

FACULTY OF ENGINEERING
Computer and systems engineering department

JUNE 25, 2014

Numerical Analysis

Assignment One

PART TWO
PARTS ONE, TWO

PRESENTED BY:

Ebtehal El_Aaraby (3).
Amany Ibrahim Hassan (11).
Remon Hanna Wadie Youssef (23).
Sarah Moustafa Mohamed (25).
Shaimaa Mousa (28).
Tarek Khaled Ragab (32).

PART ONE

Objectives:

The aim of this assignment is to compare and analyze the behavior of numerical methods studied in class {Bisection, False-position, Fixed point, Newton-Raphson, Secant}.

Specification:

The program must contain the following features:

- An interactive GUI that enables the user to enter equations containing different functions such as: {poly, exp, cos, sin}. Reading from files must be available as well.
- Differentiation and Parsing is your task.
- A way to choose a method to solve the given equation.
- A way to enter the precision and the max number of iterations otherwise default values are used, Default Max Iterations = 50, Default Epsilon = 0.00001;
- The answer for the chosen method indicating the number of iterations, execution time, all iterations, approximate root, and precision.
- Compute the theoretical bound of the error for the methods.

Analysis and Description

NUMERICAL METHODS

Bisection Method:

Description:

It's a method to get the root of the given function by iterating to get x_r which is the average of the upper and the lower value and substitute in function if the result is 0 then it's the exact root of the equation else we calculate $f(X_l)$ and $f(X_r)$ if the multiplication less than 0 we change the upper value with x_r else we change the lower value with it.

Pseudo Code:

```
Get xlower and xupper  
//Check whether the root is one of them  
If(f(xlower)==0)  
    Root = xlower  
Else if(f(xupper)==0)  
    Root = xupper  
//Make a while loop by the given iteration or when the bound  
//of error less than the given epsilon  
While(iteration < maxIteration&&error > epsilon)  
    //get  $x_r$   
     $X_r = xlower+xupper/2$ 
```

```
//Check if xr is the exact solution  
If(f(xr)==0)  
Root = xr  
//If it isn't check whether f(xlower)*f(xr)<0 then xupper = xr  
Else if(f(xlower)*f(xr)<0 )  
xupper = xr  
//if f(xlower)*f(xr)>0 then xlower = xr  
Else if(f(xlower)*f(xr)>0 )  
xlower = xr  
End while.
```

The theoretical bound of error can be calculated using this formula $\text{ess} = L_0/(2^k * \text{xlower})$, where k is the max number of iteration.

Fixed Point :

Description:

For the given function we get $g(x)$ that is a function of x to get value of x that solve the equation so we iterate and substitute in $g(x)$ by the previous value of x until we get the exact value or the approximating value of the root of the equation.

Pseudo code:

```
//get g(x) and the initial value of x (xo)
//for loop while number of iteration < max number of iteration and error
//is greater than epsilon
While(iteration < maxIteration&&error > epsilon)
    //get xi
    Xi= g(xi-1);
    //Check if xi is the exact solution
    If(f(xi)==0)
        Root =xi
    End while
```

Newton Raphson

Description:

This method calculate the root of the equation by iterating and substituting in the formula $X_i = x(i-1) - f(x(i-1)) / f'(x(i-1))$ until we get the exact or the approximating value of the root.

Pseudo code:

```
//get f(x) and the initial value of x (xo)
//for loop while number of iteration < max number of iteration and error
//is greater than epsilon
While(iteration < maxIteration&&error > epsilon)
    //get xi
    Xi= x(i-1) - f(x(i-1))/ f'(x(i-1));
    //Check if xi is the exact solution
    If(f(xi)==0)
        Root =xi
    End while
```

Newton Raphson Modified 1:

```
//get f(x) ,the initial value of x (xo)and m  
//for loop while number of iteration < max number of iteration and  
error is greater than epsilon  
While(iteration <maxIteration&&error >epsilon)  
//get xi  
Xi= x(i-1) – m*f(x(i-1))/ f'(x(i-1));  
//Check if xi is the exact solution  
If(f(xi)==0)  
Root =xi  
End while
```

Newton Raphson Modified 2:

```
//get f(x) and the initial value of x (xo)  
//for loop while number of iteration < max number of iteration and  
error is greater than epsilon  
While(iteration < maxIteration&&error >epsilon)  
//get xi  
Xi= x(i-1) – f(x(i-1))* f'(x(i-1))/ f'(x(i-1))^2- f(x(i-1))* f'(x(i-1));  
//Check if xi is the exact solution  
If(f(xi)==0)  
Root =xi  
End while
```

Secant

Description:

It is open method to find the root of given equation by evaluating X_i at every time and compare it by the previous value of it and compute relative error to get the root.

It is preferred to newton raphson method because newton raphson method needs to compute the derivatives.

But secant is used the newton raphson equation to find approximate value to the root regarding the following equation :

$$x_{i+1} = x_i - \frac{f(x_i)(x_{i-1} - x_i)}{f(x_{i-1}) - f(x_i)}$$

Pseudo Code:

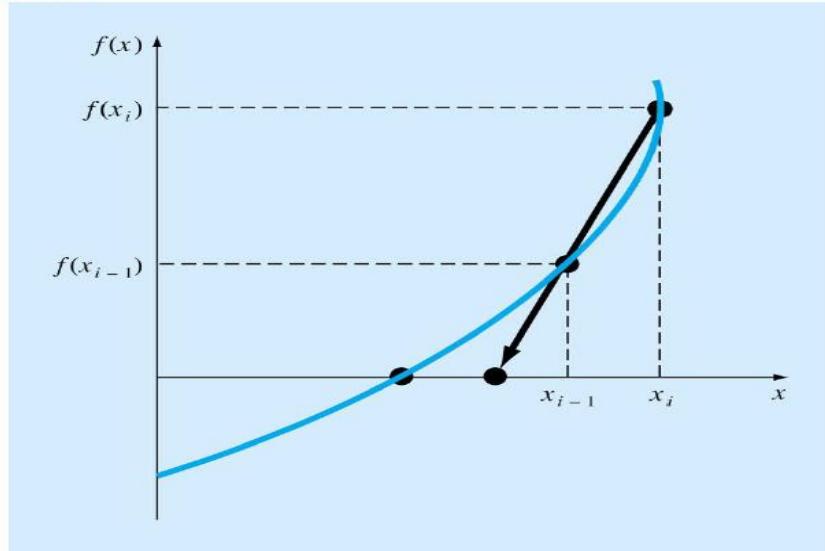
```
Secant( equation,MaxIter,epsilon,xl,xu )
output : arrayOfResult ,root
root =0;
curlter=0;
curEpsilon=100;
curtemp=100;
while(curEpsilon>epsilon )
if(curlter<MaxIter)
fl=compute(equation, xl);
fu=compute(equation, xu);
xDiff= xl - xu;
root = xu -((fu*xDiff)/(fl-fu));
curEpsilon=abs((root - xu)/ root);
curtemp=curEpsilon*100;
xl = xu;
```

```

xu = root;
curlter=curlter+1;
// add root & curEpsilon to the output array
fx=compute(equation, root);
end
end

```

Secant Method



Advantages of secant method :

- Fast (slower than Newton)
- One function evaluation per iteration
- No knowledge of derivative is needed

Disadvantages of secant :

