LU decomposition
From Wikipedia, the free encyclopedia

In linear algebra, **LU decomposition** (also called **LU factorization**) is a matrix decomposition which writes a matrix as the product of a lower triangular matrix and an upper triangular matrix. The product sometimes includes a permutation matrix as well. This decomposition is used in numerical analysis to solve systems of linear equations or calculate the determinant of a matrix. LU decomposition can be viewed as a matrix form of Gaussian elimination. LU decomposition was introduced by mathematician Alan Turing [1]

**Contents**
- 1 Definitions
- 2 Existence and uniqueness
- 3 Positive definite matrices
- 4 Explicit formulation
- 5 Algorithms
  - 5.1 Doolittle algorithm
  - 5.2 Crout and LUP algorithms
  - 5.3 Theoretical complexity
- 6 Small example
- 7 Sparse matrix decomposition
- 8 Applications
  - 8.1 Solving linear equations
  - 8.2 Inverse matrix
  - 8.3 Determinant
- 9 See also
- 10 References
- 11 External links

**Definitions**

Let $A$ be a square matrix. An **LU decomposition** is a decomposition of the form

$$A = LU,$$

where $L$ and $U$ are lower and upper triangular matrices (of the same size), respectively. This means that $L$ has only zeros above the diagonal and $U$ has only zeros below the diagonal. For a $3 \times 3$ matrix, this becomes:

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}.$$

An **LDU decomposition** is a decomposition of the form

$$A = LDU,$$

where $D$ is a diagonal matrix and $L$ and $U$ are *unit* triangular matrices, meaning that all the entries on the diagonals of $L$ and $U$
are one.

An **LUP decomposition** (also called a **LU decomposition with partial pivoting**) is a decomposition of the form

$$PA = LU,$$

where $L$ and $U$ are again lower and upper triangular matrices and $P$ is a permutation matrix, i.e., a matrix of zeros and ones that has exactly one entry 1 in each row and column.

An **LU decomposition with full pivoting** (Trefethen and Bau) takes the form

$$PAQ = LU,$$

Above we required that $A$ be a square matrix, but these decompositions can all be generalized to rectangular matrices as well. In that case, $L$ and $P$ are square matrices which each have the same number of rows as $A$, while $U$ is exactly the same shape as $A$. **Upper triangular** should be interpreted as having only zero entries below the main diagonal, which starts at the upper left corner.

### Existence and uniqueness

An invertible matrix admits an **LU** factorization if and only if all its leading principal minors are non-zero. The factorization is unique if we require that the diagonal of $L$ (or $U$) consist of ones. The matrix has a unique **LDU** factorization under the same conditions.

If the matrix is singular, then an **LU** factorization may still exist. In fact, a square matrix of rank $k$ has an **LU** factorization if the first $k$ leading principal minors are non-zero, although the converse is not true.

The exact necessary and sufficient conditions under which a not necessarily invertible matrix over any field has an **LU** factorization are known. The conditions are expressed in terms of the ranks of certain submatrices. The Gaussian elimination algorithm for obtaining **LU** decomposition has also been extended to this most general case (Okunev & Johnson 1997).

Every invertible matrix $A$ admits an **LUP** factorization.

### Positive definite matrices

If the matrix $A$ is Hermitian and positive definite, then we can arrange matters so that $U$ is the conjugate transpose of $L$. In this case, we have written $A$ as

$$A = LL^*.$$

This decomposition is called the Cholesky decomposition. The Cholesky decomposition always exists and is unique. Furthermore, computing the Cholesky decomposition is more efficient and numerically more stable than computing some other **LU** decompositions.

### Explicit formulation

When an **LDU** factorization exists and is unique there is a closed (explicit) formula for the elements of $L$, $D$, and $U$ in terms of ratios of determinants of certain submatrices of the original matrix $A$ (Householder 1975). In particular, $D_1 = A_{1,1}$ and for $i = 2, \ldots, n$, $D_i$ is the ratio of the $i^{th}$ principal submatrix to the $(i - 1)^{th}$ principal submatrix.
Algorithms

The LU decomposition is basically a modified form of Gaussian elimination. We transform the matrix $A$ into an upper triangular matrix $U$ by eliminating the entries below the main diagonal. The Doolittle algorithm does the elimination column by column starting from the left, by multiplying $A$ to the left with atomic lower triangular matrices. It results in a unit lower triangular matrix and an upper triangular matrix. The Crout algorithm is slightly different and constructs a lower triangular matrix and a unit upper triangular matrix.

Computing the LU decomposition using either of these algorithms requires $2n^3 / 3$ floating point operations, ignoring lower order terms. Partial pivoting adds only a quadratic term; this is not the case for full pivoting.[2]

Doolittle algorithm

Given an $N \times N$ matrix

$$A = (a_{n,n})$$

we define

$$A^{(0)} := A$$

and then we iterate $n = 1,...,N-1$ as follows.

