Lecture Notes 8, Math/Comp 128, Math 250

Misha Kilmer
Tufts University

October 16, 2007

Gaussian Elimination

Standard in linear algebra class (reduce to row echelon form): eliminate zeros below the main diagonal at each step, using legal row operations (e.g. $R_i \leftarrow \alpha R_i + \beta R_j$) until the (augmented) matrix is upper triangular. Then see if you can solve the system by backsubstitution.

Assuming all diagonals remain non-zero, the triangularization process is completely equivalent to multiplying A on the left by a sequence of special, lower triangular matrices L_k:

$$\underbrace{L_{m-1} \cdots L_2 L_1}_{L^{-1}} A = U.$$

Thus, $A = LU$.

GE

We'll only be considering this for square matrices. Recall we have other tools (SVD, QR) for finding bases for $\mathcal{R}(A), \mathcal{N}(A)$, whether or not A is square.

Show symbolically on the board the triangularization process for a 4×4 matrix.

Example

$$A = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix}$$

Introduce zeros below the main diagonal.

- 2 is the “pivot”
- multipliers are $\mu_{2,1} = \frac{4}{2} = 2$, $\mu_{3,1} = \frac{8}{2} = 4$, $\mu_{4,1} = \frac{6}{2} = 3$.
- Replace R_i by $R_i - \mu_{i,1} R_1$, $i = 2, 3, 4$

$$
\begin{aligned}
L_1 A &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ -4 & 0 & 1 & 0 \\ -3 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} \\
&= \begin{bmatrix} 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 3 & 5 & 5 \\ 0 & 4 & 6 & 8 \end{bmatrix}
\end{aligned}
$$

Example, cont

- the 1 in the 2,2 position is now the pivot
- multipliers are $\mu_{3,2} = \frac{3}{1} = 2$, $\mu_{4,2} = \frac{4}{1} = 4$.
- Replace R_i by $R_i - \mu_{i,2} R_2$, $i = 2, 3$.

$$
\begin{aligned}
L_2 L_1 A &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 3 & 5 & 5 \\ 0 & 4 & 6 & 8 \end{bmatrix} \\
&= \begin{bmatrix} 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 4 \end{bmatrix}
\end{aligned}
$$

Now the 2 in the 3,3 position is the pivot. So $\mu_{4,3} = \frac{2}{2} = 1$ and

$$
\begin{aligned}
L_3L_2L_1A &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 2 & 4 \end{bmatrix} \\
&= \begin{bmatrix} 2 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 2 \end{bmatrix}
\end{aligned}
$$

Where is L?

Since $L_3L_2L_1A = U$, and L_i are clearly invertible (why?!), we have

$$A = \underbrace{L_1^{-1}L_2^{-1}L_3^{-1}}_{L}U.$$

Claim 1: L as defined here is lower triangular.

Proof:

- The inverse of a lower triangular matrix is lower triangular (theorem, proved by the 250 students)
- The product of lower triangular matrices is lower triangular (theorem, previous homework)

Claim 2: In general, L is a unit lower triangular matrix (NOTE: we are assuming no pivots are zero, or else the L_i are undefined!), and that the entries below the main diagonal in L are the $\mu_{i,j}, i = 2, \ldots, m; j = 1, \ldots, m-1$, while the diagonal entries are 1's.

Proof: Note we can write $L_k = I - \ell_k e_k^T$, and that L_k^{-1} is therefore $I + \ell_k e_k^T$ (check that $L_k L_k^{-1} = I$!)

Because of the sparsity (zero) patterns, note that $e_j^* \ell_k = 0$ if $j \neq k$. So, cross terms cancel. For example, $L_k^{-1} L_{k+1}^{-1} = (I + \ell_k e_k^*)(I + \ell_{k+1} e_{k+1}^*) = I + \ell_k e_k^* + \ell_{k+1} e_{k+1}^*$. Hence,

$$L = L_1^{-1} \cdots L_{m-1}^{-1} = \begin{bmatrix} 1 & & & & \\ \mu_{2,1} & 1 & & & \\ \mu_{3,1} & \mu_{3,2} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \mu_{m,1} & \mu_{m,2} & \cdots & \mu_{m,m-1} & 1 \end{bmatrix}$$

Practical GE without pivoting

Assuming we *want* L, U explicitly, otherwise, we can overwrite A by U.