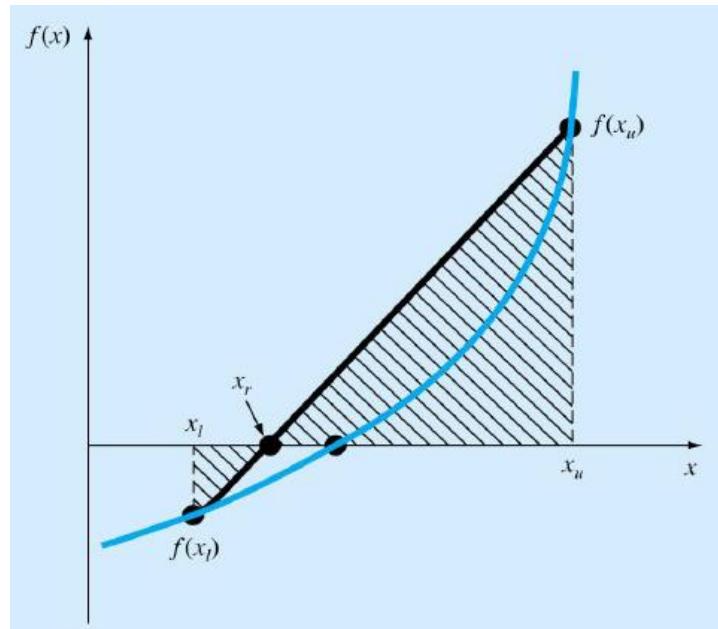
- May diverge
- Needs two initial points guess x_0, x_1 such that $f(x_0) - f(x_1)$ is nonzero

False Position:

Description:

It is bracketing method to find the root of given equation by Approximation the solution by doing a linear interpolation between $f(x_u)$ and $f(x_l)$

And find x_r in every iteration by the following rule :



$$x_r = \frac{x_l f_u - x_u f_l}{f_u - f_l}$$

```
If f(xr)<0 then xl=xr && fl=f(xr)
If f(xr)>0 then xu=xr && fu=f(xr)
If f(xr)=0, get the exact root !
```

Pseudo Code:

```
function [ arr,xr,time ] =
FalseMethod( equation,MaxIter,epsilon ,low,high)
xr=low;
curlter=0;
curEpsilon=100;
fLowValue =compute(equation,low);
fUpperValue =compute(equation,high);
if fUpperValue<0 &&fLowValue<0
    xr='Error in initial conditions';
return ;
else
while(curEpsilon>epsilon)
    if curlter< MaxIter
        xTemp=xr;
        xr=(low*fUpperValue-high*fLowValue)/(fUpperValue-
fLowValue);
        fxrValue =compute(equation,xr);
        curEpsilon=abs((xr-xTemp)/xr);
        curTemp= curEpsilon*100;
        curlter=curlter+1;
        // add xThird & error to the array of the output
    if fxrValue>0
        high=xr;
        fUpperValue=fxrValue;
    elseif fxrValue<0
        low=xr;
        fLowValue=fxrValue;
```

```
elseif fxrValue==0
    found the exact root
    return;
end
else
    return;
end
```

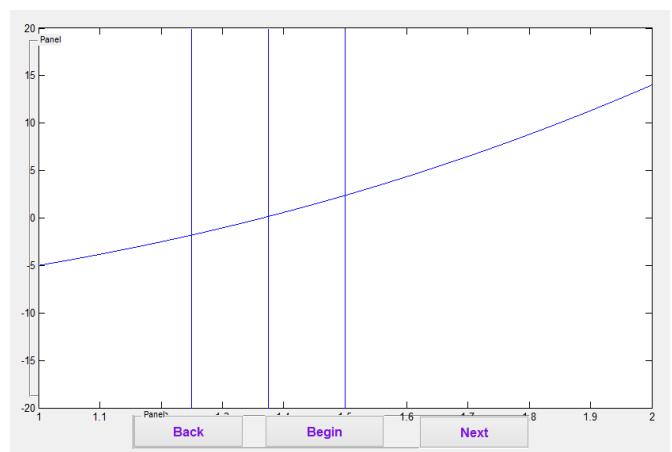
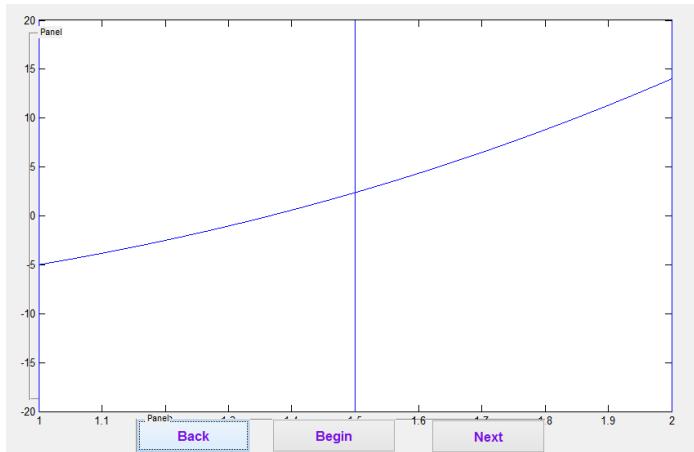
BONUS PART:

