SOLUTION OF SYSTEMS OF LINEAR EQUATIONS AND APPLICATIONS WITH MATLAB® :

I - DIRECT METHODS

Selis Önel, PhD

Solution Methods for Linear Systems
$A\,x = y$

- Solution of Linear systems
  - **Direct Methods**
    - -Elimination Methods
    - -Inverse of a matrix
    - -Cramer's Rule
    - -LU Decomposition
  - Indirect Methods
    - Iterative Methods

Gauss Elimination

- $A x = y$
- Works better for coefficient matrices with no or few zeros
- Coefficient matrix A is augmented with the y matrix
- Good if $|A_{i,j}| \sim |y_i|$, Not good if the elements of y are too different from those of A
- The goal is to:
  - obtain an upper triangular matrix (lower triangular part all zeros) by forward elimination of the coefficients below the diagonal coefficients
  - Get the values of x by back substitution

Gauss Elimination

(1) $a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n = y_1$

(2) $a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_n = y_2$

..

(n) $a_{n,1}x_1 + a_{n,2}x_2 + \ldots + a_{n,n}x_n = y_n$

- Forward elimination: Eliminate the 1st term $a_{n,1}x_1$ of all rows below the 1st row, for n>1:
  - Multiply Eq(1) by $a_{n,1}/a_{1,1}$
  - Subtract from all equations as n=2:n

Gauss Elimination

First elimination:

(1) $a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n = y_1$

(2)' $0 + a'_{2,2}x_2 + \ldots + a'_{2,n}x_n = y'_2$

…

(n)' $0 + a'_{n,2}x_2 + \ldots + a'_{n,n}x_n = y'_n$

For $i > 1$

$$a'_{i,j} = a_{i,j} - \frac{a_{i,1}}{a_{1,1}} a_{1,j}$$

$$y'_i = y_i - \frac{a_{i,1}}{a_{1,1}} y_1$$

Gauss Elimination

Second elimination:

(1) $a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + \ldots + a_{1,n}x_n = y_1$

(2)' $0 + a'_{2,2}x_2 + a'_{2,3}x_3 + \ldots + a'_{2,n}x_n = y'_2$

(3)'' $0 + 0 + a''_{3,3}x_3 + \ldots + a''_{3,n}x_n = y'_3$

…

(n)$^{(n-1)}$ $0 + 0 + 0 + \ldots + a^{(n-1)}_{n,n}x_n = y^{(n-1)}_n$

For $i > 2$

$$a''_{i,j} = a'_{i,j} - \frac{a'_{i,2}}{a'_{2,2}} a'_{2,j}$$

$$y''_i = y'_i - \frac{a'_{i,2}}{a'_{2,2}} y'_2$$

Gauss Elimination

Back substitution:

- Forward elimination

  ↓

  Upper triangular matrix
- The last equation is

  $\mathbf{a^{(n-1)}_{n,n} x_n = y^{(n-1)}_n}$

$$x_n = \frac{y_n^{(n-1)}}{a_{n,n}^{(n-1)}}$$

$$x_{n-1} = \frac{y_{n-1}^{(n-2)} - a_{n-1,n}^{(n-2)} x_n}{a_{n-1,n-1}^{(n-2)}}$$

...

$$x_1 = \frac{y_1 - \sum_{i=2}^{n} a_{1,i} x_i}{a_{1,1}}$$

Gauss Elimination

Assume $m \neq n$

(1) $a_{1,1}x_1 + a_{1,2}x_2 + \ldots + a_{1,n}x_n = y_1$

(2) $a_{2,1}x_1 + a_{2,2}x_2 + \ldots + a_{2,n}x_n = y_2$

..

(m) $a_{m,1}x_1 + a_{m,2}x_2 + \ldots + a_{m,n}x_n = y_m$

- Check if $a_{1,1}=0$, if so reorganize, then
  - Multiply Eqn. (1) by $a_{i,1}/a_{1,1}$
  - Subtract from all equations as i=2:m
  - Repeat this procedure until the elements under $a_{i,i}$ are all 0

Gauss Elimination: If m<n

If the original equations are not all linearly independent,

Reduced equations may be:

(1) $a_{1,1}x_1 + a_{1,2}x_2 + \dots \dots + a_{1,n}x_n = y_1$

(2) $a_{2,2}x_2 + \dots \dots + a_{2,n}x_n = y_2$

..