- $U = A; L = eye(m)$;
- for k=1:m-1

  for j= k+1:m

  L(j,k)=U(j,k)/U(k,k);

  U(j,k:m) = U(j,k:m) - L(j,k)*U(k,k:m);

Flop count: 2nd line in the inner loop is 1 scalar mult, one (row) vector subtraction, length $m - k + 1$.

$$\sum_{k=1}^{m-1} \sum_{j=k+1}^{m} 1 + 2(m - k + 1) \leq \sum_{k=1}^{m-1} \sum_{j=k+1}^{m} 3m = O(m^3)$$

Practical GE without pivoting

- When A is banded, we can be more clever about our algorithms.
- Example, tridiagonal matrix: only ever 1 non-zero below the diagonal to eliminate. Let's work this out.
- When A is a sparse matrix, note the potential "fill in" in the L and U factors. Bandwidth vs. sparsity pattern. (airfoil example in matlab)
- When A is real, symmetric, possible to modify to get LDL^T factorization (assuming the LU factorization exists)

Stability of LU

Note that $\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$ has no LU factorization!

Consider $A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}$. An LU factorization does exist, but in double precision arithmetic, we compute $L = \tilde{L} = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix}$, but $U = \begin{bmatrix} 10^{-20} & 1 \\ 0 & 1 - 10^{20} \end{bmatrix}$ while $\tilde{U} = \begin{bmatrix} 10^{-20} & 1 \\ 0 & -10^{20} \end{bmatrix}$.

Note that $\tilde{L}\tilde{U} = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 0 \end{bmatrix}$. Using this to solve $Ax = b$ with $b = [1; 0]$ and backward, forward substitution, we get about 100 percent error in the first component!

In general, GE without pivoting (permutations of rows and/or columns) is not backward stable. However, forward and backward substitution is backward stable. Thus, GE with no pivoting + forward/back sub, as an algorithm, is not backward stable.

Pivoting for Stability

Consider $A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}$ and let $P = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$.

Then $PA = \begin{bmatrix} 1 & 1 \\ 10^{-20} & 1 \end{bmatrix}$. Now compute L and U on this:

$$\tilde{L} = \begin{bmatrix} 1 & 0 \\ 10^{-20} & 1 \end{bmatrix}; \qquad \tilde{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

Observe that $\tilde{L}\tilde{U} = \begin{bmatrix} 1 & 1 \\ 10^{-20} & 1 + 10^{-20} \end{bmatrix}$. So

$$\|\tilde{L}\tilde{U} - PA\|_\infty / \|PA\|_\infty = 10^{-20}/2$$

we have computed the exact factors of a nearby problem.

To solve $Ax = b$, $PAx = Pb \rightarrow \tilde{L}\tilde{U}x = Pb$. For the same b that gave us problems before, the solution is nearly perfect!

GE with Partial Pivoting

Idea: Before you eliminate zeros below the pivot entry, look down the numbers (in the current column) to be zeroed, compare to the pivot entry. Swap rows for the row with the largest magnitude entry, and continue.

Equivalent to

$$L_{m-1}P_{m-1}\cdots L_2P_2L_1P_1A = U.$$

Example

See book for rest of the details.

$$A = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix}$$

$$P_1 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} ; L_1 P_1 A = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & -1/2 & -3/2 & -3/2 \\ 0 & -3/4 & -5/4 & -5/4 \\ 0 & 7/4 & 9/4 & 17/4 \end{bmatrix}$$

$$P_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$L_2 P_2 L_1 P_1 A = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & 7/4 & 9/4 & 17/4 \\ 0 & 0 & -2/7 & 4/7 \\ 0 & 0 & -6/7 & -2/7 \end{bmatrix}$$

Finally, swap the 3rd and 4th rows, compute an L_3, and

$$L_3 P_3 L_2 P_2 L_1 P_1 A = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 0 & 7/4 & 9/4 & 17/4 \\ 0 & 0 & -6/7 & -2/7 \\ 0 & 0 & 0 & 2/3 \end{bmatrix} = U$$

Where is L?? Claim that the info to construct both P and L is in there, such that $PA = LU$.

$$L_3 P_3 L_2 P_2 L_1 P_1 = L_3' L_2' L_1' P_3 P_2 P_1$$

where

$$L_k' = P_{m-1} \cdots P_{k+1} L_k P_{k+1}^{-1} \dots P_{m-1}^{-1}$$

and note that $P_j^{-1} = P_j^T$, because it's a permutation matrix.

