

5.4: LU Decomposition

A variant of the Gaussian elimination algorithm can be used to compute the **LU decomposition** of a matrix. This procedure was invented by [Alan Turing](#), the British mathematician considered the "father of computer science". The LU decomposition of a square matrix \mathbf{A} consists of a lower-triangular matrix \mathbf{L} and an upper-triangular matrix \mathbf{U} , such that

$$\mathbf{A} = \mathbf{L} \mathbf{U}. \quad (5.4.1)$$

In certain special circumstances, LU decompositions provide a very efficient method for solving linear equations. Suppose that we have to solve a set of linear equations $\mathbf{A}\vec{x} = \vec{b}$ many times, using the *same* \mathbf{A} but an indefinite number of \vec{b} 's which might not be known in advance. For example, the \vec{b} 's might represent an endless series of measurement outcomes, with \mathbf{A} representing some fixed experimental configuration. We would like to efficiently calculate \vec{x} for each \vec{b} that arrives. If this is done with Gaussian elimination, each calculation would take $O(N^3)$ time.

However, if we can perform an LU decomposition *ahead of time*, then the calculations can be performed much more quickly. The linear equations are

$$\mathbf{L} \mathbf{U} \vec{x} = \vec{b}. \quad (5.4.2)$$

This can be broken up into two separate equations:

$$\mathbf{L} \vec{y} = \vec{b}, \quad \text{and} \quad \mathbf{U} \vec{x} = \vec{y}. \quad (5.4.3)$$

Because \mathbf{L} is lower-triangular, we can solve the first equation by forward-substitution (similar to back-substitution, except that it goes from the first row to last) to find \vec{y} . Then we can solve the second equation by back-substitution, to find \vec{x} . The whole process takes $O(N^2)$ time, which is a tremendous improvement over performing a wholesale Gaussian elimination.

However, finding the LU decomposition takes $O(N^3)$ time (we won't go into details here, but it's basically a variant of the row reduction phase of the Gaussian elimination algorithm). Therefore, if we are interested in solving the linear equations only once, or a handful of times, the LU decomposition method does not improve performance. It's useful in situations where the LU decomposition is performed ahead of time. You can think of the LU decomposition as a way of re-arranging the Gaussian elimination algorithm, so that we don't need to know \vec{b} during in the first, expensive $O(N^3)$ phase.

In Python, you can perform the LU decomposition using the `scipy.linalg.lu` function. The forward-substitution and back-substitution steps can be performed using `scipy.linalg.solve_triangular`.

This page titled [5.4: LU Decomposition](#) is shared under a [CC BY-SA 4.0](#) license and was authored, remixed, and/or curated by [Y. D. Chong](#) via [source content](#) that was edited to the style and standards of the LibreTexts platform.