(m) $a_{m,m}x_m + \dots + a_{m,n}x_n = y_m$

- Family of possible solutions with m variables

Gauss Elimination: If m>n

If elimination ends before the upper triangular matrix is obtained:

(1) $a_{1,1}x_1 + a_{1,2}x_2 + \quad \ldots \quad \ldots \quad + \quad a_{1,n}x_n = y_1$

(2) $\qquad a_{2,2}x_2 + \quad \ldots \quad \ldots \quad + \quad a_{2,n}x_n = y_2$

(..) $\qquad \ldots \qquad \ldots$

(k) $\qquad a_{k,k}x_k + a_{k,k+1}x_{k+1} + \ldots + a_{k,n}x_n = y_k$

(k+1) $\qquad 0 \quad = y_{k+1}$

(..) $\qquad \ldots \qquad \ldots$

(m) $\qquad 0 \quad = y_m$

- Family of possible solutions if n>k and $y_{k+1}, \ldots, y_m = 0$
- Unique solution if n=k and $y_{k+1}, \ldots, y_m = 0$
- Inconsistent equations if at least one of $y_{k+1}, \ldots, y_m \neq 0$

Example: Inconsistent system

$$\left(\begin{array}{ccc|c} 1 & 1 & 3 & 1 \\ 1 & 2 & 5 & 2 \\ 3 & 1 & 5 & -1 \end{array}\right)\begin{array}{c}(1)\\(2)\\(3)\end{array}$$

Augmented matrix [A , y]

$$\left(\begin{array}{ccc|c} 1 & 1 & 3 & 1 \\ 0 & 1 & 2 & 1 \\ 0 & -2 & -4 & -4 \end{array}\right)\begin{array}{c}(1)\\(2')=(2)-(1)\\(3')=(3)-3(1)\end{array}$$

First elimination

$$\left(\begin{array}{ccc|c} 1 & 1 & 3 & 1 \\ 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & -2 \end{array}\right)\begin{array}{c}(1)\\(2')\\(3'')=(3')+2(2')\end{array}$$

Second elimination

Result 0=-2 shows the equations are <u>inconsistent</u>

Det(A)=0

Pivoting

- If a diagonal element (or coefficient) $a_{i,i}$ becomes zero:

Forward elimination process cannot proceed

- If diagonal coefficients are not zero:

Pivoting helps to increase the accuracy of solution

Pivoting: Exchange the order of equations such that the diagonal coef. $a_{i,i}$ becomes larger in magnitude than any other element in the column

1st pivoting before 1st elimination:

Compare if $|a_{1,1}| >= |a_{i,1}|$,

if NOT exchange rows to make diagonal coefficient largest

2nd pivoting before 2nd elimination:

Compare if $|a'_{2,2}| >= |a'_{i,2}|$

Pivoting

For successful operations in Gauss elimination:

- Diagonal elements $|A_{i,i}| \neq 0$
- Reorganize the initial matrix to perform all necessary operations successfully

A=				After pivoting			
-2	4	1	2	-6	7	0	9
5	1	3	10	5	1	3	10
-6	7	0	9	-2	4	1	2

Ex: Gauss Elimination

Solve the following set of linear equations:

$-2\,x_1 + 4\,x_2 + x_3 = 2$

$5\,x_1 + x_2 + 3\,x_3 = 10$

$-6\,x_1 + 7\,x_2 = 9$

Solution: a x = y,

$$\begin{vmatrix} -2 & 4 & 1 \\ 5 & 1 & 3 \\ -6 & 7 & 0 \end{vmatrix} \begin{vmatrix} x_1 \\ x_2 \\ x_3 \end{vmatrix} = \begin{vmatrix} 2 \\ 10 \\ 9 \end{vmatrix}$$

