LU Factorization

Facts From Linear Algebra:

1. \((\text{row vec})[\text{matrix}] = \text{weighting of rows of matrix, using entries in row vec as weights}\)

\[
\begin{pmatrix}
1 & 2 & 3
\end{pmatrix} \begin{pmatrix}
4 & 5 & 6 & 7 \\
8 & 9 & 10 & 11 \\
12 & 13 & 14 & 15
\end{pmatrix}
= 1 \begin{pmatrix}
4 & 5 & 6 & 7
\end{pmatrix}
+ 2 \begin{pmatrix}
8 & 9 & 10 & 11
\end{pmatrix}
+ 3 \begin{pmatrix}
12 & 13 & 14 & 15
\end{pmatrix}
= \begin{pmatrix}
32 & 56 & 68 & 74
\end{pmatrix}
\]

2. \([\text{matrix}] (\text{col vec}) = \text{weighting of cols of matrix, using entries in col vec as weights}\)

\[
\begin{pmatrix}
-1 & 0 & 4 & 2 \\
3 & 2 & 0 & 1 \\
4 & 1 & -3 & 2 \\
1 & -1
\end{pmatrix} \begin{pmatrix}
1 \\
0 \\
3 \\
-1
\end{pmatrix}
= 1 \begin{pmatrix}
1
\end{pmatrix}
+ 0 \begin{pmatrix}
0
\end{pmatrix}
+ 3 \begin{pmatrix}
4
\end{pmatrix}
- 1 \begin{pmatrix}
2
\end{pmatrix}
= \begin{pmatrix}
9
\end{pmatrix}
\]

3. \([\text{matrix } A][\text{matrix } B] = [\text{row } 1 \text{ of } A][\text{matrix } B] = [\text{row } n \text{ of } A][\text{matrix } B]\)

\[
[\text{matrix } A] \begin{pmatrix}
a_1 & b_1 \\
a_2 & b_2 \\
a_3 & b_3 \\
a_4 & b_4
\end{pmatrix}
= \begin{pmatrix}
a_1 [A] & b_1 \[A] \\
a_2 [A] & b_2 \[A] \\
a_3 [A] & b_3 \[A] \\
a_4 [A] & b_4 \[A]
\end{pmatrix}
\]

4. An elementary matrix is a matrix formed by performing one elementary row or column operation on the identity matrix. Left-multiplication of a matrix by an elementary matrix is equivalent to performing the same row op on the matrix that was performed on the identity to produce that elementary matrix.

(i) Scaling of a row \( \begin{pmatrix}
\frac{1}{2} & 1 \\
1 & 1
\end{pmatrix} \begin{pmatrix}
1 & 2 & 3 \\
4 & 5 & 6
\end{pmatrix}
= \begin{pmatrix}
\frac{1}{2} & 1 & 3/2 \\
4 & 5 & 6
\end{pmatrix}
\) by \( \frac{1}{2} \)

(ii) Adding a multiple of a row to another row \( \begin{pmatrix}
1 & 2 \\
2 & 1
\end{pmatrix} \begin{pmatrix}
4 & 2 & 3 \\
7 & 8 & 9
\end{pmatrix}
= \begin{pmatrix}
4 & 5 & 6 \\
15 & 18 & 21
\end{pmatrix}
\) to row 3

(iii) Permutation of rows \( \begin{pmatrix}
1 & 0 \\
0 & 1
\end{pmatrix} \begin{pmatrix}
4 & 2 & 3 \\
7 & 8 & 9
\end{pmatrix}
= \begin{pmatrix}
4 & 2 & 3 \\
7 & 8 & 9
\end{pmatrix}
\) rows 2&3 were swapped
5. Same thing for right multiplication by elementary matrices:

\[
\begin{pmatrix}
\frac{2}{3} & 2 \\
\frac{4}{3} & \frac{5}{3}
\end{pmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} \frac{4}{3} \\ \frac{8}{3} \end{pmatrix}
\]

\[
\begin{pmatrix}
\frac{3}{2} & 2 \\
\frac{5}{2} & 1
\end{pmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} \frac{3}{2} \\ \frac{4}{2} \end{pmatrix}
\]

\[
\begin{pmatrix}
\frac{2}{3} & 2 \\
\frac{4}{3} & \frac{5}{3}
\end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{3}{2} \\ \frac{6}{3} \end{pmatrix}
\]

