Part 3

Chapter 10

LU Factorization

Chapter Objectives

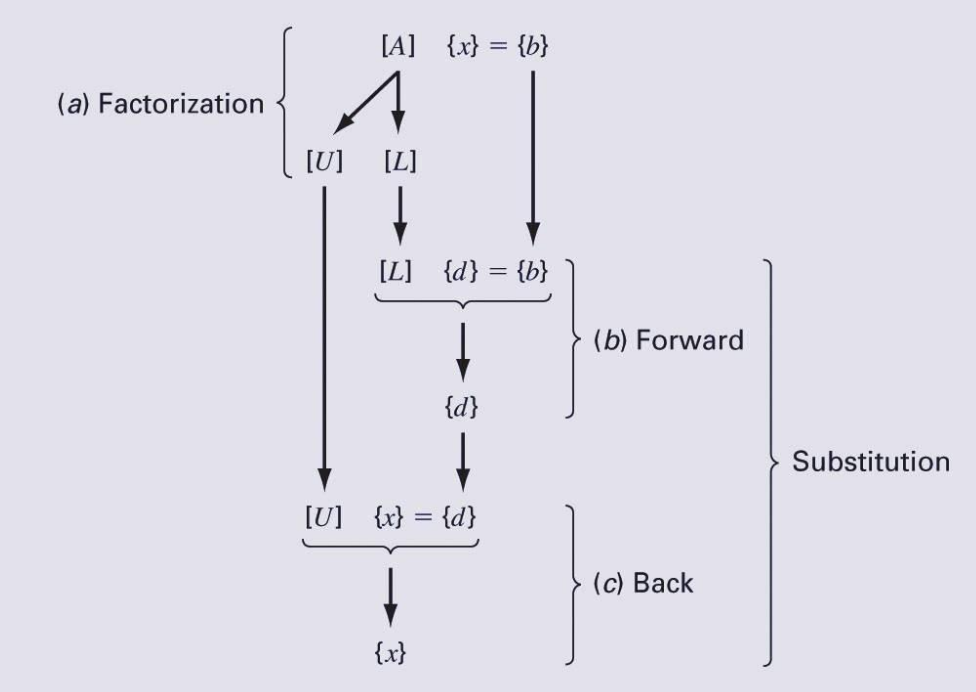
- Understanding that LU factorization involves decomposing the coefficient matrix into two triangular matrices that can then be used to efficiently evaluate different right-hand-side vectors.
- Knowing how to express Gauss elimination as an LU factorization.
- Given an LU factorization, knowing how to evaluate multiple right-hand-side vectors.
- Recognizing that Cholesky's method provides an efficient way to decompose a symmetric matrix and that the resulting triangular matrix and its transpose can be used to evaluate right-hand-side vectors efficiently.
- Understanding in general terms what happens when MATLAB's backslash operator is used to solve linear systems.

LU Factorization

- Recall that the forward-elimination step of Gauss elimination comprises the bulk of the computational effort.
- LU factorization methods separate the time-consuming elimination of the matrix [A] from the manipulations of the right-hand-side [b].
- Once [A] has been factored (or decomposed), multiple right-hand-side vectors can be evaluated in an efficient manner.

LU Factorization

- LU factorization involves two steps:
  - Factorization to decompose the [A] matrix into a product of a lower triangular matrix [L] and an upper triangular matrix [U]. [L] has 1 for each entry on the diagonal.
  - Substitution to solve for {x}
- Gauss elimination can be implemented using LU factorization



$$[L] = \begin{bmatrix} 1 & & \\ l_{21} & 1 & \\ l_{31} & l_{32} & 1 \end{bmatrix}$$

Gauss Elimination as *LU* Factorization

- To solve [A]{x}={b}, first decompose [A] to get [L][U]{x}={b}
- Set up and solve [L]{d}={b}, where {d} can be found using forward substitution.

$$d_i = b_i - \sum_{j=1}^{i-1} l_{ij} b_j \quad (i = 1, 2, \ldots, n)$$

- Set up and solve [U]{x}={d}, where {x} can be found using backward substitution.

$$x_n = d_n / a_{nn} \qquad x_i = \frac{d_i - \sum_{j=i+1}^{n} u_{ij} x_j}{u_{ij}} \quad (i = n-1, n-2, \ldots, 1)$$

- In MATLAB:

```
[L, U] = lu(A)
d = L\b
x = U\d
```

Gauss Elimination as *LU* Factorization (cont.)

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & & \\ f_{21} & 1 & \\ f_{31} & f_{32} & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ & a'_{22} & a'_{23} \\ & & a''_{33} \end{bmatrix}$$

A L U

[MATLAB built in function]
>> [L,U] = lu(X)

$$f_{21} = \frac{a_{21}}{a_{11}} \quad f_{31} = \frac{a_{31}}{a_{11}} \quad f_{32} = \frac{a'_{32}}{a'_{22}}$$

$$f_{12}a_{12} + a'_{22} \rightarrow f_{21}a_{12} + (a_{22} - f_{21}a_{12}) = a_{22}$$

Gauss Elimination as *LU* Factorization (cont.)

