Part 3

Chapter 9

Gauss Elimination

School of Mechanical Engineering
Chung-Ang University

Chapter Objectives

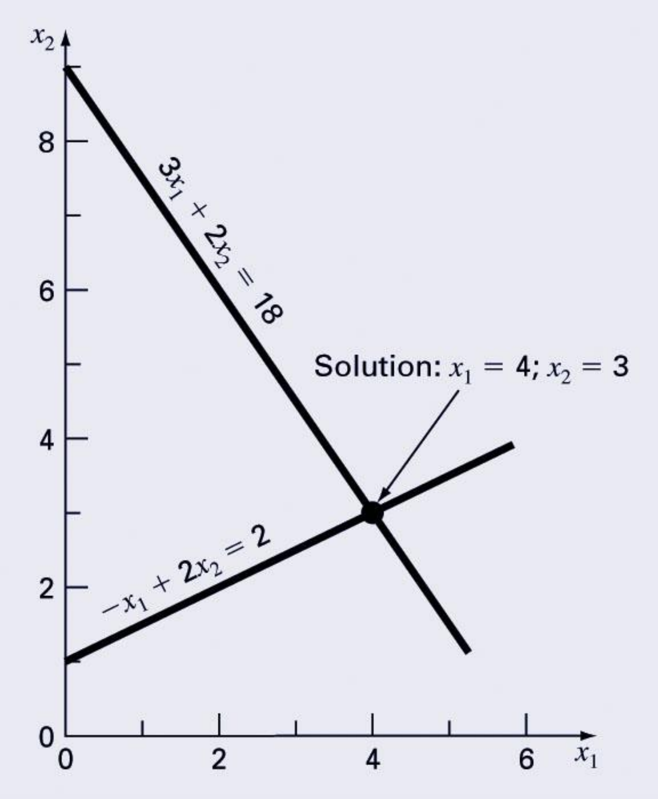
- Knowing how to solve small sets of linear equations with the graphical method and Cramer's rule.
- Understanding how to implement forward elimination and back substitution as in Gauss elimination.
- Understanding how to count flops to evaluate the efficiency of an algorithm.
- Understanding the concepts of singularity and ill-condition.
- Understanding how partial pivoting is implemented and how it differs from complete pivoting.
- Recognizing how the banded structure of a tridiagonal system can be exploited to obtain extremely efficient solutions.

Graphical Method

- Small numbers (n<=3) of equations: Graphical method,Cramer's rule, elimination of unknowns.

- For small sets of simultaneous equations, graphing them and determining the location of the intercept provides a solution.

- Ex. for three (two) simultaneous equations, the point where the three planes (two lines) intersect would represent the solution.

