Intro to Numerical Linear Algebra for AMSC 460
Lecture 4

Prof. Jacob Bedrossian
University of Maryland, College Park

These notes will supplement the lectures on numerical linear algebra. NLA is one of the most important, if not the most important, branches of numerical analysis. For many scientific computing calculations, most of the compute time is spent doing linear algebra, and so doing NLA effectively is crucial to useful code. Probably the best reference I've ever seen to introduce the topic of NLA is the book Trefethen and Bau, but its too advanced to follow directly in AMSC 460. These notes are a distillation of Bindle+Goodman's notes on scientific computing and Trefethen and Bau's book.

1 Cholesky factorization

Next we will learn a very simple but powerful variant of LU factorization that works on very special matrices.

Definition 1. A matrix $A \in \mathbb{R}^{n\times n}$ is *symmetric* if $A = A^T$.

Definition 2. A matrix $A \in \mathbb{R}^{n\times n}$ is *positive definite* if for all $x \in \mathbb{R}^n$ with $x \neq 0$, there holds $x \cdot Ax = x^T Ax > 0$.

A matrix A is (of course) called *symmetric positive definite*, abbreviated *SPD*, if it is both symmetric and positive definite. One easy consequence of being positive definite is the following.

Lemma 1. *If A is positive definite, then any diagonal element of A is strictly positive: $a_{jj} > 0$ for all $1 \leq j \leq n$.*

Proof. Let $e_j \in \mathbb{R}^n$ be a vector which is zero in every entry except for the $j-th$ coordinate and the j-th coordinate is equal to one (e.g. $e_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and so forth). Then, from matrix multiplication,

$$e_j^T A e_j = a_{jj}.$$

By positive definiteness, this must be positive. □

Let us consider specializing the Gaussian elimination algorithm to an SPD matrix A for which the first diagonal element is equal to one:

$$A = \begin{pmatrix} 1 & -- w^T -- \\ | & \\ w & K \\ | & \end{pmatrix},$$

where here $w \in \mathbb{R}^{n-1\times 1}$ and $K \in \mathbb{R}^{n-1\times n-1}$. Note that necessarily $K = K^T$ since $A = A^T$. The first of the LU factorization would do the elimination of subdiagonal column w:

$$A = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ | & & & \\ w & & I & \\ | & & & \end{pmatrix} \begin{pmatrix} 1 & -- w^T -- \\ | & \\ 0 & K - ww^T \\ | & \end{pmatrix}.$$

Here $I \in \mathbb{R}^{n-1 \times n-1}$ denotes the identity matrix. Note that ww^T is a *matrix* for which the i, j element is:

$$(ww^T)_{ij} = w_i w_j.$$

Note also that $(ww^T)x = (w^T x)w$. This first step is *exactly* the first step of the LU factorization, just written in block matrix form. Now, we are going to use symmetry to also factor out a matrix from the right:

$$\begin{aligned} A &= \begin{pmatrix} 1 & 0 & \cdots & 0 \\ | & & & \\ w & & I & \\ | & & & \end{pmatrix} \begin{pmatrix} 1 & - - 0 - - \\ | & \\ 0 & K - ww^T \\ | & \end{pmatrix} \begin{pmatrix} 1 & - - w^T - - \\ | & \\ 0 & I \\ | & \end{pmatrix} \\ &= R_1^T A_1 R_1. \end{aligned}$$

Hence, we could symmetrically factor the matrix. This suggests that if we have a non-zero element in the diagonal, we should split it evenly between the left and right matrices so that they are again just transposes of eachother:

$$\begin{aligned} A &= \begin{pmatrix} \sqrt{a_{11}} & 0 & \cdots & 0 \\ | & & & \\ w/\sqrt{a_{11}} & & I & \\ | & & & \end{pmatrix} \begin{pmatrix} 1 & - - 0 - - \\ | & \\ 0 & K - \frac{ww^T}{a_{11}} \\ | & \end{pmatrix} \begin{pmatrix} \sqrt{a_{11}} & - - w^T/\sqrt{a_{11}} - - \\ | & \\ 0 & I \\ | & \end{pmatrix} \\ &= R_1^T A_1 R_1. \end{aligned} \tag{1}$$

Again, this is not morally really any different from the normal LU factorization algorithm, and the point of doing this is not yet clear. Note that from the lemma above, $a_{11} > 0$ since A is positive definite.

Next, what we want to do is to continue the algorithm, now applying the same idea to the sub-matrix $K - \frac{ww^T}{a_{11}}$. This requires the following observation, which is not obvious.

Lemma 2. *If A is SPD, then so is the sub-matrix $K - \frac{ww^T}{a_{11}}$.*

Proof. Since $(ww^T)^T = ww^T$, it follows that the sub-matrix is symmetric.