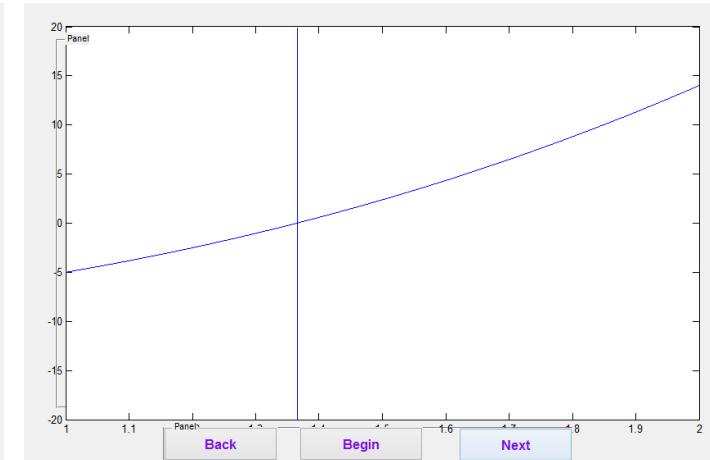
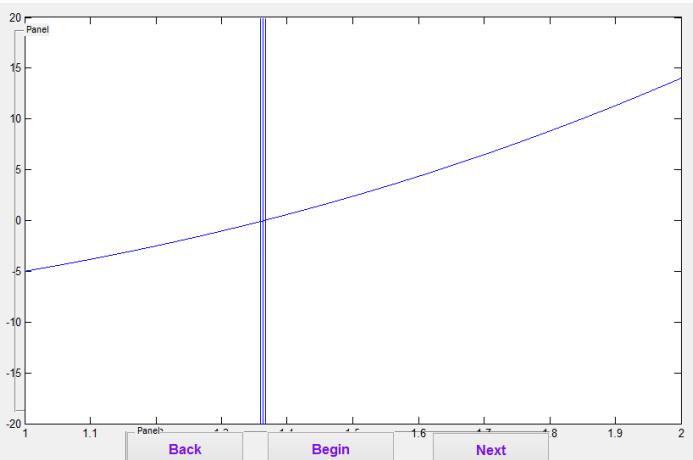
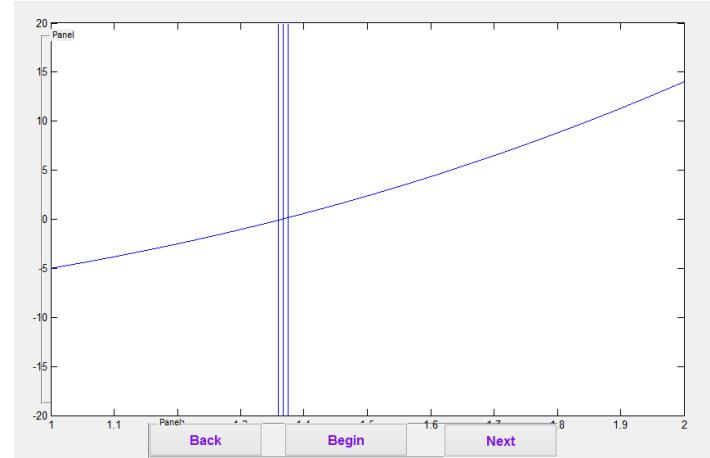
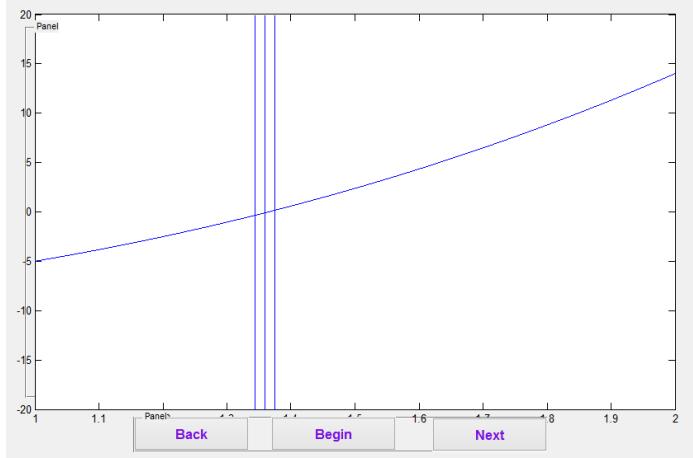
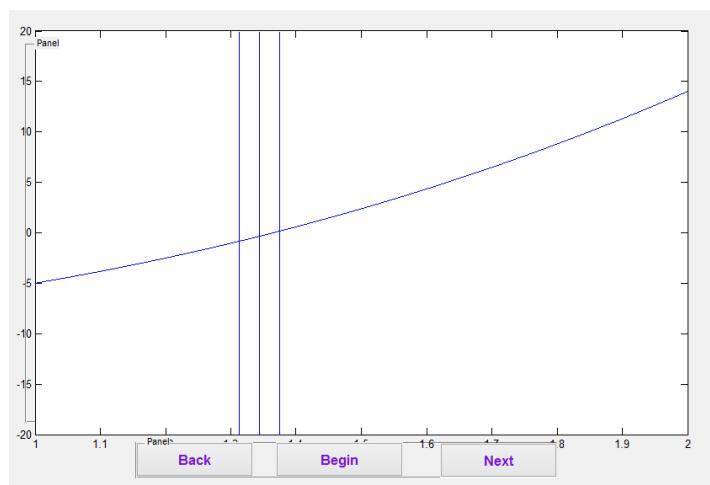
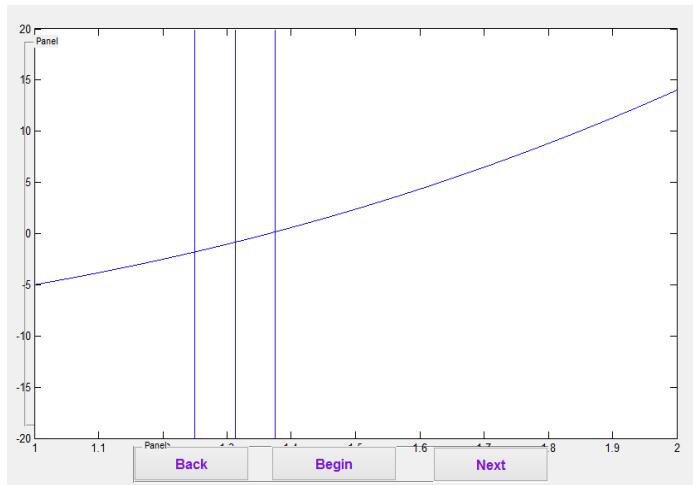
Description:

We plot the function for the Bisection method so when executing code of the Bisection method we save the result of each iteration for the values of x lower , x upper and xr then we plot every single step in that iteration and we give the user to undo or redo these steps.

Sample Runs:

For the given equation($(x)^3+4*(x)^2-10$) , $x_l = 1$ and $x_u = 2$ the output is :





Problematic functions:

For each method there are some cases that make the method diverge and never reach the exact solution or even an approximation to it for example:

- For Bisection method: if the given interval is not suitable it gives a root but it's not necessary to be true as for the equation " $(x)^2-2*(x)+1$ " the exact value is 1 it will get 1 if the interval from 0 to 2 but if we change it to any other interval it will diverge, the solution for it whether to choose a suitable interval or use the secant method which diverge less than the Bisection method.

Equation Solver

Enter The Equation Read From File

Button Group

Bisection
 False-position
 Fixed point
 Newton-Raphson
 Secant
 Newton Raphson M1
 Newton Raphson M2

bisection Theoretical Error Inf

Lower Upper

Iterations

Number of iteration	Xl	Xu	Xr	Ea%
1	0	10	5	100%
2	5	10	7.5	33.3333%
3	7.5	10	8.75	14.2857%
4	8.75	10	9.375	6.6667%
5	9.375	10	9.6875	3.2258%
6	9.6875	10	9.8438	1.5873%
7	9.8438	10	9.9219	0.7874%
8	9.9219	10	9.9600	0.30216%

Time of execution

Solve Maximum Number of iterations

Plot Solution Epsilon

Precision

Equation Solver

Enter The Equation

Bisection

False-position

Fixed point

Newton-Raphson

Secant

Newton Raphson M1

Newton Raphson M2

bisection

Theoretical Error

Lower

Upper

Iterations

Number of iteration	Xi	Xu	Xr	Ea%
1	0	2	1	100%

Time of execution

Solution

Maximum Number of iterations

Precision

Epsilon

Solve

Plot

- For Newton Raphson: because the formula of it is divided by $F'(x)$ if then it will lead to infinity and diverge as for " $(x)^2+2*(x)+1$ " , the solution is to use either Newton Raphson Modified 1 or 2 they converge faster and have a little chance to diverge.

Equation Solver

Enter The Equation

Bisection

False-position

Fixed point

Newton-Raphson

Secant

Newton Raphson M1

Newton Raphson M2

newtonRaphs

Xo

2

Iterations

Number of iteration	Xi	Ea%
1	NaN	NaN%

Time of execution

Solution

Maximum Number of iterations

Precision

Epsilon

Solve

Plot

- For Secant and False Position: it will get NaN when the dominator becomes 0 as for the example “ $(x)^3-2*(x)+x^2$ ” for false position and “ $(x)^2+4*(x)+4$ ”.

Equation Solver

Enter The Equation: $x^2+4*x+4$ | Read From File

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

Secant

Lower	-4	Upper	0	
Iterations				
Number Of Iteration	Xi-1	Xi	Xi+1	Ea

Time of execution: 0.00243362 | Maximum Number of iterations: 50

Solution: 0 | Epsilon: 0.00001

Precision: -3.29832

Solve | **Plot**

Equation Solver

Enter The Equation: $(x)^3-2*(x)+x^2$ | Read From File

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

falseposition

Lower	0	Upper	10	
Iterations				
NumOfIteration	XLow	XUpper	Xr	Ea
1	0	10	0	NaN%

Time of execution: 0.00678481 | Maximum Number of iterations: 50

Solution: 0 | Epsilon: 0.00001

Precision: NaN

Solve | **Plot**

Analysis the behavior of all methods by different examples:

Example 1:

Equation $x^3 - 0.165x^2 + 3.993 \times 10^{-4}$

In bisection:

The screenshot shows the software interface for solving the equation $x^3 - 0.165x^2 + 3.993 \times 10^{-4}$ using the Bisection method. The software is titled "part1_Gui".