A MATLAB® Program for Gauss Elimination

```
%Solve 3 linear equations for 3 unknowns by Gauss Elimination and Pivoting
%The augmented matrix is:
a=[-2 4 1 2; 5 1 3 10; -6 7 0 9];
%First pivoting (Exchange Eqn.1 with Eqn.3):
tempo=a(1,:);
a(1,:)=a(3,:);
a(3,:)=tempo;
a
%First elimination:  Multiply Eqn.1 by a(2,1)/a(1,1) and subtract from Eqn.2
a(2,:)=a(2,:)-a(1,:)*a(2,1)/a(1,1);
%Multiply Eqn.1 by a(3,1)/a(1,1) and subtract Eqn.3
a(3,:)=a(3,:)-a(1,:)*a(3,1)/a(1,1);
 a
%No more pivoting is required. 2nd elimination:
%Multiply Eqn.2 by a(3,2)/a(2,2) and subtract from Eqn.3
a(3,:)=a(3,:)-a(2,:)*a(3,2)/a(2,2);
a
%Forward elimination is done! Now backward substitution:
x(3)=a(3,4)/a(3,3);
x(2)=(a(2,4)-a(2,3)*x(3))/a(2,2);
x(1)=(a(1,4)-a(1,3)*x(3)-a(1,2)*x(2))/a(1,1);
x
```

Solution of the MATLAB® Program

After Pivoting a= -6 7 0 9
5 1 3 10
-2 4 1 2

First elimination:

a =

-6.00000000000000	7.00000000000000	0	9.00000000000000
0	6.83333333333333	3.00000000000000	17.50000000000000
0	1.66666666666667	1.00000000000000	-1.00000000000000

Second elimination:

a =

-6.00000000000000 7.00000000000000 0
9.00000000000000
0 6.83333333333333 3.00000000000000 17.50000000000000
0 0 0.26829268292683 -5.26829268292683

Back Substitution and Solution:

x =

11.54545454545455 11.18181818181818 -19.63636363636364

Homework Question 1

- Write a MATLAB® program to calculate the solution x for Ax=y using Gauss elimination method (with pivoting and normalization), where A is a coefficient matrix of any size.
- Hints:
  - Use m-files to create a general **function**
  - Make use of loops and if statements as necessary

Gauss-Jordan Elimination

- A x = y matrix is augmented [A y]
- All rows are pivoted and normalized by dividing by the pivot elements
- Coefficients both below and above the diagonal matrix are eliminated
- The goal is to:
  - Obtain an identity matrix for A, i.e. [A y]→[I y_{new}]
  - Get the unknown variables directly from [I y_{new}]
- No back substitution for the unknown variables
- Direct solution but requires more calculations than Gauss elimination

A MATLAB® Program for Gauss-Jordan Elimination

```
%Solve 3 linear equations for 3 unknowns by Gauss-Jordan Elimination
A=[-2 4 1 2; 5 1 3 10; -6 7 0 9];
%Pivoting for the first row (exchanging the 1st and 3rd rows):
tempo=A(1,:); A(1,:)=A(3,:); A(3,:)=tempo;
%Normalizing the 1st row (Dividing by its pivot):
A(1,:)=A(1,:)/A(1,1);
%Eliminating the elements below the 1st diagonal:
for i=2:3,        A(i,:)=A(i,:)-A(i,1)*A(1,:);
end,  A
%No more pivoting is required. Normalizing the 2nd row:
A(2,:)=A(2,:)/A(2,2); A
%Eliminating the elements above and below the 2nd diagonal:
for i=1:3,
    if   i~=2,      A(i,:)=A(i,:)-A(i,2)*A(2,:);
    end
end,   A
%Normalizing the 3rd row:
A(3,:)=A(3,:)/A(3,3);
%Eliminating the elements above the 3rd diagonal
for i=1:2,         A(i,:)=A(i,:)-A(i,3)*A(3,:);
end,    A
```

Solution of the MATLAB® Program

After Pivoting A=

-6	7	0	9
5	1	3	10
-2	4	1	2

Normalizing Equation 1 and First elimination:

A =

1.00000000000000	-1.16666666666667	0	-1.50000000000000
0	6.83333333333333	3.00000000000000	17.50000000000000
0	1.66666666666667	1.00000000000000	-1.00000000000000

Normalizing Equation 2

A =

1.00000000000000	-1.16666666666667	0	-1.50000000000000
0	1.00000000000000	0.43902439024390	2.56097560975610
0	1.66666666666667	1.00000000000000	-1.00000000000000

Solution of the MATLAB® Program

Second elimination:

A=

1.00000000000000	0	0.51219512195122	1.48780487804878
0	1.00000000000000	0.43902439024390	2.56097560975610
0	0	0.26829268292683	-5.26829268292683

Normalizing Equation 3 and third elimination:

A =

1.00000000000000	0	0	11.54545454545454
0	1.00000000000000	0	11.18181818181818
0	0	1.00000000000000	-19.63636363636363

Solution:

[I][x]=[y_{new}], thus

[x]=[y_{new}]

x =

11.54545454545455 11.18181818181818 -19.63636363636364

Inverse of a Matrix

- Inverse is useful in solving sets of linear equations
- Not all square matrices have inverses!
- If A is a non-singular matrix $A^{-1}A=AA^{-1}=I$
- Set of linear equations $Ax=y$

$$A^{-1}Ax=A^{-1}y \rightarrow x=A^{-1}y$$

Inverse of a Matrix

- Inverse of A is obtained by dividing the adjoint of A by the determinant of A:

$$A^{-1} = \frac{\{A_{ij}\}^t}{|A|} = \frac{1}{|A|} Adj(A)$$

Where A_{ij} is the **cofactor** of an element $a_{i,j}$ in a square matrix:
$A_{ij}=(-1)^{i+j}$ [Determinant of (n-1) by (n-1) matrix not containing i^{th} row and j^{th} column:

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \rightarrow A_{21} = (-1)^{2+1} \det \begin{pmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{pmatrix}$$

Where Adj(A) is the **adjoint** of matrix A which is the transpose of the matrix formed by replacing each element in A by its cofactor

$$Adj(A) = \{A_{ij}\}^t$$

Ex: Inverse of a Matrix

Find the inverse of $a = \begin{pmatrix} 2 & 1 & 0 \\ 1 & 3 & -1 \\ 0 & -2 & 1 \end{pmatrix}$ using the adjoint of a:

$$a^{-1} = \frac{\{A_{ij}\}^t}{\det(a)}$$

$$A_{11} = (-1)^{1+1}\det\begin{pmatrix} 3 & -1 \\ -2 & 1 \end{pmatrix} = 1,\quad A_{12} = (-1)^{1+2}\det\begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix} = -1,\quad A_{13} = (-1)^{1+3}\det\begin{pmatrix} 1 & 3 \\ 0 & -2 \end{pmatrix} = -2$$

$$A_{21} = (-1)^{2+1}\det\begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} = -1,\quad A_{22} = (-1)^{2+2}\det\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix} = 2,\quad A_{13} = (-1)^{2+3}\det\begin{pmatrix} 2 & 1 \\ 0 & -2 \end{pmatrix} = 4$$

$$A_{31} = (-1)^{3+1}\det\begin{pmatrix} 1 & 0 \\ 3 & -1 \end{pmatrix} = -1,\quad A_{12} = (-1)^{3+2}\det\begin{pmatrix} 2 & 0 \\ 1 & -1 \end{pmatrix} = 2,\quad A_{13} = (-1)^{3+3}\det\begin{pmatrix} 2 & 1 \\ 1 & 3 \end{pmatrix} = 5$$

$\{A_{ij}\} = \begin{pmatrix} 1 & -1 & -2 \\ -1 & 2 & 4 \\ -1 & 2 & 5 \end{pmatrix}$ =A Transpose of A is $A^t = \begin{pmatrix} 1 & -1 & -1 \\ -1 & 2 & 2 \\ -2 & 4 & 5 \end{pmatrix}$,

determinant of a is $\det(a) = (2\cdot 3\cdot 1) + (1\cdot -1\cdot 0) + (0\cdot 1\cdot -2) - (0\cdot 3\cdot 0) - (-2\cdot -1\cdot 2) - (1\cdot 1\cdot 1) = 1$

$$a^{-1} = \frac{\{A_{ij}\}^t}{\det(a)} = \frac{\begin{pmatrix} 1 & -1 & -1 \\ -1 & 2 & 2 \\ -2 & 4 & 5 \end{pmatrix}}{1} = \begin{pmatrix} 1 & -1 & -1 \\ -1 & 2 & 2 \\ -2 & 4 & 5 \end{pmatrix}$$