6. Inverse of elementary matrices are easy:

\[
\begin{pmatrix}
\frac{2}{3} \\ -1
\end{pmatrix}^{-1} = \begin{pmatrix} -\frac{3}{2} \\ \frac{3}{2}
\end{pmatrix}, \quad \begin{pmatrix}
\frac{1}{2} \\ -1
\end{pmatrix}^{-1} = \begin{pmatrix} 2 \\ -2
\end{pmatrix}, \quad \begin{pmatrix}
1 \\ 0
\end{pmatrix}^{-1} = \begin{pmatrix} 1 \\ 0
\end{pmatrix}
\]

7. Type (ii) elementary matrices "stack":

\[
\begin{pmatrix}
\frac{2}{3} \\ -1
\end{pmatrix} \begin{pmatrix} 1 \\ 2
\end{pmatrix} \begin{pmatrix} 1 \\ -1
\end{pmatrix} = \begin{pmatrix} \frac{12}{3} \\ \frac{9}{3}
\end{pmatrix}
\]

Note:

\[
\begin{pmatrix}
\frac{2}{3} \\ -1
\end{pmatrix}^{-1} = \begin{pmatrix} \frac{3}{2} \\ -\frac{3}{2}
\end{pmatrix}, \quad \begin{pmatrix} 1 \\ 2
\end{pmatrix}^{-1} = \begin{pmatrix} 1 \\ -2
\end{pmatrix}, \quad \begin{pmatrix} 1 \\ 0
\end{pmatrix}^{-1} = \begin{pmatrix} 1 \\ 0
\end{pmatrix}
\]

8. Finally, (identity matrices with one lower-triangular column) stack:

When multiplied with leftmost columns appearing as leftmost factors:

\[
\begin{pmatrix}
1 \\ \frac{1}{2}
\end{pmatrix} \begin{pmatrix} 1 \\ \frac{6}{2}
\end{pmatrix} \begin{pmatrix} 1 \\ \frac{4}{2}
\end{pmatrix} = \begin{pmatrix} 1 \\ \frac{1}{2}
\end{pmatrix}
\]

---

**General Strategy for LU Factorization:**

Let A be an invertible, real (or complex)-valued matrix with principal leading minors all nonzero (i.e., all submatrices of A of size k x k with the same upper left corner a_{11} are invertible). e.g. \( A = \begin{pmatrix} 1 & -1 & 3 \\ 2 & 10 & 7 \\ 4 & -5 & 100 \end{pmatrix} \), \( A_{11} = (1), A_{22} = (2 -10), \) and \( A_{33} = A \) are all invertible.

\[
A = \begin{bmatrix} \text{mess} \end{bmatrix}, \quad L_1 A = \begin{bmatrix} \vdots & \text{mess} \end{bmatrix}, \quad L_2 L_1 A = \begin{bmatrix} \vdots & \vdots & \text{mess} \end{bmatrix}, \quad \ldots, \quad L_n \cdots L_2 L_1 A = \begin{bmatrix} \vdots & \vdots & \vdots & \text{mess} \end{bmatrix} = U
\]

Peeling off \( L_i \)'s from the left: \( A = L_n^{-1} \cdots L_2^{-1} L_1^{-1} U = LU \). Each \( L_i \) can be chosen to have all 1's on the diagonal, so \( L \) will also have 1's on its diagonal (\( L \) is unit lower triangular). \( U \) will be an invertible upper triangular matrix.
By the facts, \( L_1 \cdots L_i \) is very easy to compute! Also, each \( L_i \) is obtained Gaussian-elimination style by using the \((j,j)\)th component of \( L_{j-1} \cdots L_1 A \) to kill off all other entries beneath it. The only potential problem with this procedure is that \((j,j)\)th component is 0. But this can never happen:

Suppose the \((j,j)\)th component of \( L_{j-1} \cdots L_1 A \) is 0. Then

\[
L_{j-1} \cdots L_1 A = \begin{bmatrix}
0 \\
0 \\
\vdots \\
0 \\
0
\end{bmatrix}
\]

This implies \( A \mathbf{x} = L_{j-1} \cdots L_1 \mathbf{x} = \begin{bmatrix}0 \\ 0 \\ \vdots \\ 0 \\ 0\end{bmatrix} \). In particular,

\[
\mathbf{x} = \begin{bmatrix} x_j \\ \vdots \\ x_{j-1} \\ x_{j+1} \\ \vdots \\ x_n \end{bmatrix}
\]

But \( A \mathbf{x} = (I \mathbf{0}) A \mathbf{x} = (I \mathbf{0}) \mathbf{0} = \mathbf{0} \), and so \( A \mathbf{x} = (I \mathbf{0}) \mathbf{0} = \mathbf{0} \). Conversely, that \( A \mathbf{x} = \mathbf{0} \) is invertible. Thus an LU factorization for this type of matrix always exists. Furthermore, since

\[
A_{j-1} \cdots A_1 = (I \mathbf{0}) A (I \mathbf{0}) = (L_{j-1} \cdots L_1 A) (I \mathbf{0}) = (L_{j-1} \cdots L_1 A) (I \mathbf{0}) = (L_{j-1} \cdots L_1 A) = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} A_{j-1} & 0 \\ 0 & 0 \end{bmatrix}
\]

we can say that \( A \)’s principal leading minors all nonzero \( \iff \) (an LU factorization of \( A \) exists, with \( L \) unit lower triangular and \( U \) invertible at.

We get uniqueness of the LU decomposition (with \( L \) unit) very easily:

writing \( L = \begin{bmatrix} I \\ L_{n-1}\end{bmatrix} \), \( U = \begin{bmatrix} U_n \\ \vdots \\ U_1 \end{bmatrix} \) and solving for the entries gives:

\[
\begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} = A = \begin{bmatrix} U_n \\ \vdots \\ U_1 \end{bmatrix} = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} U_{n1} = a_{n1} \\ U_{n2} = a_{n2} \\ \vdots \\ U_{nn} = a_{nn} \end{bmatrix}
\]

This algorithm (running in \( \Theta(n^3) \) time) uniquely specifies all entries, and so the LU decomposition of any matrix \( A \) with all principal leading minors nonzero, and \( L \) forced to be unit, is unique.

We can make \( U \) unit as well:

\[
U = \begin{bmatrix} U_{11} & \cdots & U_{1n} \\ \vdots & \ddots & \vdots \\ U_{n1} & \cdots & U_{nn} \end{bmatrix} = \begin{bmatrix} D_1 & \cdots & D_{n1} \\ \vdots & \ddots & \vdots \\ U_{n1} & \cdots & U_{nn} \end{bmatrix} = D \tilde{U}
\]

where \( D \) is (invertible) diagonal and \( U \) is unit \( u \times u \).

The decomposition would look like \( A = L U = L D \tilde{U} \) in this case.

What’s the purpose of doing LU factorizations?

To solve \( A \mathbf{x} = \mathbf{b} \), we may solve \( L \mathbf{w} = \mathbf{b} \) by subbing in \( \mathbf{w} = U \mathbf{x} \), back-solving \( L \mathbf{w} = \mathbf{b} \) for \( \mathbf{w} \), and then back-solving \( U \mathbf{x} = \mathbf{w} \) for \( \mathbf{x} \). Each back solve takes \( \frac{n^2}{2} \) time, so solving \( A \mathbf{x} = \mathbf{b} \) in this manner runs in \( \Theta(n^3) \) time. Obtaining \( L \) & \( U \) in the first place takes \( \Theta(n^3) \) time, same as Gaussian elimination. Just doing Gaussian elimination on \( A \mathbf{x} = \mathbf{b} \) takes \( \Theta(n^3) \) time (slower than solving \( L \mathbf{w} = \mathbf{b} \)) and even though computing \( A^{-1} \) takes \( \Theta(n^3) \) time as well using \( A^{-1} \) is often less numerically stable than computing \( L \) & \( U \) to solve \( A \mathbf{x} = \mathbf{b} \). See math 270b for details!
**Example**