We eliminate the matrix elements below the main diagonal in the $n$-th column of $A^{(n-1)}$ by adding to the $i$-th row of this matrix the $n$-th row multiplied by

$$L_{i,n} := -\frac{a_{i,n}^{(n-1)}}{a_{n,n}}$$

for $i = n + 1,\ldots,N$. This can be done by multiplying $A^{(n-1)}$ to the left with the lower triangular matrix

$$L_n = \begin{pmatrix} 1 & & & & \\ \vdots & \ddots & \ddots & & \\ & \ddots & & 1 & \\ 0 & \ldots & l_{N,n} & \ddots & \\ & & 0 & \ldots & 1 \end{pmatrix}.$$

We set

$$A^{(n)} := L_n A^{(n-1)}.$$ 

After $N-1$ steps, we eliminated all the matrix elements below the main diagonal, so we obtain an upper triangular matrix $A^{(N-1)}$. We find the decomposition

$$A = L_1^{-1} L_1 A^{(0)} = L_1^{-1} A^{(1)} = L_1^{-1} L_2^{-1} L_2 A^{(1)} = L_1^{-1} L_2^{-1} A^{(2)} = \ldots = L_1^{-1} \ldots L_{N-1}^{-1} A^{(N-1)}.$$ 

Denote the upper triangular matrix $A^{(N-1)}$ by $U$, and $L = L_1^{-1} \ldots L_{N-1}^{-1}$. Because the inverse of a lower triangular
matrix \( L_n \) is again a lower triangular matrix, and the multiplication of two lower triangular matrices is again a lower triangular matrix, it follows that \( L \) is a lower triangular matrix. Moreover, it can be seen that

\[
L = \begin{pmatrix}
1 & & & \\
-\ell_{2,1} & 1 & & \\
& \ddots & \ddots & \\
& & -\ell_{n+1,n} & 1 \\
-\ell_{N,1} & & \cdots & 1
\end{pmatrix}.
\]

We obtain \( A = LU \).

It is clear that in order for this algorithm to work, one needs to have \( a_{n,n}^{(n-1)} \neq 0 \) at each step (see the definition of \( l_{i,n} \)). If this assumption fails at some point, one needs to interchange \( n \)-th row with another row below it before continuing. This is why the LU decomposition in general looks like \( P^{-1}A = LU \).

**Crout and LUP algorithms**

The LUP decomposition algorithm by Cormen et al. generalizes Crout matrix decomposition. It can be described as follows.

1. If \( A \) has a nonzero entry in its first row, then take a permutation matrix \( P_1 \) such that \( AP_1 \) has a nonzero entry in its upper left corner. Otherwise, take for \( P_1 \) the identity matrix. Let \( A_1 = AP_1 \).
2. Let \( A_2 \) be the matrix that one gets from \( A_1 \) by deleting both the first row and the first column. Decompose \( A_2 = L_2 U_2 P_2 \) recursively. Make \( L \) from \( L_2 \) by first adding a zero row above and then adding the first column of \( A_1 \) at the left.
3. Make \( U_3 \) from \( U_2 \) by first adding a zero row above and a zero column at the left and then replacing the upper left entry (which is 0 at this point) by 1. Make \( P_3 \) from \( P_2 \) in a similar manner and define \( A_3 = A_1 / P_3 = AP_1 / P_3 \). Let \( P \) be the inverse of \( P_1 / P_3 \).
4. At this point, \( A_3 \) is the same as \( LU_3 \), except (possibly) at the first row. If the first row of \( A \) is zero, then \( A_3 = LU_3 \), since both have first row zero, and \( A = LU_3 P \) follows, as desired. Otherwise, \( A_3 \) and \( LU_3 \) have the same nonzero entry in the upper left corner, and \( A_3 = LU_3 U_1 \) for some upper triangular square matrix \( U_1 \) with ones on the diagonal (\( U_1 \) clears entries of \( LU_3 \) and adds entries of \( A_3 \) by way of the upper left corner). Now \( A = LU_3 U_1 P \) is a decomposition of the desired form.

**Theoretical complexity**

If two matrices of order \( n \) can be multiplied in time \( M(n) \), where \( M(n) \geq n^a \) for some \( a > 2 \), then the LU decomposition can be computed in time \( O(M(n)) \).\[3]\] This means, for example, that an \( O(n^{2.376}) \) algorithm exists based on the Coppersmith–Winograd algorithm.

**Small example**

\[
\begin{bmatrix}
4 & 3 \\
6 & 3
\end{bmatrix} = \begin{bmatrix}
\ell_{11} & 0 \\
\ell_{21} & \ell_{22}
\end{bmatrix} \begin{bmatrix}
u_{11} & u_{12} \\
0 & u_{22}
\end{bmatrix}.
\]
One way of finding the LU decomposition of this simple matrix would be to simply solve the linear equations by inspection. You know that:

\[
\begin{align*}
    l_{11} \cdot u_{11} + 0 \cdot 0 &= 4 \\
    l_{11} \cdot u_{12} + 0 \cdot u_{22} &= 3 \\
    l_{21} \cdot u_{11} + l_{22} \cdot 0 &= 6 \\
    l_{21} \cdot u_{12} + l_{22} \cdot u_{22} &= 3.
\end{align*}
\]

