LU Decomposition

Ch. 10

• What is LU decomposition?
  - Another Class of Elimination methods
• Why do we want to use it?
  - The time consuming elimination step need only be performed on \([A]\) NOT \([b]\)
  - Situations where \([A]\) doesn’t change and \([b]\) does.
    • Trusses with varying external loads
    • Multiple spring/mass systems with varying masses

Note: Gauss Elimination can be expressed as an LU decomposition

LU Decomposition

Coefficients defined by the Geometry of the truss (e.g., cos \(\theta\))

\[
[A] [x] = [b]
\]

External Forces (i.e., \(P\))

Reaction forces & Internal forces (i.e., \(F's\) and \(R's\))

\(R_1\) \(R_2\) \(R_3\)
\(F_1\) \(F_2\) \(F_3\)
\(\theta_1\) \(\theta_2\) \(\theta_3\)
LU Decomposition

- Trusses with varying loads

\[ A \{x\} = \{b\} \]

External Forces (i.e., P)

Reaction forces & Internal forces (i.e., F's and R's)

\[ \{x\} = A^{-1}\{b\} \]

We would like to calculate the F's and R's, need an efficient way to calculate matrix inverse:

\[ b \times A = F \]

\[ F = \begin{bmatrix} F_1 \\ F_2 \\ R_1 \\ R_2 \end{bmatrix} \]

Overview of LU Decomposition

What is LU decomposition?

\[ [A] \rightarrow [L][U] \]

Lower Triangular Matrix

\[
[L] = \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

Upper Triangular Matrix

\[
[U] = \begin{bmatrix}
u_{11} & u_{12} & u_{13} \\
0 & u_{22} & u_{23} \\
0 & 0 & u_{33}
\end{bmatrix}
\]

LU Decomposition – A little mathematical Jugglery

Consider:

\[ [A]\{x\} = \{b\} \quad (1) \]

Rewrite (1) as:

\[ [A]\{x\} - \{b\} = 0 \quad (2) \]

From our Gauss Elimination Method, we know that (1) can be written as

\[
\begin{bmatrix}
u_{11} & u_{12} & u_{13} \\
0 & u_{22} & u_{23} \\
0 & 0 & u_{33}
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix} = \begin{bmatrix}
d_1 \\
d_2 \\
d_3
\end{bmatrix} \quad (3)
\]

Rewrite (3) as:

\[ [U]\{x\} - [d] = 0 \quad (4) \]
LU Decomposition – A little mathematical Jugglery

Assume a Lower Tridiagonal Matrix Exists:

\[
\begin{bmatrix}
1 & 0 & 0 \\
L_{11} & 1 & 0 \\
L_{12} & L_{13} & 1 \\
\end{bmatrix}
\]

If \( L \) is assumed to have the following property:

\[
[L](x) - [d] = [L](x) - [b]
\]

Then distributing \( L \) gives

\[
[L](x) - [L]d = [L](x) - [b]
\]

LU Decomposition

We now have the following three equations to use in a solution Strategy:

1. LU decomposition
   a. Factor \([A]\) into \([L]\) and \([U]\) (lower & Upper triangular Matrices)
2. Substitution
   a. Use (iii) to generate \(d\) by forward substitution
   b. Substitute \(d\) into (i) and use backward substitution to solve for \(x\)

LU Decomposition

\[
\begin{align*}
[L](x) &= [b] \\
U(x) &= d \\
L(x) &= [x]
\end{align*}
\]

\[
\begin{align*}
[L](x) &= [b] \\
U(x) &= d \\
L(x) &= [x]
\end{align*}
\]

\[
\begin{align*}
[L](x) &= [b] \\
U(x) &= d \\
L(x) &= [x]
\end{align*}
\]

\[
\begin{align*}
[L](x) &= [b] \\
U(x) &= d \\
L(x) &= [x]
\end{align*}
\]

\[
\begin{align*}
[L](x) &= [b] \\
U(x) &= d \\
L(x) &= [x]
\end{align*}
\]

\[
\begin{align*}
[L](x) &= [b] \\
U(x) &= d \\
L(x) &= [x]
\end{align*}
\]
Recall that Gauss Elimination gave:

\[
\begin{bmatrix}
U
\end{bmatrix} = \begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
0 & a'_{22} & a'_{23} \\
0 & 0 & a'_{33}
\end{bmatrix}
\]