Iterations Table:

Number of iteration	Xl	Xu	Xr	Ea%
1	0.11	1e-05	0.055005	100%
2	0.11	0.055005	0.082503	33.3293%
3	0.082503	0.055005	0.068754	19.9971%
4	0.068754	0.055005	0.061879	11.1093%
5	0.068754	0.061879	0.065317	5.2624%
6	0.065317	0.061879	0.063598	2.7023%
7	0.063598	0.061879	0.062739	1.3696%
8	0.062739	0.061879	0.062309	0.68054%

Execution Details:

- Time of execution: 0.0945547
- Solution: 0.0624164
- Precision: 3.06659
- Theoretical Error: 10.7412
- Maximum Number of iterations: 10
- Epsilon: 0.000001

Method Selection: Bisection

The screenshot shows the software interface for solving the equation $x^3 - 0.165x^2 + 3.993 \times 10^{-4}$ using the Bisection method. The software is titled "part1_Gui".

Iterations Table:

Number of iteration	Xl	Xu	Xr	Ea%
1	0.11	0.055005	0.082503	33.3293%
2	0.082503	0.055005	0.068754	19.9971%
3	0.068754	0.055005	0.061879	11.1093%
4	0.068754	0.061879	0.065317	5.2624%
5	0.065317	0.061879	0.063598	2.7023%
6	0.063598	0.061879	0.062739	1.3696%
7	0.062739	0.061879	0.062309	0.68054%
8	0.062739	0.062309	0.062524	0.34359%
9	0.062739	0.062309	0.062416	0.17209%
10	0.062524	0.062309	0.062416	0.000001

Execution Details:

- Time of execution: 0.0945547
- Solution: 0.0624164
- Precision: 3.06659
- Theoretical Error: 10.7412
- Maximum Number of iterations: 10
- Epsilon: 0.000001

Method Selection: Bisection

In False Position :

part1_Gui

Equation Solver

Enter The Equation Read From File

Button Group

Bisection
 False-position
 Fixed point
 Newton-Raphson
 Secant
 Newton Raphson M1
 Newton Raphson M2

falseposition

Lower Upper

Iterations

NumOfIteration	XLow	XUpper	Xr	Ea
1	1e-05	0.11	0.066004	99.9848%
2	0.066004	0.11	0.059997	10.012%
3	0.066004	0.059997	0.0624	3.8514%
4	0.0624	0.059997	0.062378	0.035755%
5	0.062378	0.059997	0.062378	0.00066736%
6	0.062378	0.059997	0.062377	0.00066725%
7	0.062378	0.062377	0.062377	0.00038801%
8	0.062378	0.062377	0.062378	0.00016238%

Time of execution

Solution Maximum Number of iterations

Precision Epsilon

Save **Plot**

part1_Gui

Equation Solver

Enter The Equation Read From File

Button Group

Bisection
 False-position
 Fixed point
 Newton-Raphson
 Secant
 Newton Raphson M1
 Newton Raphson M2

falseposition

Lower Upper

Iterations

NumOfIteration	XLow	XUpper	Xr	Ea
2	0.066004	0.11	0.059997	10.012%
3	0.066004	0.059997	0.0624	3.8514%
4	0.0624	0.059997	0.062378	0.035755%
5	0.062378	0.059997	0.062378	0.00066736%
6	0.062378	0.059997	0.062377	0.00066725%
7	0.062378	0.062377	0.062377	0.00038801%
8	0.062378	0.062377	0.062378	0.00016238%
9	0.062378	0.062377	0.062378	6.7953e-05%
10	0.062378	0.062377	0.062378	3.9516e-05%

Time of execution

Solution Maximum Number of iterations

Precision Epsilon

Save **Plot**

In Fixed Position:

part1_Gui

Equation Solver

Enter The Equation:

Fixed point

Xo: 1

Iterations

Number of iteration	Xi	Ea%
1	0.049193	100%
2	0.056049	100%
3	0.059052	12.2312%
4	0.060564	5.0854%
5	0.061371	2.4968%
6	0.061813	1.3141%
7	0.06206	0.71627%
8	0.062198	0.20658%

Time of execution: 0.213752

Solve

Plot

Maximum Number of iterations: 50

Solution: 0.0623773

Precision: Inf

Epsilon: 0.00000000001

part1_Gui

Equation Solver

Enter The Equation:

Fixed point

Xo: 1

Iterations

Iteration	Xi	Ea%
9	0.062276	0.22291%
10	0.06232	0.12494%
11	0.062345	0.070745%
12	0.062359	0.039948%
13	0.062367	0.02271%
14	0.062372	0.012722%
15	0.062374	0.007271%
16	0.062376	0.0045448%
17	0.062377	0.001819%

Time of execution: 0.213752

Solve

Plot

Maximum Number of iterations: 50

Solution: 0.0623773

Precision: Inf

Epsilon: 0.00000000001

part1_Gui

Equation Solver

Enter The Equation:

fixedpoint

Xo: **1**:

Iterations

	12	0.062359	0.039948%
-	13	0.062367	0.02271%
-	14	0.062372	0.012722%
-	15	0.062374	0.007271%
-	16	0.062376	0.0045448%
-	17	0.062377	0.001818%
-	18	0.062377	0.0018181%
-	19	0.062377	0.00090907%

Time of execution:

Solution:

Maximum Number of iterations:

Precision:

Epsilon:

In Newton Raphson :

part1_Gui

Equation Solver

Enter The Equation:

newtonRaphs

Xo: **0.1**:

Iterations

Number of iteration	Xi	Ea%
1	12.1111	99.9992%
2	8.0925	49.6573%
3	5.4136	49.4848%
4	3.6278	49.2263%
5	2.4374	48.8374%
6	1.6441	48.25%
7	1.1157	47.3644%
8	0.76401	46.03%

Time of execution:

Solution:

Maximum Number of iterations:

Precision:

Epsilon:

Equation Solver

Enter The Equation

newtonRaphs

Xo M

Iterations

11	0.27537	36.6114%
12	0.21118	30.3927%
13	0.17283	22.1944%
14	0.15339	12.6713%
15	0.14707	4.2953%
16	0.14637	0.4796%
17	0.14636	0.0071696%
18	0.14636	0.00033841%

Time of execution

Solution

Maximum Number of iterations

Precision

Epsilon

Solve **Plot**

In Newton Raphson Modified 1 :

part1_Gui

Equation Solver

Enter The Equation

**NewtonRaph
conM1**

Xo M

Iterations

Number of iteration	Xi	Ea%
1	0.053012	5759.1105%
2	0.08101	34.5613%
3	0.016332	396.0153%
4	0.25145	93.5048%
5	0.086546	190.5409%
6	-0.0062494	1484.8654%
7	-0.54662	98.8567%
8	0.044662	1323.0017%

