#linear_algebra

A factorization of a matrix A is an equation that expresse A as a producet of two or more matrices.Whereas matrix multiplication involves a synthesis of data(combining the effects of two or more linear transformations into a single matrix),matrix factorization is an *anlaysis* of data.In a language of computer science,the expression of A as a product amounts to a preprocessing of data in A,organizing that data into two or more parts whose structures are more useful in some way,perhaps more accessible for computation.

The LU Factorization

The LU Factorization

The LU factorization, described below, is motivated by the fairly common industrial and business problem of solving a sequence of equations, all with the same coefficient matrix:

$$A\mathbf{x} = \mathbf{b}_1, \quad A\mathbf{x} = \mathbf{b}_2, \quad \ldots, \quad A\mathbf{x} = \mathbf{b}_p \tag{1}$$

See Exercise 32, for example. Also see Section 5.8, where the inverse power method is used to estimate eigenvalues of a matrix by solving equations like those in sequence (1), one at a time.

When A is invertible, one could compute A^{-1} and then compute $A^{-1}\mathbf{b}_1$, $A^{-1}\mathbf{b}_2$, and so on. However, it is more efficient to solve the first equation in sequence (1) by row reduction and obtain an LU factorization of A at the same time. Thereafter, the remaining equations in sequence (1) are solved with the LU factorization.

At first, assume that A is an $m \times n$ matrix that can be row reduced to echelon form, *without row interchanges*. (Later, we will treat the general case.) Then A can be written in the form $A = LU$, where L is an $m \times m$ lower triangular matrix with 1's on the diagonal and U is an $m \times n$ echelon form of A. For instance, see Fig. 1. Such a factorization is called an **LU factorization** of A. The matrix L is invertible and is called a *unit* lower triangular matrix.

$$A = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ * & 1 & 0 & 0 \\ * & * & 1 & 0 \\ * & * & * & 1 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} \blacksquare & * & * & * & * \\ 0 & \blacksquare & * & * & * \\ 0 & 0 & 0 & \blacksquare & * \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{U}$$

FIGURE 1 An LU factorization.

Before studying how to construct L and U, we should look at why they are so useful. When $A = LU$, the equation $A\mathbf{x} = \mathbf{b}$ can be written as $L(U\mathbf{x}) = \mathbf{b}$. Writing $\mathbf{y}$ for $U\mathbf{x}$, we can find $\mathbf{x}$ by solving the *pair* of equations

$$\boxed{\begin{aligned} L\mathbf{y} &= \mathbf{b} \\ U\mathbf{x} &= \mathbf{y} \end{aligned}} \tag{2}$$

First solve $L\mathbf{y} = \mathbf{b}$ for $\mathbf{y}$, and then solve $U\mathbf{x} = \mathbf{y}$ for $\mathbf{x}$. See Fig. 2. Each equation is easy to solve because L and U are triangular.

EXAMPLE 1 It can be verified that

$$A = \begin{bmatrix} 3 & -7 & -2 & 2 \\ -3 & 5 & 1 & 0 \\ 6 & -4 & 0 & -5 \\ -9 & 5 & -5 & 12 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 2 & -5 & 1 & 0 \\ -3 & 8 & 3 & 1 \end{bmatrix} \begin{bmatrix} 3 & -7 & -2 & 2 \\ 0 & -2 & -1 & 2 \\ 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix} = LU$$

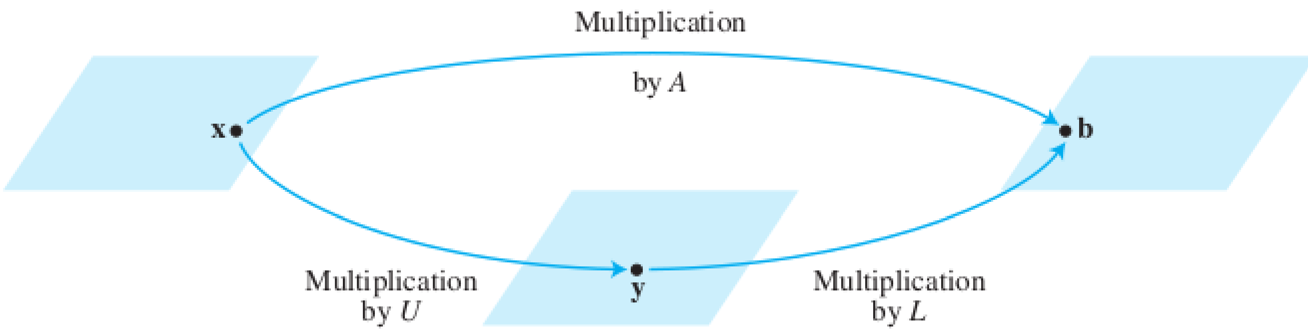

FIGURE 2 Factorization of the mapping $\mathbf{x} \mapsto A\mathbf{x}$.

NUMERICAL NOTES

The following operation counts apply to an $n \times n$ dense matrix A (with most entries nonzero) for n moderately large, say, $n \geq 30$.[1]

1. Computing an LU factorization of A takes about $2n^3/3$ flops (about the same as row reducing $[\,A \quad \mathbf{b}\,]$), whereas finding A^{-1} requires about $2n^3$ flops.
2. Solving $L\mathbf{y} = \mathbf{b}$ and $U\mathbf{x} = \mathbf{y}$ requires about $2n^2$ flops, because any $n \times n$ triangular system can be solved in about n^2 flops.
3. Multiplication of $\mathbf{b}$ by A^{-1} also requires about $2n^2$ flops, but the result may not be as accurate as that obtained from L and U (because of roundoff error when computing both A^{-1} and $A^{-1}\mathbf{b}$).
4. If A is sparse (with mostly zero entries), then L and U may be sparse, too, whereas A^{-1} is likely to be dense. In this case, a solution of $A\mathbf{x} = \mathbf{b}$ with an LU factorization is *much* faster than using A^{-1}. See Exercise 31.

HOW to implement LU factorization in python