Numerical Analysis Review

Chapter 1

December 20, 2020

This chapter contains following algorithms.

- Forward/backward elimination
- LU decomposition (with pivot)
- Cholesky decompositon
- $LDL^\top$ decomposition

Forward/backward elimination

Consider a linear system given by

$$Ax = b$$

and we assume that $A \in \mathbb{R}^{n\times n}$ is full-rank.

Specially if we have $A = L$ or U, where

$$L = \begin{pmatrix} l_{11} & & & \\ l_{21} & \ddots & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{pmatrix}, \qquad U = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ & \ddots & \ddots & \vdots \\ & & \ddots & u_{n-1,n} \\ & & & u_{nn} \end{pmatrix}$$

, then the system can be solved in a recursive manner. e.g., for $Lx = b$, we have

$$x_k = \frac{1}{l_{kk}}\left(b_k - \sum_{i=1}^{k-1} l_{ki}x_i \right), k = 1, \dots, n$$

and for $Ux = b$, we have

$$x_k = \frac{1}{u_{kk}}\left(b_k - \sum_{i=k+1}^{n} l_{ki}x_i \right), k = n, \dots, 1.$$

The above gives basic algorithm for computing solution to linear system in terms of lower/upper triangular coefficient matrices. Now we format the above as following algorithm.

Algorithm 1. L/U solve

Input $L(U), b$
for $k = 1, \dots, n$ ($k = n, \dots, 1$ for U)

$x_k = \frac{1}{l_{kk}}(b_k - \sum_{i=1}^{k-1} l_{ki}x_i)$
or $x_k = \frac{1}{u_{kk}}(b_k - \sum_{i=k+1}^{n} l_{ki}x_i)$

end for
Output x

LU Decomposition (with pivot)

Now we consider, more generally, a linear system

$$Ax = b,$$

where $A \in \mathbb{R}^{n\times n}$ is full-rank.

By theory of Gauss-Elimination, we have

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \overset{\text{Gauss}}{\Longrightarrow} \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22}-\frac{a_{21}}{a_{11}}a_{12} & \cdots & a_{2n}-\frac{a_{21}}{a_{11}}a_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}-\frac{a_{n1}}{a_{11}}a_{12} & \dots & a_{nn}-\frac{a_{21}}{a_{11}}a_{1n} \end{pmatrix}$$

and process of elimination can be viewed as imposing some row transformation L_1 to the original matrix

$$L_1 A = \begin{pmatrix} 1 & & & \\ -\frac{a_{21}}{a_{11}} & 1 & & \\ \vdots & \vdots & \ddots & \\ -\frac{a_{n1}}{a_{11}} & 0 & \dots & 1 \end{pmatrix} \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} = A_1.$$

Repeat the above procedures and assume that we never encounter 0 in the diagonal, then we have

$$(L_{n-1}\cdots L_2 L_1)A = U$$

and

$$A = (L_{n-1}\cdots L_2 L_1)^{-1}U = LU.$$

Here we recall that

$$L_1 = \begin{pmatrix} 1 & & & \\ -\frac{a_{21}}{a_{11}} & 1 & & \\ \vdots & \vdots & \ddots & \\ -\frac{a_{n1}}{a_{11}} & 0 & \dots & 1 \end{pmatrix} \qquad L_1^{-1} = \begin{pmatrix} 1 & & & \\ \frac{a_{21}}{a_{11}} & 1 & & \\ \vdots & \vdots & \ddots & \\ \frac{a_{n1}}{a_{11}} & 0 & \dots & 1 \end{pmatrix}$$

and that

$$L_1^{-1}L_2^{-1} = L_1^{-1} = \begin{pmatrix} 1 & & & \\ \frac{a_{21}}{a_{11}} & 1 & & \\ \vdots & \vdots & \ddots & \\ \frac{a_{n1}}{a_{11}} & 0 & \dots & 1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ 0 & 1 & & \\ \vdots & \vdots & \ddots & \\ 0 & \frac{\hat{a}_{n2}}{\hat{a}_{22}} & \dots & 1 \end{pmatrix} = \begin{pmatrix} 1 & & & \\ \frac{a_{21}}{a_{11}} & 1 & & \\ \vdots & \vdots & \ddots & \\ \frac{a_{n1}}{a_{11}} & \frac{\hat{a}_{n2}}{\hat{a}_{22}} & \dots & 1 \end{pmatrix}.$$

Therefore we can compute

$$L = L_1^{-1}L_2^{-1}\cdots L_{n-1}^{-1} = \begin{pmatrix} 1 & & & \\ \frac{a_{21}}{a_{11}} & 1 & & \\ \vdots & \vdots & \ddots & \\ \frac{a_{n1}}{a_{11}} & \frac{\hat{a}_{n2}}{\hat{a}_{22}} & \dots & 1 \end{pmatrix}.$$

Summarize the above and we get LU decomposition without pivoting.

