

# Computer Programming

Dr. Deepak B Phatak  
Dr. Supratik Chakraborty  
Department of Computer Science and Engineering  
IIT Bombay

Session: Gaussian Elimination

# Quick Recap



- A system of linear algebraic equations in  $N$  variables can be represented by 3 matrices
  - An  $N \times N$  matrix of coefficients
  - An array of  $N$  variables
  - An array of corresponding RHS values
- Gaussian elimination technique
  - Reduces coefficient matrix to upper triangular form, making corresponding changes to the RHS array
  - Uses back-substitution to calculate values of all variables

# Simultaneous Equations ...

- In general, a system of linear equations in  $n$  variables can be represented by the following matrices

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & \dots & a_{0n-1} \\ a_{10} & a_{11} & a_{12} & \dots & a_{1n-1} \\ a_{20} & a_{21} & a_{22} & \dots & a_{2n-1} \\ \vdots & & & & \\ \vdots & & & & \\ a_{n0} & a_{n1} & a_{n2} & \dots & a_{n-1n-1} \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ \vdots \\ \vdots \\ b_{n-1} \end{pmatrix}$$

# Reduction to upper triangular form...

- The Gaussian elimination technique essentially reduces the coefficient matrix to an upper triangular form:

$$\begin{pmatrix} 1 & a_{01} & a_{02} & \dots & a_{0n-1} \\ 0 & 1 & a_{12} & \dots & a_{1n-1} \\ 0 & 0 & 1 & & . \\ & & & & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ . \\ . \\ x_{n-1} \end{pmatrix} = \begin{pmatrix} b_0 \\ b_1 \\ . \\ . \\ b_{n-1} \end{pmatrix}$$

# System in upper triangular form

- When the coefficient matrix is reduced to the upper triangular form, we have the following system of equations

$$x[0] + a[0][1] x[1] + a[0][2] x[2] + \dots + a[0][n-1] x[n-1] = b[0]$$

$$x[1] + a[1][2] x[2] + \dots + a[1][n-1] x[n-1] = b[1]$$

...

$$x[n-1] = b[n-1]$$

- Note that values of  $a[][]$  and  $b[]$  now, will be different from the original values
- Back substitution can be applied to calculate values of variables

# System of equations in 2 variables



- Consider  $2x + 4y = 8$

$$4x + 3y = 1$$

- Representing  $x$  by  $x[0]$  and  $y$  by  $x[1]$ , this can be represented as:

$$a[0][0] x[0] + a[0][1] x[1] = b[0]$$

$$a[1][0] x[0] + a[1][1] x[1] = b[1]$$

Where  $a[0][0]$  is 2,  $a[0][1]$  is 4,  $a[1][0]$  is 4,  $a[1][1]$  is 3,  $b[0]$  is 8,  $b[1]$  is 1

After reducing matrix  $a[][]$  to upper triangular form, the coefficients will be

$a[0][0]$  is 1,  $a[0][1]$  is 2,  $a[1][0]$  is 0,  $a[1][1]$  is -5,

The RHS array  $b[]$  will now be:  $b[0]$  is 4,  $b[1]$  is 15

# Program: gauss.cpp

---



```
#include<iostream>
using namespace std;
int main(){
    int i, j, k, n;
    float MatA[100][100], MatB[100], X[100];
    float Divisor, Factor, sum;
```

# gauss.cpp ...

```
cin >> n;
//reading matrix A
for(i=0; i< n; i++){
    for(j=0; j < n; j++){
        cin >> MatA[i][j];
    }
}
//reading matrix B
for(i=0; i< n; i++){
    cin >> MatB[i];
}
```



```
//Gauss elimination
for (i=0; i< n; i++){
    Divisor = MatA[i][i];
    MatA[i][i] = 1.0;
    // divide all values in the row by the divisor
    // to recalculate all coefficients in that row
    for (j = i+1; j < n; j++){
        MatA[i][j] = MatA[i][j]/Divisor;
    }
    //Also divide the corresponding RHS element
    MatB[i] = MatB[i]/Divisor;
```

```
// now replace subsequent rows, by subtracting the
// appropriate portion of the ith equation from it
if (i+1 < n) {
    for (k=i+1; k<n; k++){
        Factor = MatA[k][i];
        MatA[k][i] = 0.0;
        for (j = i+1; j < n; j++){
            MatA[k][j] = MatA[k][j] - Factor * MatA[i][j];
        }
        MatB[k] = MatB[k] - Factor * MatB[i];
    }
}
}
```

```
// back substitution starting with last variable
X[n-1] = MatB[n-1];
for (i = n-2; i >= 0; i--){
// Sum up ith row using values of X already determined
    sum = 0.0;
    for (j = i+1; j < n; j++){
        sum = sum + MatA[i][j] * X[j];
    }
    X[i] = MatB[i] - sum;
}
```

# gauss.cpp ...

```
//output the results
for(i=0;i< n;i++){
    for (j = 0; j < n; j++) {
        cout << MatA[i][j] << " ";
    }
    cout << " " << MatB[i] << endl;
}
for (i=0; i<n; i++){
    cout << "X[" << i << "]" is: " ;
    cout << X[i] << endl;
}
return 0;
}
```

# Sample input data ...

n 4

MatA[][]

2.0 1.0 3.0 -4.0

1.0 -2.0 -2.0 3.0

5.0 3.0 -1.0 -1.0

3.0 4.0 1.0 -2.0

MatB[]

-3.0 3.0 4.0 6.0

# Results

MatA[][] (Reduced to upper triangular form)

1	0.5	1.5	-2	-1.5
0	1	1.4	-2	-1.8
0	0	1	-1.08696	-1.34783
0	0	0	1	4

Values of the variables X[]

X[0] is: 1

X[1] is: 2

X[2] is: 3

X[3] is: 4

# Summary

---



- In this session,
  - We wrote a C++ program to implement the Gaussian elimination method for solving simultaneous equations
  - Saw sample input for a system of 4 equations, and the results
- The program is also available in the file `gauss.cpp`
  - Download, compile, and run it with sample data of your own