\([L]\) was also produced during the forward elimination procedure, through our ‘factors’. Consider,

\[
\begin{align*}
a_{11} & \ a_{12} & \ a_{13} \\ a_{21} & \ a_{22} & \ a_{23} \\ a_{31} & \ a_{32} & \ a_{33}
\end{align*} = \begin{bmatrix}
x_1 \\
x_2 \\
x_3
\end{bmatrix}, \quad \begin{bmatrix}
b_1 \\
b_2 \\
b_3
\end{bmatrix}
\]

(1) \quad (2) \quad (3)

1. First elimination step, multiply (1) by \( f_{12} = \frac{a_{12}}{a_{11}} \) and subtract the result from (2) to Eliminate \( a_{21} \)
2. Next, Multiply (1) by \( f_{13} = \frac{a_{13}}{a_{11}} \) and subtract the result from (3) to eliminate \( a_{31} \)
3. Multiply (2’) by \( f_{23} = \frac{a'_{23}}{a'_{22}} \) and subtract the result from (3’) to eliminate \( a'_{32} \)

Now Perform operations only on \([A]\)
• Do NOT operate on \([b]\)
• Save \( f_{ij} \) and operate on \([b]\) later
• Store \( f_{ij} \) in the appropriate \( a_{ij} \) locations, i.e.,

\[
\begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
f_{21} & f_{22} & f_{23} \\
f_{31} & f_{32} & f_{33}
\end{bmatrix} \quad \text{Represents an efficient storage of} \quad \begin{bmatrix}
L \\
U
\end{bmatrix}
\]

Where,

\[
\begin{bmatrix}
1 & 0 & 0 \\
f_{21} & 1 & 0 \\
f_{31} & f_{32} & 1
\end{bmatrix} \quad \begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
0 & a'_{22} & a'_{23} \\
0 & 0 & a'_{33}
\end{bmatrix}
\]

Confirm:

\[
[A] \rightarrow [L][U]
\]

\[
\begin{bmatrix}
1 & 0 & 0 & a_{11} & a_{12} & a_{13} \\
f_{21} & 1 & 0 & 0 & a'_{22} & a'_{23} \\
f_{31} & f_{32} & 1 & 0 & 0 & a'_{33}
\end{bmatrix} \quad \begin{bmatrix}
a_{11} & a_{12} & a_{13} \\
a_{21} & a_{22} & a_{23} \\
a_{31} & a_{32} & a_{33}
\end{bmatrix}
\]
LU Decomposition – Gauss Elimination

Algorithm for Forward & Backward Substitution:

1. Forward Substitution, Solve for \( \{d_i\} \):

\[
\begin{bmatrix}
    L \cdot d \n
\end{bmatrix} = \begin{bmatrix} b \end{bmatrix}
\]

\[
\begin{bmatrix}
    1 & 0 & 0 & d_1 \\
    l_{21} & 1 & 0 & d_2 \\
    l_{31} & l_{32} & 1 & d_3
\end{bmatrix}
\begin{bmatrix}
    b_1 \\
    b_2 \\
    b_3
\end{bmatrix}
\]

\[
d_i = b_i \\
\text{for } i = 1
\]

\[
d_i = b_i - \sum_{j=1}^{i-1} l_{ij} d_j \\
\text{for } i = 2, 3, \ldots, n
\]