Time of execution

Solution

Maximum Number of iterations

Precision

Epsilon

Solve **Plot**

part1_Gui

Equation Solver

Enter The Equation: $x^3-0.165*x^2+3.993*10^{-4}$

NewtonRaphsonM1

X₀: -3 M: 3

Iterations

	X _i	f(X _i)	% Error
9	0.09924	54.9959%	
10	-0.13333	174.432%	
11	0.01783	847.7973%	
12	0.23233	92.3258%	
13	0.09041	156.9773%	
14	-0.028382	418.5475%	
15	-0.090391	68.6009%	
16	0.0027662	3367.6631%	
17	1.3429	0.704%	

Time of execution: 0.619957 Maximum Number of iterations: 40

Solution: 0.0635064 Epsilon: 0.00001

Precision: -5.68937

part1_Gui

Equation Solver

Enter The Equation: $x^3-0.165*x^2+3.993*10^{-4}$

NewtonRaphsonM1

X₀: -3 M: 3

Iterations

	X _i	f(X _i)	% Error
17	1.3429	99.794%	
18	0.059676	2150.3262%	
19	0.06774	11.9046%	
20	0.051323	31.9882%	
21	0.084486	39.2527%	
22	0.003107	2619.2238%	
23	1.1996	99.741%	
24	0.060267	1890.4079%	
25	0.066573	0.1711%	

Time of execution: 0.619957 Maximum Number of iterations: 40

Solution: 0.0635064 Epsilon: 0.00001

Precision: -5.68937

part1_Gui

Equation Solver

Enter The Equation: $x^3 - 0.165x^2 + 3.993 \times 10^{-4}$

NewtonRaph sonM1

Xo: -3 **M:** 3

Iterations

-	25	0.066573	9.4711%
-	26	0.053801	23.739%
-	27	0.079409	32.2488%
-	28	0.021604	267.5718%
-	29	0.19567	88.9592%
-	30	0.1018	92.2099%
-	31	-0.20446	149.7896%
-	32	0.029547	791.9821%
-	33	0.14777	80.0042%

Time of execution: 0.619957

Solution: 0.0635064

Maximum Number of iterations: 40

Precision: -5.68937

Epsilon: 0.00001

Solve **Plot**

In Newton Raphson Modified 2:

part1_Gui

Equation Solver

Enter The Equation: $x^3 - 0.165x^2 + 3.993 \times 10^{-4}$

NewtonRaph sonM2

Xo: -3 **M:** 3

Iterations

Number of iteration	Xi	Ea%
1	0.051018	5980.2879%
2	0.062054	17.7846%
3	0.062378	0.51913%
4	0.062377	0.0006709%

Time of execution: 0.117882

Solution: 0.0623773

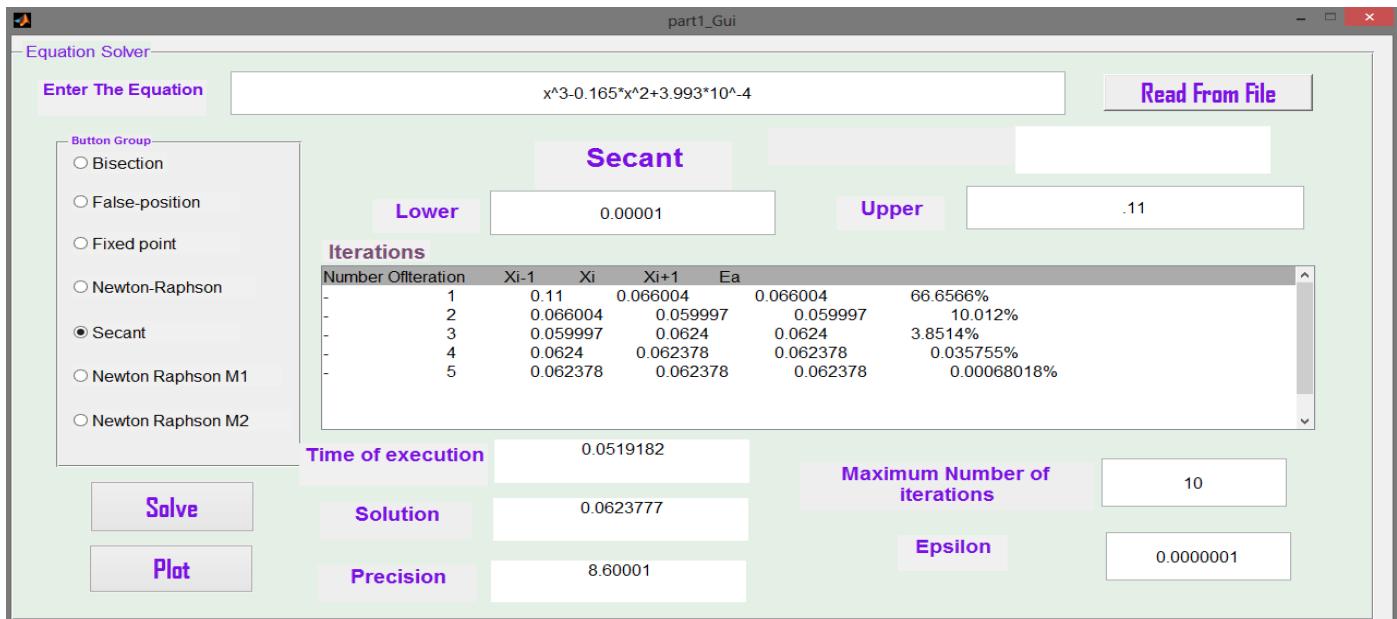
Maximum Number of iterations: 50

Precision: 8.61374

Epsilon: 0.00001

Solve **Plot**

In secant method :



Method	Time	Solution	Precision	Iterations	relativeError
Bisection	0.094557	0.0624164	3.06659	10	0.17209%
Falseposition	0.0413807	0.0623775	11.4457	10	3.9516e-05%
Fixedpoint	0.213752	0.0623773	inf	19	0.0009006 %
Newton	0.189097	3.20212	-2.58	10	49.1208%
Newton 1	0.619957	0.0635064	-5.689	40	40.5%
Newton 2	0.117882	0.0623773	8.61374	4	0.000679%
Secant	0.0519182	0.062377	8.60001	5	0.000681%

We can see that false Position take the least time of execution and it got the accurate root, Newton Raphson Modified 2 takes the least number of iteration and precisions are almost equal in every method so we can say that false position is the accurate and the fastest method in this example.

Example 2:

Equation : $(x)^3+4*x^2-10$

In bisection :

part1_Gui

Equation Solver

Enter The Equation: $(x)^3+4*x^2-10$ Read From File

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

bisection

Theoretical Error: 0.000976563

Lower	1	Upper	2
-------	---	-------	---

Iterations

Number of iteration	Xl	Xu	Xr	Ea%
1	2	1	1.5	100%
2	1.5	1	1.25	20%
3	1.5	1.25	1.375	9.0909%
4	1.375	1.25	1.3125	4.7619%
5	1.375	1.3125	1.3438	2.3256%
6	1.375	1.3438	1.3594	1.1494%
7	1.375	1.3594	1.3672	0.57143%
8	1.3672	1.3594	1.3633	0.28653%

Time of execution: 0.081056

Maximum Number of iterations: 10

Solution: 1.36621

Precision: 3.9452

Epsilon: 0.00001

Solve

Plot

part1_Gui

Equation Solver

Enter The Equation **Read From File**

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

bisection

	Lower	Upper	Theoretical Error		
	1	2	0.000976563		
Iterations					
1	2	1.5	1	1.25	
2	3	1.5	1.25	1.375	9.0909%
3	4	1.375	1.25	1.3125	4.7619%
4	5	1.375	1.3125	1.3438	2.3256%
5	6	1.375	1.3438	1.3594	1.1494%
6	7	1.375	1.3594	1.3672	0.57143%
7	8	1.3672	1.3594	1.3633	0.28653%
8	9	1.3672	1.3633	1.3652	0.14306%
9	10	1.3672	1.3652	1.3662	0.07148%