Let’s hope that all of A’s principal leading minors are nonzero.

\[
A = \begin{pmatrix} 5 & 6 & 7 \\ 10 & 23 & 5 \\ 5 & 0 & 67 \end{pmatrix} \quad \text{and} \quad L_2L_1A = U.
\]

If we do row into the problem \((L_{j1}...L_{1j})A = 0\), we can use pivoting (as long as \(A\) is invertible):

\[
A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 7 \\ 2 & 0 & 3 \end{pmatrix} = L_1A.
\]

We get \(L_2L_1AP_1 = U\), so

\[
A = L_1^T L_2^T U P_1^T = \begin{pmatrix} 2 & 1 \\ 4 & 0 \\ 1 & 3 \end{pmatrix} \begin{pmatrix} 3 & 2 \\ 4 & 0 \\ 1 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 3 & 2 \\ 4 & 2 & 1 \\ 4 & 0 & 1 \end{pmatrix}.
\]


**LDLT Factorization**

A real, symmetric matrix is positive definite if \(\forall x \in \mathbb{R}^n, x^T A x \geq 0\), and \(x^T A x = 0\) iff \(x = 0\). Let \(A\) be a real symmetric positive definite matrix. Suppose \(A_k = (I_k \ 0) A (I_k \ 0)^T\) is not invertible. Then

\[3 \ x_1, ..., x_k \in \mathbb{R},\] not all zero, st.

\[0 = A_k x_k = (I_k \ 0) A (I_k \ 0)^T x_k = (I_k \ 0) A x_k = x_k^T A x_k\]

Thus all of \(A\)'s principal leading minors are nonzero, so \(A\) admits a unique LDLT factorization with \(L = L_1 ... L_n\) and \(U\) unique up to signs. But we may take it a step further, using the strategy:

\[
A = \begin{bmatrix} \text{symmetric} \\\n\end{bmatrix}, \quad L_1 A L_1^T = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix}, \quad L_2 A L_2^T = \begin{bmatrix} 1 & 0 \\ \vdots & \ddots \end{bmatrix}, \quad ..., \quad L_n A L_n^T = \begin{bmatrix} 1 & 0 \\ 0 & \ddots \end{bmatrix} = D.
\]

Thus, \(A = L_1^T ... L_n^T D L_n ... L_1 = (L_1^T ... L_n^T) D (L_1 ... L_n) = LDL^T\).

The key idea here is since \(L_1^T A L_1^T\) is symmetric, the same row ops that use the \((j+1)\text{st}\) entry to kill off all entries below it can be used to kill off all entries to the right of it by treating them as column ops instead.

If \((L_1^T ... L_n^T) = L\) for some \(j\), then using \(e_j^T (L_m^T ... L_1^T) e_j = (L_m^T ... L_1^T) A (L_m^T ... L_1^T)^T e_j = 0\) gives \(0 = 0\), contradicting that \(A\) is positive definite. So this process always works.

Also, \(\forall j \in \{1, ..., n\}\)

\[0 < (L_1^T e_j)^T A (L_1^T e_j) = (L_1^T e_j)^T D e_j = D_{jj},\]

so all of \(D\)'s diagonal entries are positive.
Setting \( U = DL^T \) and picking \( L_i \)'s so that diagonals of all \( L_i \)'s are zeros, we get that

\[ A = LDL^T = LU \] uniquely determines the unit lower triangular matrix \( L \) and the invertible upper triangular matrix \( U \).

So \( D = U L^T \) is also uniquely determined. This gives the uniqueness of the \( LDL^T \) decomposition of a real symmetric positive definite matrix.