Inverse of a Matrix: Gauss-Jordan Elimination

- $x=\mathbf{A}^{-1}y$
- If A is a non-singular square matrix $A^{-1}A=AA^{-1}=I$
- Write an augmented matrix of the original matrix [A] and the identity matrix [I]
- Pivot
- Normalize with respect to the pivoting element
- Eliminate elements below and above the diagonal

A MATLAB® Program for Matrix Inversing

```
a=[-2 4 1; 5 1 3; -6 7 0];   %Create an augmented matrix of a and I:
A=[a, eye(3)];    A
%Pivoting for the first row (exchanging the 1st and 3rd rows):
tempo=A(1,:); A(1,:)=A(3,:); A(3,:)=tempo;
%Normalizing the 1st row (Dividing by its pivot):
A(1,:)=A(1,:)/A(1,1);
%Eliminating the elements below the 1st diagonal:
for i=2:3,        A(i,:)=A(i,:)-A(i,1)*A(1,:);
end,              A
%No more pivoting is required. Normalizing the second row:
A(2,:)=A(2,:)/A(2,2); A
%Eliminating the elements above and below the 2nd diagonal:
for i=1:3,
   if i~=2,             A(i,:)=A(i,:)-A(i,2)*A(2,:);
   end
end,                    A
%Normalizing the 3rd row:
A(3,:)=A(3,:)/A(3,3);
%Eliminating the elements above the 3rd diagonal
for i=1:2,              A(i,:)=A(i,:)-A(i,3)*A(3,:);
end,        A
```

Solution of the MATLAB® Program

A=

-2	4	1	1	0	0
5	1	3	0	1	0
-6	7	0	0	0	1

……………………….. Augmented matrix A

A=

1.0000	-1.1667	0	0	0	-0.1667
0	6.8333	3.0000	0	1.0000	0.8333
0	1.6667	1.0000	1.0000	0	-0.3333

Normalizing 1st row and 1st elimination

A =

1.0000	-1.1667	0	0	0	-0.1667
0	1.0000	0.4390	0	0.1463	0.1220
0	1.6667	1.0000	1.0000	0	-0.3333

Normalizing 2nd row

Solution of the MATLAB® Program

A =

1.0000	0	0.5122	0	0.1707	-0.0244
0	1.0000	0.4390	0	0.1463	0.1220
0	0	0.2683	1.0000	-0.2439	-0.5366

2nd elimination

A =

1.0000	0	0	-1.9091	0.6364	1.0000
0	1.0000	0	-1.6364	0.5455	1.0000
0	0	1.0000	3.7273	-0.9091	-2.0000

Normalizing 3rd row

and 3rd elimination

a^{-1}=

-1.9091	0.6364	1.0000
-1.6364	0.5455	1.0000
3.7273	-0.9091	-2.0000

Finding the solution from the inverse

```
>> a_inv=[A(:,4) A(:,5) A(:,6)]
>> y=[2;10;9];        %This is the y vector of the original problem
>> x=a_inv*y
a_inv =
  -1.9091    0.6364    1.0000
  -1.6364    0.5455    1.0000
   3.7273   -0.9091   -2.0000
x =
  11.5455
  11.1818
 -19.6364
```

Cramer's Rule

- Gives the solution of a system of linear equations in terms of determinants
- Named after Gabriel Cramer (1704 - 1752), who published the rule in his 1750 *Introduction à l'analyse des lignes courbes algébriques*
- Inefficient for large matrices and thus not used in practical applications that involve many equations.
- No pivoting is needed, so more efficient than Gauss elimination for small matrices

$$x_i = \frac{\det(\mathbf{A}_i)}{\det(\mathbf{A})}$$

Ax=y

where A_i is the matrix formed by replacing the *i*th column of A by the column vector *y*

Example: Cramer's Rule

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= y_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= y_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 &= y_3 \end{aligned}$$

is a linear system of equations and can be written in matrix notation as |a||x|=|c|:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

Replacing each column of |a| with |y| gives the following solutions for |x|:

$$x_1 = \frac{\det\begin{pmatrix} y_1 & a_{12} & a_{13} \\ y_2 & a_{22} & a_{23} \\ y_3 & a_{32} & a_{33} \end{pmatrix}}{\det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}} \quad x_2 = \frac{\det\begin{pmatrix} a_{11} & y_1 & a_{13} \\ a_{21} & y_2 & a_{23} \\ a_{31} & y_3 & a_{33} \end{pmatrix}}{\det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}} \quad x_3 = \frac{\det\begin{pmatrix} a_{11} & a_{12} & y_1 \\ a_{21} & a_{22} & y_2 \\ a_{31} & a_{32} & y_3 \end{pmatrix}}{\det\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}}$$