Time of execution 0.081056

Solution 1.36621

Precision 3.9452

Maximum Number of iterations 10

Epsilon 0.00001

Plot **Solve**

In falseposition :

part1_Gui

Equation Solver

Enter The Equation **Read From File**

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

falseposition

	Lower	Upper		
	1	2		
Iterations				
1	XLow	XUpper	Xr	Ea
2	1	2	1.2632	20.8333%
3	1.2632	2	1.3388	5.6501%
4	1.3388	2	1.3585	1.453%
5	1.3585	2	1.3636	0.36928%
6	1.3636	2	1.3648	0.089412%
7	1.3648	2	1.3651	0.023584%
8	1.3651	2	1.3652	0.0071301%
9	1.3652	2	1.3652	0.001646%

Time of execution 0.0295073

Solution 1.36524

Precision 8.07317

Maximum Number of iterations 10

Epsilon 0.00001

Plot **Solve**

part1_Gui

Equation Solver

Enter The Equation: $(x)^{(3)}+4*x^{(2)}-10$

falseposition

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

	Lower	Upper
1	2	

Iterations

	Z	1.2632	Z	1.3388	5.6501%
3	1.3388	2	1.3585	1.453%	
4	1.3585	2	1.3636	0.36928%	
5	1.3636	2	1.3648	0.089412%	
6	1.3648	2	1.3651	0.023584%	
7	1.3651	2	1.3652	0.0071301%	
8	1.3652	2	1.3652	0.001646%	
9	1.3652	2	1.3653	0.0016459%	
10	1.3652	1.3653	1.3652	0.0011519%	

Time of execution: 0.0295073

Solution: 1.36524

Maximum Number of iterations: 10

Precision: 8.07317

Epsilon: 0.00001

Solve

Plot

In fixedpoint :

part1_Gui

Equation Solver

Enter The Equation: $\sqrt{((10-x^3)/9)}$

fixedpoint

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

X ₀	1	2
----------------	---	---

Iterations

Number of iteration	X _i	Ea%
1	1	100%

Time of execution: 0.00503147

Solution: 1

Maximum Number of iterations: 10

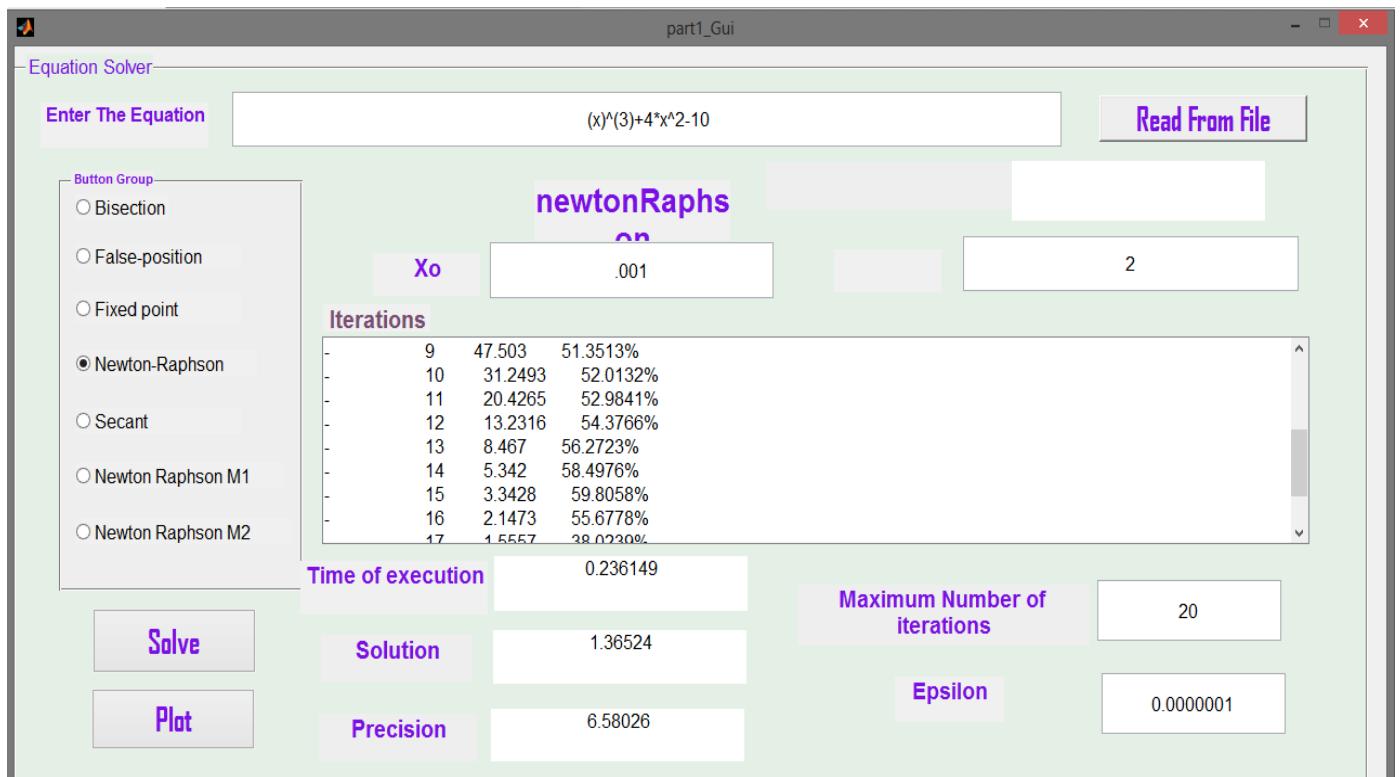
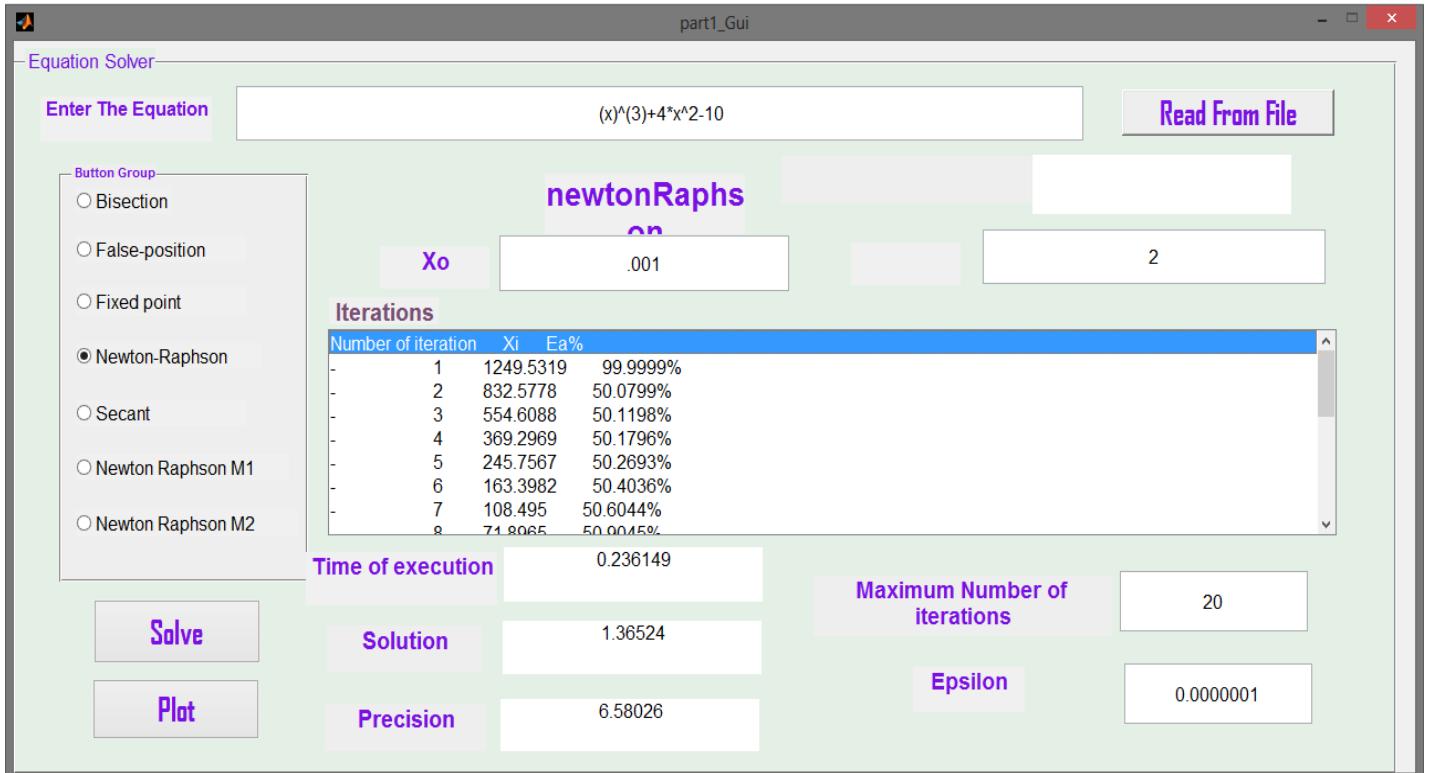
Precision: Inf