- [A]{x}={b} can be rewritten as [L][U]{x}={b} using LU factorization.
- The LU factorization algorithm requires the same total flops as for Gauss elimination.
- The main advantage is once [A] is decomposed, the same [L] and [U] can be used for multiple {b} vectors.
- MATLAB's lu function can be used to generate the [L] and [U] matrices:

[L, U] = lu(A)

Example 10.1 +10.2 (1/4)

- Q. To use LU decomposition to solve this problem

$$
\begin{aligned}
3x_1 &- 0.1x_2 - 0.2x_3 = 7.85 \\
0.1x_1 &+ 7x_2 - 0.3x_3 = -19.3 \\
0.3x_1 &- 0.2x_2 + 10x_3 = 71.4
\end{aligned}
$$

$$
\begin{bmatrix} 3 & -0.1 & -0.2 \\ 0.0999999 & 7 & -0.3 \\ 0.3 & -0.2 & 9.99996 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_{=3} \end{Bmatrix} = \begin{Bmatrix} 7.85 \\ -19.3 \\ 71.4 \end{Bmatrix}
$$

Forward Elimination :

$$
[U] = \begin{bmatrix} 3 & -0.1 & -0.2 \\ 0 & 7.00333 & -0.293333 \\ 0 & 0 & 10.0120 \end{bmatrix}
$$

$$
\begin{bmatrix} 3 & -0.1 & -0.2 \\ 0 & 7.00333 & -0.293333 \\ 0 & 0 & 10.0120 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_{=3} \end{Bmatrix} = \begin{Bmatrix} 7.85 \\ -19.5617 \\ 70.0843 \end{Bmatrix}
$$

Example 10.1 +10.2 (2/4)

$$f_{21} = \frac{0.1}{3} = 0.0333333 \qquad f_{31} = \frac{0.3}{3} = 0.1000000$$

$$f_{32} = \frac{-0.19}{7.00333} = -0.0271300$$

$$[A] = [L][U] = \begin{bmatrix} 1 & & \\ 0.0333333 & 1 & \\ 0.100000 & -0.0271300 & 1 \end{bmatrix} \begin{bmatrix} 3 & -0.1 & -0.2 \\ 0 & 7.00333 & -0.293333 \\ 0 & 0 & 10.0120 \end{bmatrix}$$

Confirm : $$[L][U] = \begin{bmatrix} 3 & -0.1 & -0.2 \\ 0.0999999 & 7 & -0.3 \\ 0.3 & -0.2 & 9.99996 \end{bmatrix} = [A]$$

Example 10.1 +10.2 (3/4)

Forward substitution:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0.0333333 & 1 & 0 \\ 0.100000 & -0.0271300 & 1 \end{bmatrix} \begin{Bmatrix} d_1 \\ d_2 \\ d_3 \end{Bmatrix} = \begin{Bmatrix} 7.85 \\ -19.3 \\ 71.4 \end{Bmatrix}$$

$$\begin{array}{rcrcrcl} d_1 & & & & & = & 7.85 \\ 0.0333333 d_1 & + & d_2 & & & = & -19.3 \\ 0.100000 d_1 & - & 0.027300 d_2 & + & d_3 & = & 71.4 \end{array}$$

$$d_2 = -19.3 - 0.0333333(7.85) = -19.5617$$

$$d_3 = 71.4 - 0.1(7.85) + 0.02713(-19.5617) = 70.0843$$

Example 10.1 +10.2 (4/4)

$$\{d\} = \begin{Bmatrix} 7.85 \\ -19.5617 \\ 70.0843 \end{Bmatrix}$$

To solve for x,

$$\begin{bmatrix} 3 & -0.1 & -0.2 \\ 0 & 7.00333 & -0.293333 \\ 0 & 0 & 10.0120 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 7.85 \\ -19.5617 \\ 70.0843 \end{Bmatrix}$$

$$\{x\} = \begin{Bmatrix} 3 \\ -2.5 \\ 7.00003 \end{Bmatrix}$$

Cholesky Factorization

- Symmetric systems occur commonly in both mathematical and engineering/science problem contexts, and there are special solution techniques available for such systems.

- The Cholesky factorization is one of the most popular of these techniques, and is based on the fact that a symmetric matrix can be decomposed as $[A]= [U]^T[U]$, where T stands for transpose.

- The rest of the process is similar to LU decomposition and Gauss elimination, except only one matrix, [U], needs to be stored.

MATLAB

- MATLAB can perform a Cholesky factorization with the built-in chol command:

    U = chol(A)

- MATLAB's left division operator \ examines the system to see which method will most efficiently solve the problem. This includes trying banded solvers, back and forward substitutions, Cholesky factorization for symmetric systems. If these do not work and the system is square, Gauss elimination with partial pivoting is used.