Homework Question 2:

- Solve the following set of equations using
  1. Matrix inversion
  2. Gauss-Jordan relation
  3. Cramer's rule

$$x_1 + x_2 + 3x_3 = 1$$
$$x_1 + 2x_2 + 5x_3 = 2$$
$$3x_1 + x_2 + x_3 = 1$$

LU Decomposition

A x = y, where A = L U $\rightarrow$ LUx=y

Let Ux=z, then Lz=y

- A matrix may be decomposed to L (lower triangular matrix) and U (upper triangular matrix) using Gauss elimination
- Remember: The matrix after forward elimination is U
- Taking the inverse of L and U is easy and fast
- Inverse of L is a lower triangular matrix

LU Decomposition

$A\,x = y$, where $A = L\,U \rightarrow LUx=y$
Let $Ux=z$, then $Lz=y$

1. Apply Gauss elimination to [A,I] together
2. Transform $A \rightarrow U$ and $I \rightarrow F$ (Gauss elimination)
3. $FA=U$ and $FI=F$
4. $F=L^{-1}$, so $L=F^{-1}$
5. $z=L^{-1}y$, $x=U^{-1}z$

In MATLAB® **[l,u,p]=lu(A)** where $PA=LU$ or
[l,u]=lu(A) where $l=P^{-1}L$ and $u=U$
(P : Permutation matrix)

Example: LU Decomposition I

```
>> a=[-2 4 1 ; 5 1 3 ; -6 7 0 ];
>> [l,u,p]=lu(a)
l =
  1.0000        0         0
 -0.8333   1.0000         0
  0.3333   0.2439    1.0000
u =
 -6.0000   7.0000         0
       0   6.8333    3.0000
       0        0    0.2683
p =
   0   0   1
   0   1   0
   1   0   0
```

```
%gives the pivoted matrix a,
%i.e. pA=lu
>> l*u
ans =
   -6   7   0
    5   1   3
   -2   4   1
```

```
%gives the original matrix a,
%i.e. A=p^-1 lu
>> p^(-1)*l*u
ans =
   -2   4   1
    5   1   3
   -6   7   0
```

```
>> z=(p^(-1)*l)\y
z =
    9.0000
   17.5000
   -5.2683
>> x=u\z
x =
   11.5455
   11.1818
  -19.6364
```

Example: LU Decomposition II

```
>> a=[-2 4 1 ; 5 1 3 ; -6 7 0 ];
>> [l,u]=lu(a)
l =
    0.3333    0.2439    1.0000
   -0.8333    1.0000         0
    1.0000         0         0
u =
   -6.0000    7.0000         0
         0    6.8333    3.0000
         0         0    0.2683
```

```
% gives the original matrix a,
% i.e. A=lu, where l=P^-1L
>> l*u
ans =
    -2     4     1
     5     1     3
    -6     7     0
```

```
% gives the pivoted matrix a,
% i.e. A=p^-1lu
>> p^(-1)*l*u
ans =
    -6     7     0
     5     1     3
    -2     4     1
```

```
>> z=l\y
z =
    9.0000
   17.5000
   -5.2683
>> x=u\z
x =
   11.5455
   11.1818
  -19.6364
```

MATLAB® Commands for Creating Vectors

A=[3 pi sqrt(5) (3-5j)]	Create row vector A containing elements specified
A=1:100	Create row vector A starting with 1, counting by 1, ending at or before 100
A=100:-10:0	Create row vector A starting with 100, counting by -10 (decrementing), ending at or before 0
A=linspace(0,100,11)	Create row vector A starting with 10, ending at 100, having 11 elements (ex: 0,10,20,30,…100)
A=logspace(0,3,4)	Create row vector A starting with 10^0, ending at 10^3, having 4 elements
length(A)	Returns the length of vector A
d=size(M)	For m-by-n matrix M, returns the two-element row vector
[m,n]=size(M)	For m-by-n matrix M, returns the number of rows and columns in M as separate output variables