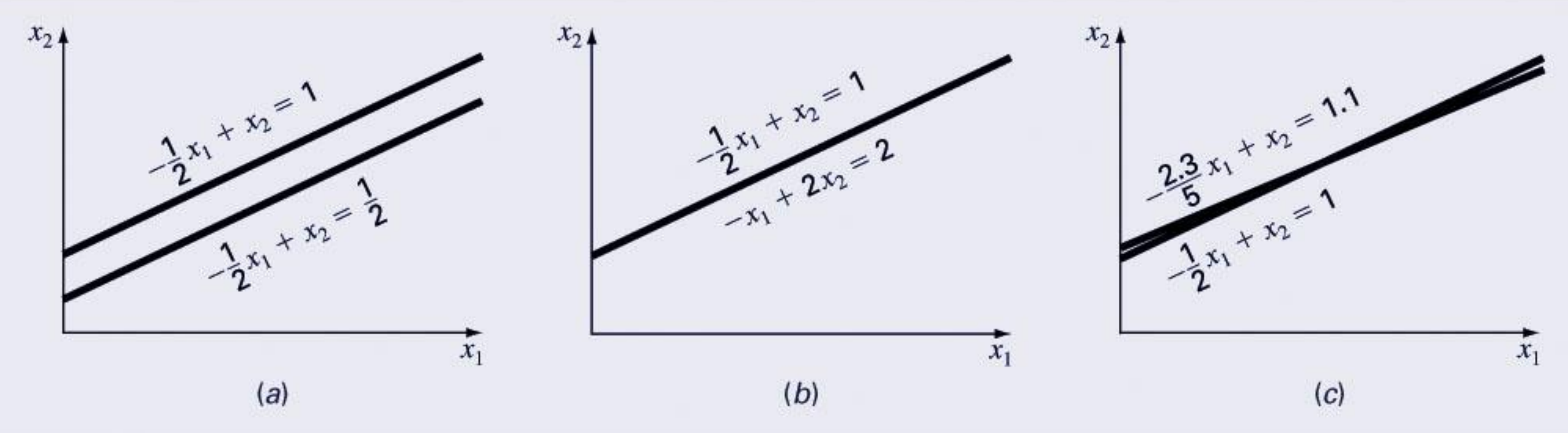


Graphical Method (cont)

- Graphing the equations can also show systems where:
  - a) No solution exists
  - b) Infinite solutions exist
  - c) System is ill-conditioned
    - Extremely sensitive to round off error.
    - The point of intersection is difficult to detect visually

a), b) → Singular

c) → Almost Singular



Determinants

- The determinant D=|A| of a matrix is formed from the coefficients of [A].
- Determinants for small matrices are:

1×1	$\left\lvert a_{11}\right\rvert = a_{11}$
2×2	$\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$
3×3	$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix}$

- Determinants for matrices larger than 3 x 3 can be very complicated.

Cramer's Rule

- Cramer's Rule states that each unknown in a system of linear algebraic equations may be expressed as a fraction of two determinants with denominator D and with the numerator obtained from D by replacing the column of coefficients of the unknown in question by the constants $b_1, b_2, \ldots, b_n$.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

$$x_1 = \frac{\begin{vmatrix} b_1 & a_{12} & a_{13} \\ b_2 & a_{22} & a_{23} \\ b_3 & a_{32} & a_{33} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}}, x_2 = \frac{\begin{vmatrix} a_{11} & b_1 & a_{13} \\ a_{21} & b_2 & a_{23} \\ a_{31} & b_3 & a_{33} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}}, x_3 = \frac{\begin{vmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ a_{31} & a_{32} & b_3 \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}}$$

Cramer's Rule Example

- Find x_2 in the following system of equations:

$$\begin{aligned} 0.3x_1 + 0.52x_2 + x_3 &= \boxed{-0.01} \\ 0.5x_1 + x_2 + 1.9x_3 &= \boxed{0.67} \\ 0.1x_1 + 0.3x_2 + 0.5x_3 &= \boxed{-0.44} \end{aligned}$$

- Find the determinant D

$$D = \begin{vmatrix} 0.3 & 0.52 & 1 \\ 0.5 & 1 & 1.9 \\ 0.1 & 0.3 & 0.5 \end{vmatrix} = 0.3\begin{vmatrix} 1 & 1.9 \\ 0.3 & 0.5 \end{vmatrix} - 0.52\begin{vmatrix} 0.5 & 1.9 \\ 0.1 & 0.5 \end{vmatrix} + 1\begin{vmatrix} 0.5 & 1 \\ 0.1 & 0.4 \end{vmatrix} = -0.0022$$

- Find determinant D_2 by replacing D's second column with b

$$D_2 = \begin{vmatrix} 0.3 & \boxed{-0.01} & 1 \\ 0.5 & \boxed{0.67} & 1.9 \\ 0.1 & \boxed{-0.44} & 0.5 \end{vmatrix} = 0.3\begin{vmatrix} 0.67 & 1.9 \\ -0.44 & 0.5 \end{vmatrix} - 0.01\begin{vmatrix} 0.5 & 1.9 \\ 0.1 & 0.5 \end{vmatrix} + 1\begin{vmatrix} 0.5 & 0.67 \\ 0.1 & -0.44 \end{vmatrix} = 0.0649$$

- Divide

$$x_2 = \frac{D_2}{D} = \frac{0.0649}{-0.0022} = -29.5$$

Naïve Gauss Elimination

- For larger systems, Cramer's Rule can become unwieldy.
- Instead, a sequential process of removing unknowns from equations using forward elimination followed by backward substitution may be used - this is Gauss elimination.
- "Naïve" Gauss elimination simply means the process does not check for potential problems resulting from division by zero.

