LU Decomposition Method

Jamie Trahan, Autar Kaw, Kevin Martin
University of South Florida
United States of America
kaw@eng.usf.edu

Introduction

When solving multiple sets of simultaneous linear equations with the same coefficient matrix but different right hand sides, LU Decomposition is advantageous over other numerical methods in that it proves to be numerically more efficient in computational time than other techniques. In this worksheet, the reader can choose a system of equations and see how each step of LU decomposition method is conducted. To learn more about LU Decomposition method as well as the efficiency of its computational time click here.

LU Decomposition method is used to solve a set of simultaneous linear equations, [A] [X] = [C], where $[A]_{nxn}$ is a non-singular square coefficient matrix, $[X]_{nx1}$ is the solution vector, and $[C]_{nx1}$ is the right hand side array. When conducting LU decomposition method, one must first decompose the coefficent matrix $[A]_{nxn}$ into a lower triangular matrix $[L]_{nxn}$, and upper triangular matrix $[U]_{nxn}$. These two matrices can then be used to solve for the solution vector $[X]_{nx1}$ in the following sequence:

Recall that
[A] [X] = [C].
Knowing that
[A] = [L] [U]
then first solving with forward substitution
[L] [Z] = [C]
and then solving with back substitution
[U] [X] = [Z]
gives the solution vector [X].

A simulation of LU Decomposition method follows.

Section 1: Input Data

Below are the input parameters to begin the simulation. This is the only section that requires user input. The user can change the values that are highlighted and *Mathematica* will calculate the solution vector [X].

- Number of equations

```
n = 4
4
```

- [A] *nxn* coefficient matrix

```
A = Table[{{175, 25, 5, 1}, {512, 64, 8, 1}, {1728, 144, 12, 1}, {8, 4, 2, 1}}];
A // MatrixForm
```

```
⎛ 175   25   5  1 ⎞
⎜ 512   64   8  1 ⎟
⎜1728  144  12  1 ⎟
⎝  8     4   2  1 ⎠
```

- [RHS] *n*x1 right hand side array

```
RHS = Table[{106.8, 177.2, 279.2, 79.3}]; RHS // MatrixForm
```

```
⎛106.8⎞
⎜177.2⎟
⎜279.2⎟
⎝ 79.3⎠
```

Section 2: LU Decomposition method

This section divides LU Decomposition into 3 steps:
1) Decomposition of coefficient matrix $[A]_{nxn}$
2) Forward Substitution
3) Back Substitution

■ Step 1: Finding [L] and [U]

How does one decompose a non-singular matrix [A], that is, how do you find [L] and [U]?
The following procedure decomposes the coefficient matrix [A] into a lower triangular matrix [L] and upper triangular matrix [U], given [A]=[L][U].

- For [U], the elements of the matrix are exactly the same as the coefficient matrix one obtains at the end of forward elimination steps in Naive Gauss Elimination.

- For [L], the matrix has 1 in its diagonal entries. The non-zero elements on the non-diagonal elements are multipliers that made the corresponding entries zero in the upper triangular matrix during forward elimination.

LU Decomposition procedure:

Variable names:
n = number of equations
U = *n*x*n* upper triangular matrix
L = *n*x*n* lower triangular matrix

```
LUdecompose[n_, A_] := Module[{AA, multiplier},
  L = Array[0, {n, n}];
  U = Array[0, {n, n}];
  AA = Array[0, {n, n}];
  For[i = 1, i ≤ n, i++, L[[i, i]] = 1;
   For[j = 1, j <= n, j++, AA[[i, j]] = A[[i, j]]; If[j != i, L[[i, j]] = 0]]];
  For[k = 1, k ≤ n - 1, k++,
   For[i = k + 1, i ≤ n, i++,
    multiplier = N[AA[[i, k]] / AA[[k, k]]];
    L[[i, k]] = multiplier;
    For[j = 1, j ≤ n, j++, AA[[i, j]] = AA[[i, j]] - multiplier * AA[[k, j]]]]]
   For[i = 1, i ≤ n, i++,
     For[j = 1, j ≤ n, j++, U[[i, j]] = AA[[i, j]]; If[j < i, U[[i, j]] = 0]]];];
  Print["L", "=", L // MatrixForm];
  Print["U", "=", U // MatrixForm]]
```

```
LUdecompose[n, A]
```

```
  (    1          0           0        0 )
  (  2.92571      1           0        0 )
L=(  9.87429    11.25         1        0 )
  ( 0.0457143  -0.3125  -0.00806452    1 )

  ( 175      25          5          1     )
  (  0   -9.14286   -6.62857   -1.92571   )
U=(  0       0         37.2       12.79   )
  (  0       0           0      0.455645  )
```

Notice that matrix [L] has only one unknown to be solved for in its *first* row, and matrix [U] has only one unknown to be solved for in its *last* row. This will prove useful in solving for the solution vector [X] in the following steps of LU decomposition method.

■ Step 2: Forward Substitution

Now that the [L] matrix has been formed, forward substitution step, [L][Z]=[C], can be conducted, beginning with the first equation as it has only one unknown,

$$z_1 = \frac{c_1}{l_{1,1}}$$ Equation (2.1)

Subsequent steps of forward substitution can be represented by the following formula:

$$z_i = \frac{(c_i - [\sum_{j=1}^{i-1} (l_{i,j} * z_j)]_{i=2..n})}{l_{i,i}}$$ Equation (2.2)

The procedure below conducts forward substitution steps to solve for $[Z]_{n\text{x}1}$.