Epsilon: 0.0000000001

Solve

Plot

In newtonraphson :



part1_Gui

Equation Solver

Enter The Equation Read From File

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

Iterations

	TZ	Xi	Ea%
-	12	13.2316	54.3766%
-	13	8.467	56.2723%
-	14	5.342	58.4976%
-	15	3.3428	59.8058%
-	16	2.1473	55.6778%
-	17	1.5557	38.0239%
-	18	1.3809	12.6634%
-	19	1.3653	1.139%
-	20	1.3652	0.0051261%

Xo M

Time of execution Maximum Number of iterations

Solution Epsilon

Plot Precision

NewtonRaphson

In newtonraphon 1:

part1_Gui

Equation Solver

Enter The Equation Read From File

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

Iterations

Number of iteration	Xi	Ea%
1	24999.0625	100%
2	8332.1321	200.032%
3	2776.4887	200.096%
4	924.6082	200.2881%
5	307.3164	200.8652%
6	101.5577	202.6029%
7	32.987	207.8716%
8	10.170	221.0700%

Xo M

Time of execution Maximum Number of iterations

Solution Epsilon

Plot Precision

NewtonRaphsonM1

part1_Gui

Equation Solver

Enter The Equation

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM1

X ₀	0.0001	M	2
----------------	--------	---	---

Iterations

	Iteration	X _i	% Error
-	9	2.7396	271.5522%
-	10	0.91285	200.1122%
-	11	2.1177	56.8937%
-	12	0.97042	118.2233%
-	13	1.9752	50.8701%
-	14	1.0073	96.0922%
-	15	1.8934	46.8011%
-	16	1.0342	83.0776%
-	17	1.8382	43.7371%

Time of execution: 0.234291

Solution: 1.07211

Maximum Number of iterations: 20

Precision: -2.90752

Epsilon: 0.0000001

part1_Gui

Equation Solver

Enter The Equation

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM1

X ₀	0.0001	M	2
----------------	--------	---	---

Iterations

	Iteration	X _i	% Error
-	12	0.97042	118.2233%
-	13	1.9752	50.8701%
-	14	1.0073	96.0922%
-	15	1.8934	46.8011%
-	16	1.0342	83.0776%
-	17	1.8382	43.7371%
-	18	1.0551	74.2216%
-	19	1.7974	41.2991%
-	20	1.0721	67.6515%

Time of execution: 0.234291

Solution: 1.07211

Maximum Number of iterations: 20

Precision: -2.90752

Epsilon: 0.0000001

In newtonraphson 2:

part1_Gui

Equation Solver

Enter The Equation: $(x)^3+4*x^2-10$

NewtonRaph sonM2

Xo: 0.0001

Iterations

Number of iteration	Xi	Ea%
1	0.0002	49.9991%
2	0.00039998	49.9986%
3	0.00079992	49.9974%
4	0.0015997	49.995%
5	0.0031987	49.9899%
6	0.0063949	49.9799%
7	0.012769	49.9205%
8	0.025477	40.870%

Time of execution: 0.40689

Solution: 1.36519

Plot

part1_Gui

Equation Solver

Enter The Equation: $(x)^3+4*x^2-10$

NewtonRaph sonM2

Xo: 0.0001

Iterations

Number of iteration	Xi	Ea%
9	0.050705	49.7542%
10	0.10038	49.4885%
11	0.19647	48.9067%
12	0.3747	47.5664%
13	0.67216	44.2537%
14	1.0509	36.0375%
15	1.3092	19.7335%
16	1.3637	3.9928%
17	1.3652	0.11100%

Time of execution: 0.40689

Solution: 1.36519

Plot

Equation Solver

Enter The Equation: $(x)^{(3)}+4*x^2-10$

NewtonRaph sonM2

X₀: 0.0001 Iterations: 2

Iteration	X _i	E _a
12	0.3747	47.5064%
13	0.67216	44.2537%
14	1.0509	36.0375%
15	1.3092	19.7335%
16	1.3637	3.9928%
17	1.3652	0.11199%
18	1.3652	0.0021984%
19	1.3653	0.0021983%
20	1.3652	0.0051267%

Time of execution: 0.40689 Maximum Number of iterations: 20

Solution: 1.36519 Epsilon: 0.0000001

Precision: 6.58014

Solve **Plot**

In secant :