Such a system of equations is underdetermined. In this case any two non-zero elements of L and U matrices are parameters of the solution and can be set arbitrarily to any non-zero value. Therefore to find the unique LU decomposition, it is necessary to put some restriction on L and U matrices. For example, we can require the lower triangular matrix L to be a unit one (i.e. set all the entries of its main diagonal to ones). Then the system of equations has the following solution:

\[
\begin{align*}
    l_{21} &= 1.5 \\
    u_{11} &= 4 \\
    u_{12} &= 3 \\
    u_{22} &= -1.5.
\end{align*}
\]

Substituting these values into the LU decomposition above:

\[
\begin{bmatrix}
    4 & 3 \\
    6 & 3
\end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1.5 & 1 \end{bmatrix} \begin{bmatrix} 4 & 3 \\ 0 & -1.5 \end{bmatrix}.
\]

### Sparse matrix decomposition

Special algorithms have been developed for factorizing large sparse matrices. These algorithms attempt to find sparse factors L and U. Ideally, the cost of computation is determined by the number of nonzero entries, rather than by the size of the matrix.

These algorithms use the freedom to exchange rows and columns to minimize fill-in (entries which change from an initial zero to a non-zero value during the execution of an algorithm).

General treatment of orderings that minimize fill-in can be addressed using graph theory.

### Applications

#### Solving linear equations

Given a matrix equation

\[
    Ax = LUx = b
\]

we want to solve the equation for \(x\) given \(A\) and \(b\). In this case the solution is done in two logical steps:

1. First, we solve the equation \(Ly = b\) for \(y\)
2. Second, we solve the equation \(Ux = y\) for \(x\).

Note that in both cases we have triangular matrices (lower and upper) which can be solved directly using forward and backward substitution without using the Gaussian elimination process (however we need this process or equivalent to compute the \(LU\) decomposition itself). Thus the \(LU\) decomposition is computationally efficient only when we have to
solve a matrix equation multiple times for different $b$; it is faster in this case to do an LU decomposition of the matrix $A$ once and then solve the triangular matrices for the different $b$, than to use Gaussian elimination each time.

**Inverse matrix**

When solving systems of equations, $b$ is usually treated as a vector with a length equal to the height of matrix $A$. Instead of vector $b$, we have matrix $B$, where $B$ is an $n$-by-$p$ matrix, so that we are trying to find a matrix $X$ (also an $n$-by-$p$ matrix):

$$AX = LUX = B.$$

We can use the same algorithm presented earlier to solve for each column of matrix $X$. Now suppose that $B$ is the identity matrix of size $n$. It would follow that the result $X$ must be the inverse of $A$.[4]

**Determinant**

The matrices $L$ and $U$ can be used to compute the determinant of the matrix $A$ very quickly, because $\det(A) = \det(L) \cdot \det(U)$ and the determinant of a triangular matrix is simply the product of its diagonal entries. In particular, if $L$ is a unit triangular matrix, then

$$\det(A) = \det(L) \cdot \det(U) = 1 \cdot \prod_{i=1}^{n} u_{ii}.$$

The same approach can be used for $LUP$ decompositions. The determinant of the permutation matrix $P$ is $(-1)^S$, where $S$ is the number of row exchanges in the decomposition.

**See also**

- Block LU decomposition
- Cholesky decomposition
- Matrix decomposition
- QR decomposition
- LU Reduction

**References**


External links

References

- Module for LU Factorization with Pivoting (http://math.fullerton.edu/mathews/n2003/LUFactorMod.html), Prof. J. H. Mathews, California State University, Fullerton
- LU decomposition (http://numericalmethods.eng.usf.edu/topics/lu_decomposition.html) at Holistic Numerical Methods Institute

Computer code

- LAPACK (http://www.netlib.org/lapack/) is a collection of FORTRAN subroutines for solving dense linear algebra problems
- ALGLIB (http://www.alglib.net/) includes a partial port of the LAPACK to C++, C#, Delphi, etc.
- C++ code (http://www.johnloomis.org/ece538/notes/Matrix/ludcmp.html), Prof. J. Loomis, University of Dayton
- C code (http://mymathlib.webtrellis.net/matrices/linearsystems/doolittle.html), Mathematics Source Library
- Pseudo-code: Gaussian elimination approach (http://www.math.vt.edu/people/wapperom/class_home/4445/alg_luge.pdf), Prof. P. Wapperom, Virginia Tech
- Pseudo-code: Doolittle algorithm (http://www.math.vt.edu/people/wapperom/class_home/4445/alg_ludoolittle.pdf), Prof. P. Wapperom, Virginia Tech
- LU in X10 (http://docs.codehaus.org/display/XTENLANG/LU)

Online resources

- WebApp descriptively solving systems of linear equations with LU Decomposition (http://sole.ooz.ie/)
- Matrix Calculator (http://www.bluebit.gr/matrix-calculator/), bluebit.gr
- LU Decomposition Tool (http://www.uni-bonn.de/~manfear/matrix_lu.php), uni-bonn.de


Categories: Matrix decompositions Numerical linear algebra

- This page was last modified on 2 December 2011 at 19:26.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. See Terms of use for details.
  Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.