Let $u \in \mathbb{R}^{n-1}$ be non-zero. We want to prove that:

$$u^T \left(K - \frac{ww^T}{a_{11}} \right) u > 0.$$

Define the vector $y \in \mathbb{R}^n$ via:

$$y = \begin{pmatrix} 0 \\ u_1 \\ \vdots \\ u_{n-1} \end{pmatrix}.$$

Then we observe that

$$u^T \left(K - \frac{ww^T}{a_{11}} \right) u = y^T A_1 y,$$

where A_1 is defined up in (1). The matrix R_1 is invertible: it is upper triangular, so we can compute that the determinant is $det(R_1) = \sqrt{a_{11}} \neq 0$. Hence, we can define $x = R_1^{-1}y$ and so

$$y^T A_1 y = (R_1 x)^T A_1 (R_1 x) = x^T (R_1^T A_1 R_1) x = x^T A x.$$

Note that since R_1 is invertible and $y \neq 0$, it follows that $x \neq 0$ as well. Summing up, we have

$$u^T \left(K - \frac{ww^T}{a_{11}} \right) u = x^T A x > 0,$$

and so the proof is complete. □

Therefore, we can do the same symmetric elimination step on the sub-matrix as we did on A. That is, if we write

$$K - \frac{ww^T}{a_{11}} = \begin{pmatrix} u_{22} & - - v^T - - \\ | & & \\ v & K^{(2)} & \\ | & & \end{pmatrix},$$

then we further eliminate via:

$$\begin{aligned} A &= R_1^T R_2^T \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots \\ 0 & 0 & & \\ | & | & K^{(2)} - \frac{vv^T}{u_{11}} & \\ 0 & 0 & & \end{pmatrix} R_2 R_1 \\ &= R_1^T R_2^T A_2 R_2 R_1, \end{aligned}$$

where

$$R_2^T = \begin{pmatrix} 1 & 0 & \cdots & \cdots & 0 \\ 0 & \sqrt{u_{22}} & 0 & \cdots & 0 \\ \vdots & | & & & \\ \vdots & v/\sqrt{u_{22}} & & I & \\ 0 & | & & \cdots & \end{pmatrix}$$

Observe actually that

$$R_1^T R_2^T = \begin{pmatrix} \sqrt{a_{11}} & 0 & \cdots & \cdots & 0 \\ | & \sqrt{u_{22}} & 0 & \cdots & 0 \\ | & | & & & \\ w/\sqrt{a_{11}} & v/\sqrt{u_{22}} & & I & \\ | & | & & \cdots & \end{pmatrix}, \tag{2}$$

which is analogous to what happens in the LU factorization algorithm. Now, it makes sense what we are going to do: we will continue eliminating on the submatrices until we only have the identity left:

$$\begin{aligned} &A = R_n^T R_{n-1}^T ... R_1^T I R_1 R_2 ... R_n = R^T R, \\ &R := R_1 R_2 ... R_n. \end{aligned}$$

Hence, we have factored A into a lower triangular matrix R^T and an upper triangular matrix R, just like LU, except that here the lower and upper pieces are just transposes of each-other.

The Cholesky factorization has two advantages over LU: firstly, and maybe most importantly, it turns out (for reasons beyond the scope of this course), that we never have to do any pivoting and we still have a very stable factorization algorithm. This makes the implementation of Cholesky factorizations much simpler and more reliable. Secondly, symmetric matrices contain roughly half the amount of information as non-symmetric matrices (for a non-symmetric matrix, we have to have both L and U, but for an SPD matrix, we just need to keep R). Hence, we should be able to write a Cholesky factorization which runs in about half the time as an LU factorization. That is, we should expect to be able to find an algorithm which only costs $\frac{1}{3}n^3 + O(n^2)$. This does indeed turn out to be the case.

The pseudo-code will actually build R row-by-row, but remember that this is the same as building R^T column by column, so it still is the same as LU. Moreover, since A is symmetric, we will only need to look at half the matrix. We will look at the top half, but of course one could write the algorithm to look at the other half.

Algorithm 1 (Cholesky factorization)**.** The pseudo-code assumes vectors are indexed from one.

```
R = A
for k = 1 to n:          #loop over diagonals
    for j=k+1 to n:      #apply elimination step to submatrix
        b = R[k,j]/R[k,k]
        for i = j to n:
            R[j,i] = R[j,i] - b*R[k,i]
    R[k,k:n] = R[k,k:n]/(R[k,k])^{1/2}  #Set next row of R
```

I will leave it as an exercise to you to compute that the cost in flops is $Cost(n) = \frac{1}{3}n^3 + O(n^2)$.