  → not need pivoting

Naïve Gauss Elimination (cont)

- Forward elimination
  - Starting with the first row, add or subtract multiples of that row to eliminate the first coefficient from the second row and beyond.
  - Continue this process with the second row to remove the second coefficient from the third row and beyond.
  - Stop when an upper triangular matrix remains.
- Back substitution
  - Starting with the last row, solve for the unknown, then substitute that value into the next highest row.
  - Because of the upper-triangular nature of the matrix, each row will contain only one more unknown.

$$\left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & b_3 \end{array}\right]$$

$$\downarrow$$

$$\left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & b_1 \\ & a'_{22} & a'_{23} & b'_2 \\ & & a''_{33} & b''_3 \end{array}\right]$$

(*a*) Forward elimination

$$\downarrow$$

$$x_3 = b''_3/a''_{33}$$

$$x_2 = (b'_2 - a'_{23}x_3)/a'_{22}$$

$$x_1 = (b_1 - a_{13}x_3 - a_{12}x_2)/a_{11}$$

(*b*) Back substitution

Naïve Gauss Elimination (cont)

- N equations:

Pivot element

Pivot equation

$$
\begin{array}{ccccccccccccc}
\boxed{a_{11}x_1 & + & a_{12}x_1 & + & a_{13}x_1 & + & \cdots & + & a_{1n}x_1 & = & b_1} \\
a_{21}x_1 & + & a_{22}x_2 & + & a_{23}x_2 & + & \cdots & + & a_{2n}x_2 & = & b_2 \\
 & & \vdots & & \vdots & & & & & & \\
a_{n1}x_1 & + & a_{n2}x_2 & + & a_{n3}x_3 & + & \cdots & + & a_{nn}x_n & = & b_n
\end{array}
$$

Naïve Gauss Elimination (cont)

- Forward Elimination:

  Reduce the set of equations to an upper triangular matrix.

- Eliminate the first unknown x_1 from the second through the n th equations. To do this, multiply the first equation by a_{21}/a_{11} to give

$$a_{21}x_1 + \frac{a_{21}}{a_{11}}a_{12}x_2 + \frac{a_{21}}{a_{11}}a_{13}x_3 + \cdots + \frac{a_{21}}{a_{11}}a_{1n}x_n = \frac{a_{21}}{a_{11}}b_1$$

- This equation can be subtracted from the second equation by:

$$\left(a_{22} - \frac{a_{21}}{a_{11}}a_{12}\right)x_2 + \cdots + \left(a_{2n} - \frac{a_{21}}{a_{11}}a_{1n}\right)x_n = b_2 - \frac{a_{21}}{a_{11}}b_1$$

or $a'_{22}x_2 + \cdots + a'_{2n}x_n = b'_2$

Naïve Gauss Elimination (cont)

- The procedure is repeated for the remaining equations.

$$\begin{array}{ccccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \cdots & + & a_{1n}x_n & = & b_1 \\
 & & a'_{22}x_2 & + & a'_{23}x_3 & + & \cdots & + & a'_{2n}x_n & = & b'_2 \\
 & & a'_{32}x_2 & + & a'_{33}x_3 & + & \cdots & + & a'_{3n}x_n & = & b'_3 \\
 & & \vdots & & \vdots & & & & & & \\
 & & a'_{n2}x_2 & + & a'_{n3}x_3 & + & \cdots & + & a'_{nn}x_n & = & b'_n
\end{array}$$

