A are the squares of the singular values of L. In particular, this means that $\|A\|_2 = \|L\|_2\|L^T\|_2$. Thus, the Cholesky algorithm is *backward stable* in the sense that the computed factor $\hat{L}$ satisfies

$$\hat{L}\hat{L}^T = A + E$$

where $\|E\|/\|A\| = O(\epsilon_{\text{mach}})$.

Here is a more heuristic way to see what is going on. Suppose A is some general nonsymmetric matrix, and a leading principle minor A_{11} is nearly singular (having almost a one-dimensional null space). Then we write

$$A_{11} = U\Sigma V^T$$

where $\sigma_n \ll 1$, and we find that if x is a random vector, then $A_{11}^{-1}x$ tends to align with the null vector:

$$A_{11}^{-1}x \approx v_n \sigma_n^{-1}(u_n^T x).$$

Thus, after the factorization routine has processed A_{11}, we have

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} L_{11} & 0 \\ L_{21} & I \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & A_{22} - A_{21}A_{11}^{-1}A_{12} \end{bmatrix},$$

and the Schur complement is likely to be dominated by the contribution $-A_{21}A_{11}^{-1}A_{12}$, which is in turn mostly made up of the rank-1 contribution $-(A_{21}v_n)(u_n^T A_{12})/\sigma_n$. Thus, the Schur complement likely becomes very ill-conditioned; and If everything else involved is much smaller than this rank 1 piece, then it tends to be drowned due to inexact arithmetic effects. This cannot happen with a symmetric positive definite matrix, since the (2-norm) condition number of any minor or Schur complement is bounded by the condition number of the original matrix.

Diagonally dominant matrices

Part of the appeal of the Cholesky factorization is that it requires roughly half the flops and memory of standard Gaussian elimination. But much of the appeal is that there is no need to pivot. Another class of matrices where Gaussian elimination can be done without pivoting is the *diagonally dominant* matrices.

A matrix A is (strictly) column diagonally dominant if for every j,

$$|A_{jj}| \geq \sum_i |A_{ij}|.$$

If we write $A = D + F$, where D is the diagonal part and F is the off-diagonal part, then A is diagonally dominant iff

$$\|D^{-1}F\|_1 = \max_j \left(\sum_i |A_{ij}|/|A_{jj}| \right) < 1.$$

If $\|D^{-1}F\|_1 < 1$, then we know $I + D^{-1}F$ is invertible with

$$\|(I + D^{-1}F)^{-1}\|_1 \leq \frac{1}{1 - \|D^{-1}F\|_1},$$

and so $A = D(I + D^{-1}F)$ is invertible with

$$\|A^{-1}\|_1 \leq \frac{\|D^{-1}\|_1}{1 - \|D^{-1}F\|_1}.$$

So a diagonally dominant matrix is always nonsingular, and we have a bound on the inverse. From the analysis of iterative refinement that we did earlier, we also know that iterative refinement using D^{-1} as an approximate inverse,

$$x_{k+1} = x_k + D^{-1}(b - Ax_k),$$

converges linearly.

While diagonally dominant matrices have a number of wonderful properties, some of which we will return to later in the course, the one that concerns us today is the fact that Gaussian elimination with partial pivoting does no row exchanges when factoring a diagonally dominant matrix. In the first step, it is clear that we don't need to pivot; by construction, A_{11} has the largest magnitude of the elements in the first column. What is slightly less obvious, and is left as an exercise to the inquisitive reader (who might look at Theorem 3.4.3 in the book), is that the Schur complement after the first step is again diagonally dominant. Thus, the next step also requires no row exchange, and so on.

Sparse matrices and fill-in

Consider the following *sparse* matrix, where to make the nonzero structure more obvious, we will leave blank spaces where there are zero entries in the matrix:

$$A = \begin{bmatrix} 1 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.1 & 1 & & & \\ 0.1 & & 1 & & \\ 0.1 & & & 1 & \\ 0.1 & & & & 1 \end{bmatrix}.$$

If we perform one step of Gaussian elimination on A, we find that the Schur complement immediately becomes dense, with every element nonzero. Putting in $\times$ to indicate a nonzero element, we have

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & & & \\ \times & & \times & & \\ \times & & & \times & \\ \times & & & & \times \end{bmatrix} = \begin{bmatrix} \times & & & & \\ \times & \times & & & \\ \times & \times & \times & & \\ \times & \times & \times & \times & \\ \times & \times & \times & \times & \times \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix}.$$

That is, L and U have many more nonzeros than A. These nonzero locations that appear in L and U and not in A are called *fill-in*.

Now, suppose that we cyclically permute the rows and columns of A to get

$$PAP^T = \begin{bmatrix} 1 & & & & 0.1 \\ & 1 & & & 0.1 \\ & & 1 & & 0.1 \\ & & & 1 & 0.1 \\ 0.1 & 0.1 & 0.1 & 0.1 & 1 \end{bmatrix}.$$

Structurally, the LU factorization of PAP^T looks like

$$\begin{bmatrix} \times & & & & \times \\ & \times & & & \times \\ & & \times & & \times \\ & & & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} = \begin{bmatrix} \times & & & & \\ & \times & & & \\ & & \times & & \\ & & & \times & \\ \times & \times & \times & \times & \times \end{bmatrix} \begin{bmatrix} \times & & & & \times \\ & \times & & & \times \\ & & \times & & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix}.$$

That is, the factorization of PAP^T has *no* fill-in. Clearly, the ordering of equations and unknowns matters! Unfortunately, even leaving aside the possible need for pivoting in order to ensure stability, optimally ordering the rows and columns of A in order to minimize fill-in is an NP-complete problem (i.e. one for which we only know exponential-time algorithms). We can do pretty well with heuristic orderings, though, and most sparse matrix packages provide access to such orderings. For example, in MATLAB, you would usually do a sparse factorization of a matrix A with the command

```
[L,U,P,Q] = lu(A);
```

which computes the factorization

$$PAQ = LU,$$

where P and Q are permutation matrices and L and U are (sparse) lower and upper triangular factors. The column permutation Q is chosen to try to ensure sparsity of L and U; the row permutation P is chosen to ensure stability of the factorization. By default, P is the permutation given by the usual partial pivoting rule, but the user can pass in arguments to use a relaxed "threshold" pivoting strategy, in which a row exchange only occurs if some candidate pivot element is larger than the diagonal element by a sufficiently large margin. See the MATLAB help on `lu` for details.