L_k was unit lower triangular. Because the permutations are applied "symmetrically" (i.e. same permutation of rows [left mult by P_j] as permutation of columns [right mult by P_j^T]), L_k' is going to be unit lower triangular still, but the non-zeros in the lower part may have switched position.

So now $L = (L_1')^{-1}(L_2')^{-1}(L_3')^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3/4 & 1 & 0 & 0 \\ 1/2 & -2/7 & 1 & 0 \\ 1/4 & -3/7 & 1/3 & 1 \end{bmatrix}$

GEPP

- U=A, L=I, P=I
- for k=1:m-1

    select $i \geq k$ to maximize $|U(i,k)|$

    swap $U(k, k:m)$ with $U(i, k:m)$

    swap $L(k, 1:k-1)$ with $L(i, 1:k-1)$

    swap $P(k,:)$ with $P(i,:)$

    for j=k+1:n

      L(j,k) = U(j,k)/U(k,k);

      U(j,k:m) = U(j,k:m) - L(j,k)*U(k,k:m)

Odds and Ends

- The same number of Flops as without pivoting.
- Overhead for the search down the columns at each step.
- In practice, P is not created explicitly.
- If only desire is to solve a system, don't need L, U, can overwrite A and compute Pb on the fly.
- Beware! Permutations can destroy structure!
- Storage and fill-in. Bad bet for sparse problems.
- Cases where no pivoting is needed for stability.
- Approximate ILU for preconditioning.

More on PLU

$$PA = LU$$

Remember, if we have partial pivoting, ALL entries in L are less than or equal to 1 in magnitude.

See 4×4 example pg. 157-159 (needs 3 steps)

$$L_3 P_3 L_2 P_2 L_1 P_1 A = U$$

Means ($m = 4$)

$$\underbrace{(L_3' L_2' L_1')}_{L^{-1}} \underbrace{(P_3 P_2 P_1)}_{P} A = U$$

where $L_k' = P_{m-1} \cdots P_{k+1} L_k P_{k+1}^{-1} \cdots P_{m-1}^{-1}$.

Note that L_k' will be unit lower triangular, so L^{-1} and L will be, too.

PLU

If the entries in L are kept in check (note $\|L\| = O(1)$), what about entries in U?

Definition: The growth factor for A is

$$\rho = \frac{\max_{i,j} |u_{ij}|}{\max_{i,j} |a_{ij}|}$$

Implies $\|U\| = O(\rho\|A\|)$.

Theorem 22.2: Given the computation of $PA = LU$ by Alg. 21.1, then the computed mats $\tilde{P}, \tilde{L}, \tilde{U}$ satisfy

$$\tilde{L}\tilde{U} = \tilde{P}A + \delta A, \qquad \frac{\|\delta A\|}{\|A\|} = O(\rho\epsilon_{mach}).$$

Means that backward stability *depends on the growth factor*.

Worst-case Instability

See example matrix (22.4) in text. Growth factor is 2^{m-1}.

Fortunately, evil growth factors like this one are not common in practice.

"Large factors U like (22.5) never seem to appear in real applications."

No pivoting needed?

When is no pivoting needed to ensure stability?

- Matrix is strictly column diagonally dominant

$$|a_{kk}| > \sum_{j \neq k} |a_{jk}|, \qquad k = 1, \ldots, m$$

- Matrix is symmetric/Hermitian positive definite

$A^* = A$ and $x^* A x > 0$ for every nonzero x

Sparse and Banded Matrices

Definition: A sparse matrix is one with a high percentage of zero entries.