- Using the second pivot equation to remove x_2 from the third through the nth equations to give

$$\begin{array}{ccccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \cdots & + & a_{1n}x_n & = & b_1 \\
 & & a'_{22}x_2 & + & a'_{23}x_3 & + & \cdots & + & a'_{2n}x_n & = & b'_2 \\
 & & & & a''_{33}x_3 & + & \cdots & + & a''_{3n}x_n & = & b''_3 \\
 & & & & \vdots & & & & & \vdots & \\
 & & & & a''_{n3}x_3 & + & \cdots & + & a''_{nn}x_n & = & b''_n
\end{array}$$

Naïve Gauss Elimination (cont)

- Lastly we can have the following set of equations:

$$
\begin{array}{ccccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \cdots & + & a_{1n}x_n & = & b_1 \\
 & & a_{22}^{'}x_2 & + & a_{23}^{'}x_3 & + & \cdots & + & a_{2n}^{'}x_n & = & b_2^{'} \\
 & & & & a_{33}^{''}x_3 & + & \cdots & + & a_{3n}^{''}x_n & = & b_3^{''} \\
 & & & & & \ddots & & & & \vdots & \\
 & & & & & & & & a_{nn}^{(n-1)}x_n & = & b_n^{(n-1)}
\end{array}
$$

Naïve Gauss Elimination (cont)

- Backward substitution

$$
\begin{array}{ccccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & a_{13}x_3 & + & \cdots & + & a_{1n}x_n & = & b_1 \\
 & & a'_{22}x_2 & + & a'_{23}x_3 & + & \cdots & + & a'_{2n}x_n & = & b'_2 \\
 & & & & a''_{33}x_3 & + & \cdots & + & a''_{3n}x_n & = & b''_3 \\
 & & & & & \ddots & & & & \vdots & \\
 & & & & & & & & a_{nn}^{(n-1)}x_n & = & b_n^{(n-1)}
\end{array}
$$

$$x_n = \frac{b_n^{(n-1)}}{a_{nn}^{(n-1)}}$$

There is only one variable in the n th row.

$$x_i = \frac{b_i^{(i-1)} - \sum_{j=i+1}^{n} a_{ij}^{(i-1)} x_j}{a_{ii}^{(i-1)}}$$

(i = n-1, n-2, …,1)

Example 9.3 (1/2)

- Q. Use the Gaussian elimination to find the solution.

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

Forward elimination:

$$\begin{aligned} 3x_1 - 0.1x_2 - 0.2x_3 &= 7.85 \\ 7.00333x_2 - 0.293333x_3 &= -19.5617 \\ -0.190000x_2 + 10.0200x_3 &= 70.6150 \end{aligned}$$

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

Example 9.3 (2/2)

- Backward substitution:

$$x_3 = \frac{70.0843}{10.0120} = 7.00003$$

$$x_2 = \frac{-19.5617 + 0.293333(7.00003)}{7.00333} = -2.50000$$

$$x_1 = \frac{7.85 + 0.1(-2.50000) + 0.2(7.00003)}{3} = 3.00000$$

- Verification:

$$3(3) - 0.1(-2.5) - 0.2(7.00003) = 7.84999 \cong 7.85$$
$$0.1(3) + 7(-2.5) - 0.3(7.00003) = -19.30000 \cong -19.3$$
$$0.3(3) - 0.2(-2.5) + 10(7.00003) = 71.4003 \cong 71.4$$

Naïve Gauss Elimination Program


```
function x = GaussNaive(A,b)
% GaussNaive (A,b):
%   Gauss elimination without pivoting
% input:
%   A = coefficient matrix
%   b = right hand side vector
% output:
%   x = solution vector
[m,n] = size(A);
if m ~= n, error('Matrix A must be square'); end
nb = n+1;
Aug =[A b];
```

```
% forward elimination
for k = 1:n-1
   for i = k+1:n
      factor = Aug(i,k)/Aug(k,k);
      Aug(i,k:nb) = Aug(i,k:nb)-factor*Aug(k,k:nb);
   end
end
% back substitution
x = zeros(n,1);
x(n) = Aug(n,nb)/Aug(n,n);
for i = n-1:-1:1
   x(i) = (Aug(i,nb)-Aug(i,i+1:n)*x(i+1:n))/Aug(i,i);
end
```