Algorithm 2. LU decomposition (without pivoting)

Input A
Initialize $\hat{A} = A, L = I$
for $k = 1, \dots, n-1$

$L_{[k+1:n,k]} = \frac{\hat{A}_{[k+1:n,k]}}{\hat{A}_{[k,k]}}$

$\hat{A} = L_k \hat{A}$

end for
Output $L, U = \hat{A}$

Recally that we require that no element encountered on the diagonal is 0 and this is often not the case. Hence we consider LU decomposition with **column pivoting**. i.e., if we encounter 0 on the diagonal, we implement row permutation P to move element of largest scale in the current column to diagonal

$$\begin{pmatrix} a_{11}=0 & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \overset{\text{Permutation}}{\Longrightarrow} \begin{pmatrix} \hat{a}_{21} & a_{12} & \cdots & a_{1n} \\ 0 & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix}.$$

The above procedure gives, sequentially a series of row transformations as well as permutations to give

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

and if we let $P=P_{n-1}P_{n-2}\cdots P_1$ and we have

$$PA=(P_{n-1}P_{n-2}\cdots P_1)(L_{n-1}P_{n-1}\cdots L_2P_2L_1P_1)^{-1}U=LU,$$

where $L=(P_{n-1}P_{n-2}\cdots P_1)(L_{n-1}P_{n-1}\cdots L_2P_2L_1P_1)^{-1}$ is a lower-triangular matrix. More detailedly, we can write

$$\begin{aligned} & (P_{n-1}P_{n-2}\cdots P_1)(L_{n-1}P_{n-1}\cdots L_2P_2L_1P_1)^{-1} \\ = \; & (P_{n-1}P_{n-2}\cdots P_2)(L_1^{-1}P_2^{-1}L_2^{-1}\cdots P_{n-1}^{-1}L_{n-1}^{-1}) \\ = \; & P_{n-1}P_{n-2}\cdots P_3[(P_2L_1^{-1}P_2^{-1})L_2^{-1}]P_3^{-1}\cdots P_{n-1}^{-1}L_{n-1}^{-1} \\ = \; & P_{n-1}P_{n-2}\cdots[(P_3\hat{L}_2P_3^{-1})L_3^{-1}]\cdots P_{n-1}^{-1}L_{n-1}^{-1} \\ \cdots & \\ = \; & P_{n-1}\hat{L}_{n-2}L_{n-1}^{-1} \end{aligned}$$

and every matrix is lower-triangular by induction.

Last we summarize above results in following algorithm.

Algorithm 3. LU decomposition (with column pivoting)

Input A
Initialize $\hat{A}=A, L=I$
for $k=1,\dots,n-1$

$L_{[k+1:n,k]}=\frac{\hat{A}_{[k+1:n,k]}}{\hat{A}_{[k,k]}}$

Choose $P_k=I_{k,p}, p=\arg\max_{p\geq k}|\hat{a}_{p,k}|$

$\hat{A}=L_kP_k\hat{A}$

end for
Output $P=\prod_k P_{n-k-1}, L, U=\hat{A}$

Cholesky Decomposition

Now we consider a special case where the coefficient matrix A is symmetric and positive definite. i.e., $A \succeq 0$ and by Cholesky decomposition theorem, we have that there exists a lower-triangular matrix L such that

$$A = LL^\top = \begin{pmatrix} l_{11} & & & \\ l_{21} & l_{22} & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{pmatrix} \begin{pmatrix} l_{11} & l_{21} & \cdots & l_{n1} \\ & l_{22} & \cdots & l_{n2} \\ & & \ddots & \vdots \\ & & & l_{nn} \end{pmatrix}$$

It follows from formula of matrix multiplication that

$$a_{ij} = \sum_{k=1}^{n} l_{ik} l_{kj} = \sum_{k \le \min\{i,j\}}^{n} l_{ik} l_{kj}$$

and we can recursively get

$$l_{11}^2 = a_{11}, l_{11} l_{21} = a_{21}, l_{21}^2 + l_{22}^2 = a_{22} \dots$$

To sum up, we give the following algorithm on Cholesky decomposition.

Algorithm 4. Cholesky decomposition

Input A
Initialize $L = \mathbf{0}$
for $k = 1, \dots, n$
  for $m = 1, \dots, k-1$
    $l_{km} = \frac{1}{l_{mm}} (a_{km} - L_{[k,1:m-1]} \cdot L_{[m,1:m-1]})$
  end for
  $l_{kk} = \sqrt{a_{kk} - \|L_{[k,1:k-1]}\|_2^2}$
end for
Output L

LDL Decomposition

Recall that in Cholesky decomposition we need to take squareroot of diagonal elements and this is not desirable for large-scale computations. Hence we can preserve the diagonal elements and alternatively decompose $A \succeq 0$ into

$$A = LDL^\top = \begin{pmatrix} 1 & & & \\ l_{21} & 1 & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & l_{n2} & \cdots & 1 \end{pmatrix} \begin{pmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{pmatrix} \begin{pmatrix} 1 & l_{21} & \cdots & l_{n1} \\ & 1 & \cdots & l_{n2} \\ & & \ddots & \vdots \\ & & & 1 \end{pmatrix}$$

where $D = \mathrm{diag}(d)$ is a diagonal matrix and $\mathrm{diag}(L) = \mathbf{1}$.

LDL decomposition is implemented exactly the same as Cholesky decomposition as we can write

$$a_{ij} = \sum_{k \le \min\{i,j\}}^{n} d_k l_{ik} l_{kj}.$$

This gives the algorithm below.

Algorithm 5. LDL decomposition

Input A
Initialize $L = I, d = \mathbf{0}$
for $k = 1, \ldots, n$
    for $m = 1, \ldots, k-1$
        $l_{km} = \frac{1}{d_m}(a_{km} - d_{[1:m-1]} \circ L_{[k,1:m-1]} \cdot L_{[m,1:m-1]})$
    end for
    $d_k = a_{kk} - \|d_{[1:k-1]} \circ L_{[k,1:k-1]}\|_2^2$
end for
Output L, d