Since \( D \)'s diagonal entries are positive, we may find a square root of \( D \), \( \sqrt{D} \), i.e. with \( \sqrt{D}^2 = D \). Then \( A = LDL^T = L \sqrt{D} \sqrt{D} L^T = L \sqrt{D} \sqrt{D} L^T = L \sqrt{D} \sqrt{D} L^T \), where \( L \) is invertible lower triangular. There are exactly \( 2^n \) many such decompositions \( A = LL^T \), since we have a choice whether to make each individual diagonal entry of \( \sqrt{D} \) positive or negative.

**Example** (note: after killing off entries beneath the \((ij)\)'th entry, the analogous column operations will kill off all entries to the right of that entry, and will leave all other entries unchanged, since the column 'doing the killing' only has one nonzero term in it only).

\[
A = \begin{pmatrix}
1 & -1 & 4 \\
-1 & 5 & -4 \\
4 & -4 & 25
\end{pmatrix} \rightarrow \begin{pmatrix}
1 & 0 & 0 \\
0 & 4 & 0 \\
0 & 0 & 9
\end{pmatrix} =: D = L_1 A L_1^T \text{, so } A = L_1^T D L_1^T = \begin{pmatrix}
1 & 0 & 0 \\
0 & 4 & 0 \\
0 & 0 & 9
\end{pmatrix} \begin{pmatrix}
1 & 1 & 1 \\
4 & 1 & 1 \\
4 & 1 & 1
\end{pmatrix} =: L D L^T.
\]

Setting \( D = \begin{pmatrix}
1 & 2 \\
0 & 3
\end{pmatrix} \) gives us \( \sqrt{D}^2 = D \). \( A = \sqrt{D} \sqrt{D} L^T \).

\[
A = \begin{pmatrix}
1 & 1 & 1 \\
-1 & 4 & 1 \\
3 & 2 & 3
\end{pmatrix} \begin{pmatrix}
1 & 1 & 1 \\
1 & 4 & 1 \\
1 & 2 & 3
\end{pmatrix} =: L \sqrt{D} \sqrt{D} L^T.
\]

**Note:** if \( A \) is real symmetric, then \((A \text{ positive definite}) \iff (\det(A_{kk}) > 0) \forall k \in \{1, \ldots, n\}\)

Not so easy to show this.

**Preliminaries for iterative procedures to solve** \( AX = b \):

Vector norms, and their induced matrix norm.

For \( \vec{x} \in \mathbb{R}^n \), \[ \| \vec{x} \|_\infty := \max_i |x_i|, \quad \| \vec{x} \|_1 := \sum_i |x_i|, \quad \| \vec{x} \|_2 := \left( \sum_i |x_i|^2 \right)^{1/2}. \]

These three norms all obey the usual properties of norms. They each give rise to an induced matrix norm as well, whose norm properties follow from those of the associated norm on \( \mathbb{R}^n \). Formally, \[ \| A \|_{\text{induced}} := \sup_{\vec{x} \neq \vec{0}} \frac{\| A \vec{x} \|}{\| \vec{x} \|}. \]

We will assume \( A \) is square here, but in general it need not be.
\[ \| A x \|_\infty = \max |a_{ij}| = \max i \{ \max j (\sum |a_{ij}|) \} = \| x \|_\infty \cdot \max j (\sum |a_{ij}|) \leq \max j (\max i |a_{ij}|) \leq \| x \|_1 . \]

Setting \( x_j = \frac{a_{ij}}{\max j (\sum |a_{ij}|)} \), so choosing \( i \) st \( \sum |a_{ij}| = \max j (\sum |a_{ij}|) \) gives \( \| A x \|_\infty \geq \max j (\sum |a_{ij}|) \). Thus \( \| A \|_\infty = \max j (\sum |a_{ij}|) = \max |a_{ij}| \) as sum.