Gauss Program Efficiency

- The execution of Gauss elimination depends on the amount of floating-point operations (or flops). The flop count for an n x n system is:

Forward Elimination	$\frac{2n^3}{3} + O(n^2)$
Back Substitution	$n^2 + O(n)$
Total	$\frac{2n^3}{3} + O(n^2)$

- Conclusions:
  - As the system gets larger, the computation time increases greatly.
  - Most of the effort is incurred in the elimination step.

Pivoting

- Problems arise with naïve Gauss elimination if a coefficient along the diagonal is 0 (problem: division by 0) or close to 0 (problem: round-off error)
- One way to combat these issues is to determine the coefficient with the largest absolute value in the column below the pivot element. The rows can then be switched so that the largest element is the pivot element. This is called partial pivoting.
- If the rows to the right of the pivot element are also checked and columns switched, this is called complete pivoting.

Example 9.4 (partial pivoting) [1/3]

- Q. Use the Gaussian elimination to solve this.

$$0.0003x_1 + 3.0000x_2 = 2.0001 \quad \text{(1)}$$
$$1.0000x_1 + 1.0000x_2 = 1.0000 \quad \text{(2)}$$

Solution: $x_1 = 1/3$ and $x_2 = 2/3$.

- If no partial pivoting,

$$x_1 + 10{,}000x_2 = 6667 \quad \text{(1) * 1/0.0003} \rightarrow \text{(3)}$$
$$-9999x_2 = -6666 \quad \text{(2) – (3)}$$

$$x_2 = 2/3$$

$$x_1 = \frac{2.0001 - 3(2/3)}{0.0003}$$

Example 9.4 (partial pivoting) [2/3]

$$x_2 = 2/3 \qquad x_1 = \frac{2.0001 - 3(2/3)}{0.0003}$$

- The result is very sensitive to the number of significant figures carried in the computation.

Significant Figures	x_2	x_1	***Absolute Value of Percent Relative Error for x1***
3	0.667	-3.33	1099
4	0.6667	0.0000	100
5	0.66667	0.30000	10
6	0.666667	0.330000	1
7	0.6666667	0.3330000	0.1

Example 9.4 (partial pivoting) [2/3]

- With partial pivoting

$$1.0000x_1 + 1.0000x_2 = 1.0000 \quad (1')$$

$$0.0003x_1 + 3.0000x_2 = 2.0001 \quad (2')$$

$$x_2 = \frac{2}{3} \text{ and } x_1 = \frac{1-(2/3)}{1}$$

Significant Figures	***x_2***	***x_1***	***Absolute Value of Percent Relative Error for x1***
3	0.667	0.333	0.1
4	0.6667	0.3333	0.01
5	0.66667	0.33333	0.001
6	0.666667	0.333333	0.0001
7	0.6666667	0.3333333	0.00001

Partial Pivoting Program

Untitled

File Edit View Text Debug Breakpoints Web Window Help

Stack: Base

```
function x = GaussPivot(A,b)
% GaussPivot (A,b):
%   Gauss elimination with partial pivoting
% input:
%   A = coefficient matrix
%   b = right hand side vector
% output:
%   x = solution vector
[m,n] = size(A);
if m ~= n, error('Matrix A must be square'); end
nb = n+1;
Aug =[A b];
```

Ready

```
% forward elimination
for k = 1:n-1
    % partial pivoting
    [big, i] = max(abs(Aug(k:n,k)));
    ipr = i + k -1;
    if ipr  ~= k
        % pivot the row
        Aug([k,ipr],:) = Aug([ipr,k],:);
    end
```

```
    for i = k+1:n
    factor = Aug(i,k)/Aug(k,k);
    Aug(i,k:nb) = Aug(i,k:nb)-factor*Aug(k,k:nb);
    end
end
% back substitution
x = zeros(n,1);
x(n) = Aug(n,nb)/Aug(n,n);
for i = n-1:-1:1
    x(i) = (Aug(i,nb)-Aug(i,i+1:n)*x(i+1:n))/Aug(i,i);
end
```