Variable names:
n = number of equations
RHS = *n*x1 right hand side array
L = *n*x*n* lower triangular matrix

```
forwardsubstitution[n_, L_, C_] := Module[{},
  Z = Array[z, n];
             C[[1]]
  Z[[1]] = ---------- ;
           L[[1, 1]]
  For[i = 2, i ≤ n, i++, summm = 0;
   For[j = 1, j ≤ i - 1, j++, summ = summ + L[[i, j]] * Z[[j]]];
             (C[[i]] - summ)
   Z[[i]] = ---------------- ];
               L[[i, i]]
  Print["Z", "=", Z // MatrixForm]]

forwardsubstitution[n, L, RHS]

     106.8
   -135.266
Z= 746.372
   38.1661
```

■ Step 3: Back Substitution

Now that [Z] has been found, it can be used in the back substitution step, [U][X] = [Z], to solve for solution vector $[X]_{nx1}$, where $[U]_{nxn}$is the upper triangular matrix calculated in Step 1, and $[Z]_{nx1}$is the right hand side array.

Back substitution begins with solving the n^{th} equation as it has only one unknown.

$$x_n = \frac{z_n}{u_{n,n}} \qquad \text{Equation (2.3)}$$

The remaining unknowns are solved for working backwards from the $(n-1)^{\text{th}}$equation to the first equation using the formula below

$$x_i = \frac{(z_i - [\sum_{j=i+1}^{n}(u_{i,j} * x_j)]_{i=n-1..1})}{u_{i,i}} \qquad \text{Equation (2.4)}$$

The following procedure solves for $[X]_{nx1}$.

Variable names:
n = number of equations
Z = *n*x1 right hand side array
U = *n*x*n* upper triangular matrix

```
backsubstitution[n_, U_, Z_] := Module[{},
  X = Array[x, n];
            Z[[n]]
  X[[n]] = ―――――――――― ;
           U[[n, n]]
  For[i = n - 1, i >= 1, i--,
   summ = 0;
   For[j = i + 1, j ≤ n, j++, summ = summ + U[[i, j]] * X[[j]]];
             (Z[[i]] - summ)
   X[[i]] = ――――――――――――――― ];
                U[[i, i]]
  Print["X", "=", X // MatrixForm]]
```

The solution vector $[X]_{nx1}$ is:

```
backsubstitution[n, U, Z]
```

```
   ⎛-0.116681⎞
   ⎜ 3.48532 ⎟
X= ⎜-8.73534 ⎟
   ⎝ 83.7628 ⎠
```

Conclusion

Mathematica helped us apply our knowledge of LU Decomposition method to solve a system of *n* simultaneous linear equations. The results are as follows:

```
Print["L", "=", L // MatrixForm]
Print["U", "=", U // MatrixForm]
forwardsubstitution[n, L, RHS]
backsubstitution[n, U, Z]
```

$$L=\begin{pmatrix} 1 & 0 & 0 & 0 \\ 2.92571 & 1 & 0 & 0 \\ 9.87429 & 11.25 & 1 & 0 \\ 0.0457143 & -0.3125 & -0.00806452 & 1 \end{pmatrix}$$

$$U=\begin{pmatrix} 175 & 25 & 5 & 1 \\ 0 & -9.14286 & -6.62857 & -1.92571 \\ 0 & 0 & 37.2 & 12.79 \\ 0 & 0 & 0 & 0.455645 \end{pmatrix}$$

$$Z=\begin{pmatrix} 106.8 \\ -135.266 \\ 746.372 \\ 38.1661 \end{pmatrix}$$

$$X=\begin{pmatrix} -0.116681 \\ 3.48532 \\ -8.73534 \\ 83.7628 \end{pmatrix}$$

Question 1: Solve the following set of simultaneous linear equations using LU decomposition method.

$$\begin{pmatrix} 5 & 6 & 2.3 & 6 \\ 9 & 2 & 3.5 & 7 \\ 3.5 & 6 & 2 & 3 \\ 1.5 & 2 & 1.5 & 6 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 4 \\ 5 \\ 6.7 \\ 7.8 \end{pmatrix}$$

Question 2: Use LU decomposition repeatedly to find the inverse of

$$\begin{pmatrix} 5 & 6 & 2.3 & 6 \\ 9 & 2 & 3.5 & 7 \\ 3.5 & 6 & 2 & 3 \\ 1.5 & 2 & 1.5 & 6 \end{pmatrix}$$

Question 3: Look at the [Z] matrix in [L][Z]=[C] step in LU decomposition method of Question 1. Is it the same as the [RHS] matrix at the end of forward elimination steps in Naive Gauss Elimination method? If yes, is this a coincidence?

References

Autar Kaw, *Holistic Numerical Methods Institute*, *http://numericalmethods.eng.usf.edu/mws*, See
Introduction to Systems of Equations
How does LU Decomposition method work?
Saving of computational time for finding inverse of a matrix using LU decomposition