\[ \| A x \|_1 = \sum_j |\sum_i a_{ij} x_j| \leq \sum_j (\sum_i |a_{ij}|) \sum_j x_j = \sum_j (\sum_i a_{ij} x_j) = \sum_j (\sum_j x_j) \sum_i |a_{ij}| = \sum_j (\sum_j x_j) \max i (\sum |a_{ij}|) = \| x \|_1 \cdot \max i (\sum |a_{ij}|) \]

and the quality is realized by \( x = e_k \), where \( k \in \{1, \ldots, n\} \) is such that \( \sum |a_{ik}| = \max i (\sum |a_{ij}|) \). Thus, \( \| A \|_1 = \max j (\sum |a_{ij}|) = \max |a_{ij}| \) as sum.

\[ \| A x \|_2 = (\sum |a_{ij}^2 x_j|^2)^{1/2} = (\sum |(\sum_j a_{ij} x_j)^2|^2)^{1/2} \leq \left( \sum |(\sum_j a_{ij}^2 (x_j^2))^2|^2 \right)^{1/2} = \| x \|_2 \cdot \left( \sum_j |a_{ij}|^2 x_j^2 \right)^{1/2} \]

Hölder, p=2 (a.k.a. Cauchy-Schwarz)

But the inequality is not tight! Another method is needed.

\[ \| A x \|_2 = (A^T A x, x) = (A x, A x)^T = x^T U D U^T x = (U^T x)^T D (U^T x) \leq \left( \max i (\sum |a_{ij}|) \right) \| U^T x \|_2^2 \]

where \( \sigma(A^T A) \) is the largest singular value of \( A^T A \). Equality is realized by \( x = (\text{corresponding eigenvector}) \).

Thus \( \| A \|_2 = \sigma(A) = \sigma(A^T A) \), the largest singular values of \( A \).

---

Gauss-Jacobi: Trying to solve \( A x = b \), but don't want to spend \( O(n^3) \) time inverting, doing Gaussian elimination, or getting an \( LU \) factorization.

Strategy: write \( A = D + N \), where \( D_{ij} = \{a_{ij} \text{ if } i=j \text{ and } N = \{0 \text{ if } i \neq j \} \}

(\bar{A} x = \bar{b}) \rightarrow (D+N) x = \bar{b} \rightarrow (D x = \bar{b} - Nx)

Set \( x_0 = \overline{b} \). Use \( D x_{k+1} = \bar{b} - Nx_k \) to generate a sequence of solution iterates \( (x_k) \).

Error: Suppose \( D \) is invertible (i.e., has nonzero diagonal entries) and \( x \) solves \( A x = b \).

The algorithm described above ("Gauss-Jacobi") determines the \( x_k \)'s uniquely, and

\[ \| x_k - x \| = \| D^{-1} (b - N x_k) - D^{-1} (b - N x) \| = \| D^{-1} N (x_k - x) \| \leq \sigma(N) \| x_k - x \| \].

This works for any norm \( \| \cdot \| \) on \( \mathbb{R}^n \) though all norms on finite-dimensional vector spaces are equivalent (see math 245 b). In particular, if \( \| \cdot \| = \| \cdot \|_\infty \) and \( A \) is (strictly) diagonally dominant \( (|a_{ii}| > \sum_{j \neq i} |a_{ij}|) \), then

\[ \| D^{-1} N \|_{\infty} = \max j (|a_{jj}|) = 1 \Rightarrow \| x_k - x \| = \| D^{-1} N (x_k - x) \| \leq |a_{ii}| \leq 1, \]

so \( \| x_k - x \| \leq \| D^{-1} N \|_{\infty} \| x_k - x \| \).

Thus we get at least linear convergence, \( \lambda = \| D^{-1} N \|_{\infty} \in [0,1) \) if \( \lambda < 1 \).

Each iterate \( x_k \) costs \( O(n^2) \) time to compute (because of \( N x_k \)), so each iteration is vastly cheaper than solving \( Ax = b \) exactly (\( O(n^3) \) vs. \( O(n^2) \)).

Another common "splitting" is \( A = M + N \), where \( M \) is lower triangular invertible and \( N \) is upper triangular with 0's on diagonal, leading to the "Gauss-Seidel algorithm".