Command Window

File Edit View Web Window Help

```
>> y = [1; 3; 5; 2; 4];
>> [ymax, imax] =max(y)
ymax =
     5
imax =
     3

>> A=[0.0003 3; 1 1]; b=[2.0001; 1];
>> x = GaussPivot(A,b)
x =
   0.3333
   0.6667
```

Ready

Tridiagonal Systems

- A tridiagonal system is a banded system with a bandwidth of 3:

$$\begin{bmatrix} f_1 & g_1 & & & & & & \\ e_2 & f_2 & g_2 & & & & & \\ & e_3 & f_3 & g_3 & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & \cdot & \cdot & \cdot & \\ & & & & & e_{n-1} & f_{n-1} & g_{n-1} \\ & & & & & & e_n & f_n \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ \cdot \\ \cdot \\ \cdot \\ x_{n-1} \\ x_n \end{Bmatrix} = \begin{Bmatrix} r_1 \\ r_2 \\ r_3 \\ \cdot \\ \cdot \\ \cdot \\ r_{n-1} \\ r_n \end{Bmatrix}$$

- Tridiagonal systems can be solved using the same method as Gauss elimination, but with much less effort because most of the matrix elements are already 0.

Example 9.5

- Solve the following tridiagonal system

$$\begin{bmatrix} 2.04 & -1 & & \\ -1 & 2.04 & -1 & \\ & -1 & 2.04 & -1 \\ & & -1 & 2.04 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} 40.8 \\ 0.8 \\ 0.8 \\ 40.8 \end{Bmatrix}$$

- Forward Elimination of Unknowns

$$\begin{bmatrix} 2.04 & -1 & & \\ & 1.550 & -1 & \\ & & 1.395 & -1 \\ & & & 1.323 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} 40.8 \\ 20.8 \\ 14.221 \\ 50.996 \end{Bmatrix}$$

- Backward Substitution

$$x_4 = \frac{r_4}{f_4} = \frac{50.996}{1.323} = 38.545$$

$$x_3 = \frac{r_3 - g_3 x_4}{f_3} = \frac{14.221 - (-1)38.545}{1.395} = 37.832$$

$$x_2 = \frac{r_2 - g_2 x_3}{f_2} = \frac{20.800 - (-1)37.832}{1.550} = 37.832$$

$$x_1 = \frac{r_1 - g_1 x_2}{f_1} = \frac{40.800 - (-1)37.832}{2.040} = 38.545$$

Tridiagonal System Solver

```
function x = Tridiag(e,f,g,r)
% Tridiag: Tridiagonal equation solver banded system
%   x = Tridiag(e,f,g,r): Tridiagonal system solver.
% input:
%   e = subdiagonal vector
%   f = diagonal vector
%   g = superdiagonal vector
%   r = right hand side vector
% output:
%   x = solution vector
n=length(f);
% forward elimination
for k = 2:n
  factor = e(k)/f(k-1);
  f(k) = f(k) - factor*g(k-1);
  r(k) = r(k) - factor*r(k-1);
end
% back substitution
x(n) = r(n)/f(n);
for k = n-1:-1:1
  x(k) = (r(k)-g(k)*x(k+1))/f(k);
end
```

```
>>
>> e=[0; -1; -1; -1];
>> f=[2.04; 2.04; 2.04; 2.04];
>> g=[-1; -1; -1; 0];
>> r=[40.8; 0.8; 0.8; 40.8];
>> x = Tridiag(e, f, g, r)
    2.0400    1.5498    1.3948    1.3230
   40.8000   20.8000   14.2211   50.9961
x =
   38.5449   37.8317   37.8317   38.5449
```

Ready