2. Backward Substitution:

\[
\begin{bmatrix}
    U \cdot x \n
\end{bmatrix} = \begin{bmatrix} d \end{bmatrix}
\]

\[
\begin{bmatrix}
    u_{11} & u_{12} & u_{13} & x_1 \\
    0 & u_{22} & u_{23} & x_2 \\
    0 & 0 & u_{33} & x_3
\end{bmatrix}
\begin{bmatrix}
    d_1 \\
    d_2 \\
    d_3
\end{bmatrix}
\]

\[
x_i = d_i / u_{ii}
\]

\[
x_i = \frac{d_i - \sum_{j=1}^{i-1} u_{ij} x_j}{u_{ii}} \\
\text{for } i = n, n-1, \ldots, 1
\]

Just like in Naïve Gauss Elimination

Simple LU Decomposition Example

Solve the following system of algebraic equations using Gauss Elimination based LU decomposition:

\[
\begin{bmatrix}
    4 & 3 & 1 & | & x_1 \\
    3 & -1 & 5 & | & x_2 \\
    2 & 4 & 2 & | & x_3
\end{bmatrix}
\begin{bmatrix}
    0 \\
    = 4 \\
    = 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
    L \cdot d \n
\end{bmatrix} = \begin{bmatrix} b \end{bmatrix}
\]

\[
\begin{bmatrix}
    1 & 0 & 0 \\
    l_{21} & 1 & 0 \\
    l_{31} & l_{32} & 1
\end{bmatrix}
\begin{bmatrix}
    d_1 \\
    d_2 \\
    d_3
\end{bmatrix}
\begin{bmatrix}
    b_1 \\
    b_2 \\
    b_3
\end{bmatrix}
\]

\[
d_i = b_i \\
\text{for } i = 1
\]

\[
d_i = b_i - \sum_{j=1}^{i-1} l_{ij} d_j \\
\text{for } i = 2, 3, \ldots, n
\]

\[
\begin{bmatrix}
    u_{11} & u_{12} & u_{13} & x_1 \\
    0 & u_{22} & u_{23} & x_2 \\
    0 & 0 & u_{33} & x_3
\end{bmatrix}
\begin{bmatrix}
    d_1 \\
    d_2 \\
    d_3
\end{bmatrix}
\begin{bmatrix}
    x_1 \\
    x_2 \\
    x_3
\end{bmatrix}
\]

\[
x_i = d_i / u_{ii}
\]

\[
x_i = \frac{d_i - \sum_{j=1}^{i-1} u_{ij} x_j}{u_{ii}} \\
\text{for } i = n, n-1, \ldots, 1
\]
Do Little LU Decomposition

- Requires all diagonal terms to be non-zero, \( a_{ii} \neq 0 \)
- Pivoting can be added
- Decompose \( A \) using
  
  \[
  u_{ji} = a_{ji} - \sum_{k=1}^{i-1} l_{ki} u_{kj} \quad \text{for } j=1,2,\ldots,n
  \]
  
  \[
  l_{ii} = a_{ii} - \sum_{k=1}^{i-1} l_{ki} u_{ki} \quad \text{for } i=1,2,\ldots,n
  \]

- Computation begins by defining
  
  \[
  l_{ij} = \frac{1}{u_{ij}} \quad \text{for } i=1,2,\ldots,n
  \]

- Next, for each \( j=2,\ldots,n \) compute \( u_{i,j} \) and \( l_{i,j} \) using (1) and (2) for increasing \( i \)
- Now, each coefficient is known when it is needed.
- \( U \) and \( L \) are thus found column by column.

Computational Effort

- Gauss Elimination
- Forward Elimination
- Backward Substitution
- LU Decomposition
- Forward Elimination
- Forward Substitution
- Backward Substitution
  - Less Effort during forward decomposition
  - Extra effort to do Forward Substitution

Both techniques require the same effort if only 1 set of \( \{b\}'s \) are used – \( n^3 \)

Benefits from LU decomposition result if you have many \( \{b\}'s \)