(maybe 20 percent or fewer)

Example from first day of class: Discrete partial differential operators.

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u(x_{i+1}) - 2u(x_i) + u(x_{i-1})}{h^2}$$

etc.

So $\Delta u(x,y) + \gamma^2 u(x,y) = g(x,y)$, $u(x,y) = 0$ on boundary of square, leads to a set of n^2 equations in n^2 unknowns on an $n \times n$ grid.

Sparse

Airfoil example in Matlab.

Different orderings of the unknowns and equations give rise to different amounts of fill in the L and U factors.

Orderings: `symrcm, colperm, colmmd, symmmd`

Orderings for Sparse Matrices

Bottom line: The L and U factors can take up significantly more space to store than A. Active research problem. Try to reorder rows and cols to minimize fill BUT STILL GET STABILITY. (Sometimes, these can be in conflict.)

Banded Matrices

Definition: A is banded with bandwidth 2p+1 if all the non-zeros are contained within the first p super and first p subdiagonals. ($a_{ij} = 0, |i - j| > p$).

Definition: A has upper bandwidth q if $a_{ij} = 0, j > i + q$ and lower bandwidth r if $a_{ij} = 0, i > j + r$.

Banded Matrices

It can be shown that if $A = LU$ (i.e. no pivoting) exists and A has upper bandwidth q and lower bandwidth r then U has upper bandwidth q and L has lower bandwidth r.

We should take advantage of this in our algorithms!

Banded Matrices

If partial pivoting is needed, at worst the upper bandwidth of U is $q + r$ (whereas L still has at most $r + 1$ per column).

Our loops in finding L,U should be adjusted for bandwidth, as should our forward and backward substitution algorithms! (Do forward sub.)

Nice data structures for storing banded matrices. (Generic sparse are more difficult, since you don't a priori know where fill will occur.)

Cholesky Factorization

Works only for

- $A \in \mathbb{R}^{n \times n}$ and A is SPD.
- $A \in \mathbb{C}^{n \times n}$ and A is HPD.

Regardless, we have $A^* = A$, $x^* A x > 0$ for all nonzero x. (Note $x^* A x$ is real.)

Cholesky Factorization

Find upper triangular R such that $A = R^*R, r_{jj} > 0$.

Theorem 23.1 Every HPD $A \in \mathbb{C}^{m \times m}$ has a unique Cholesky factorization.

Proof 1: by construction.

Proof 2: From the $LU = LDM^T$ factorization. First show $M = L$. Then show D is positive. Then fold $D^{1/2}$ into each term.

Cholesky (Overwriting)

- $R = A$
- for $k = 1 : m$

  for $j = k + 1 : m$

  $R(j, j : m) = R(j, j : m) - (R(k, j)/R(k, k)) * R(k, j : m);$

  end

  $R(k, k : m) = R(k, k : m)/\sqrt{R(k, k)}$
- end

Cholesky

No need for pivoting... bwd. stable, $O(m^3)$ algorithm.

Theorem 23.2 Let $A \in \mathbb{C}^{m\times m}$ be HPD, run (23.1). For all sufficiently small ϵ_{mach}, it will run to completion and generate $\tilde{R}$:

$$\tilde{R}^*\tilde{R} = A + \delta A, \qquad \frac{\|\delta A\|}{\|A\|} = O(\epsilon_{mach}).$$

Note that it may not produce accurate $\tilde{R}$:

$$\|\tilde{R} - R\| / \|R\| = O(\kappa(A)\epsilon_{mach})$$

(only the product satisfies the backstable error bound.)

Cholesky and $Ax = b$

Recall forward/backsub as in text are backward stable algorithms. Consequently,

Theorem 23.3 The solution of HPD systems $Ax = b$ via Cholesky (Alg 23.1) and fwd/back sub is backward stable in that $\tilde{x}$ satisfies

$$(A + \delta A)\tilde{x} = b, \qquad \frac{\|\delta A\|}{\|A\|} = O(\epsilon_{mach})$$

Misc.

- May know if A is HPD from application.
- If `chol(A)` gives an error, probably not HPD
- HPD matrices are nice for other reasons, too!
- Hermitian matrices that are strictly DD are HPD.