Equation Solver

Enter The Equation: $(x)^{(3)}+4*x^2-10$

Secant

Lower: .001 Upper: 2

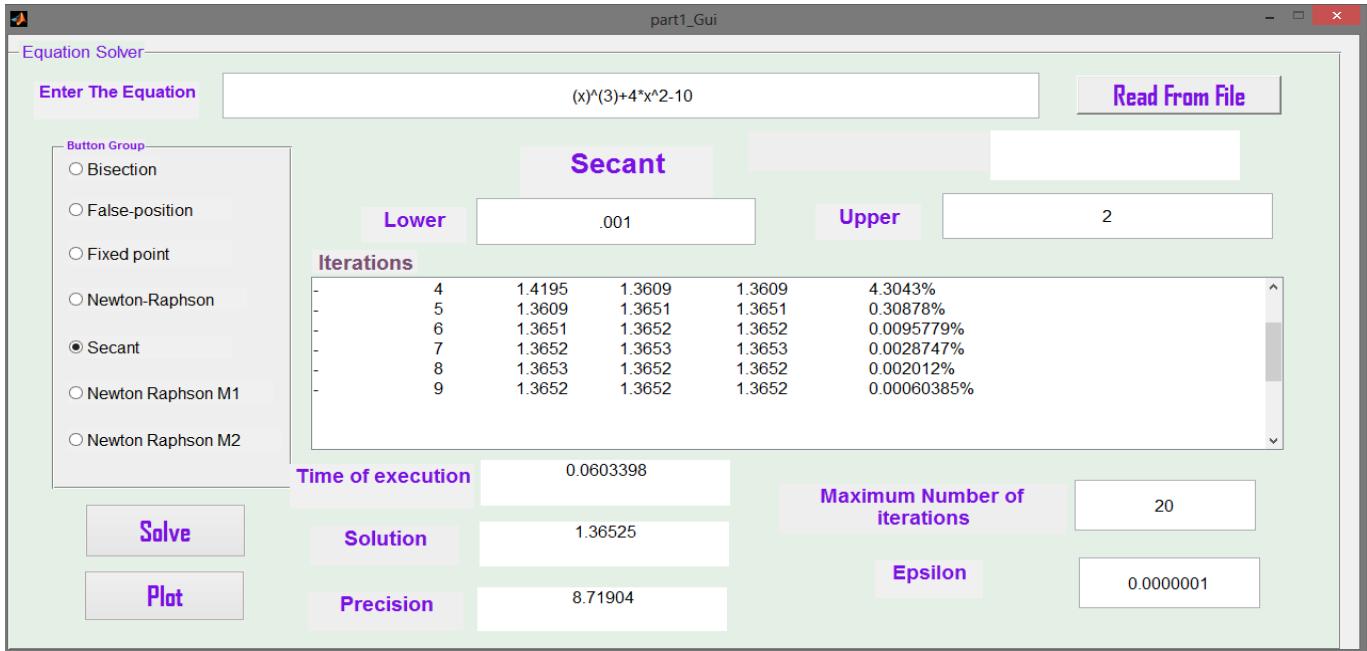
Iteration	X _{i-1}	X _i	X _{i+1}	E _a
1	2	0.83392	0.83392	139.8322%
2	0.83392	1.209	1.209	31.0237%
3	1.209	1.4195	1.4195	14.8273%
4	1.4195	1.3609	1.3609	4.3043%
5	1.3609	1.3651	1.3651	0.30878%
6	1.3651	1.3652	1.3652	0.0095779%
7	1.3652	1.3653	1.3653	0.0028747%
8	1.3653	1.3652	1.3652	0.0002012%

Time of execution: 0.0603398 Maximum Number of iterations: 20

Solution: 1.36525 Epsilon: 0.0000001

Precision: 8.71904

Solve **Plot**



Method	Time	Solution	Precision	Iterations	Relativeerror
Bisection	0.081056	1.36621	3.9452	10	0.07148%
Falseposition	0.0295073	1.36524	8.07317	10	0.001519%
Fixedpoint	0.0050317	1	inf	1	100%
Newton	0.236149	1.36245	6.58026	20	0.0051216%
Newton 1	0.234291	1.07211	-2.9705	20	67.6511%
Newton2	0.40689	1.36159	6.58014	20	0.0051267%
Secant	0.0603398	1.36525	8.71904	9	0.00060358%

We can see that **Secant** take the least time of execution ,it got the accurate root and the least number of and the least precision is in **bisection** method so we can say that **Secant** is the accurate and the fastest method in this example.

Example 3:

$$\text{Equation} = e^{-x} - x$$

- In Bisection

part1_Gui

Equation Solver

Enter The Equation: Read From File

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

bisection Theoretical Error: 0.00381279

Iterations

Number of iteration	Xl	Xu	Xr	Ea%
1	2	0.001	1.0005	100%
2	1.0005	0.001	0.50075	99.8003%
3	1.0005	0.50075	0.75062	33.2889%
4	0.75062	0.50075	0.62569	19.968%
5	0.62569	0.50075	0.56322	11.0914%
6	0.62569	0.56322	0.59445	5.2543%
7	0.59445	0.56322	0.57884	2.698%
8	0.57884	0.56322	0.57103	1.3675%

Time of execution: 0.112694 Maximum Number of iterations: 25

Solve Solution: 0.567142 Epsilon: 0.00001

Plot Precision: 8.61169

part1_Gui

Equation Solver

Enter The Equation: Read From File

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

bisection Theoretical Error: 0.00381279

Iterations

Number of iteration	Xl	Xu	Xr	Ea%
9	0.57103	0.56322	0.56712	0.68844%
10	0.57103	0.56712	0.56908	0.34304%
11	0.56908	0.56712	0.5681	0.17181%
12	0.5681	0.56712	0.56761	0.085981%
13	0.56761	0.56712	0.56737	0.043009%
14	0.56737	0.56712	0.56725	0.021509%
15	0.56725	0.56712	0.56718	0.010756%
16	0.56718	0.56712	0.56715	0.0053781%
17	0.56715	0.56712	0.56711	0.0026801%

Time of execution: 0.112694 Maximum Number of iterations: 25

Solve Solution: 0.567142 Epsilon: 0.00001

Plot Precision: 8.61169

Equation Solver

Enter The Equation:

bisection

	Lower	.001	Upper	2	
Iterations	13	0.56761	0.56712	0.56737	0.043009%
	14	0.56737	0.56712	0.56725	0.021509%
	15	0.56725	0.56712	0.56718	0.010756%
	16	0.56718	0.56712	0.56715	0.0053781%
	17	0.56715	0.56712	0.56714	0.0026891%
	18	0.56715	0.56714	0.56715	0.0013446%
	19	0.56715	0.56714	0.56714	0.00067228%

Time of execution: 0.112694

Solution: 0.567142

Precision: 8.61169

Theoretical Error: 0.00381279

Maximum Number of iterations: 25

Epsilon: 0.00001

- In False Position

Equation Solver

Enter The Equation:

falseposition

	Lower	.001	Upper	2	
Iterations	NumOfIteration	XLow	XUpper	Xr	Ea
	1	0.001	2	0.6979	99.8567%
	2	0.6979	2	0.54123	28.9484%
	3	0.6979	0.54123	0.56775	4.6707%
	4	0.56775	0.54123	0.56714	0.10646%
	5	0.56775	0.56714	0.56714	0.00057434%

Time of execution: 0.0116607

Solution: 0.567145

Precision: 8.76914

Maximum Number of iterations: 25

Epsilon: 0.00001

- In Fixed point

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

fixedpoint

X ₀	.001	2
Iterations		
Number of iteration	X _i	Ea%
1	0.999	100%
2	0.36825	99.8999%
3	0.69194	171.2851%
4	0.5006	46.7807%
5	0.60617	38.2219%
6	0.54544	17.4148%
7	0.57959	11.1344%
8	0.56013	5.8022%

Time of execution: 0.092562

Solution: 0.567145

Maximum Number of iterations: 25

Precision: 8.21461

Epsilon: 0.00001

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

fixedpoint

X ₀	.001	2
Iterations		
Number of iteration	X _i	Ea%
9	0.57113	3.474%
10	0.56489	1.9272%
11	0.56842	1.1061%
12	0.56642	0.62206%
13	0.56755	0.35362%
14	0.56691	0.1998%
15	0.56728	0.11306%
16	0.56707	0.06398%
17	0.56718	0.027007%

Time of execution: 0.092562

Solution: 0.567145

Maximum Number of iterations: 25

Precision: 8.21461

Epsilon: 0.00001

part1_Gui

Equation Solver

Enter The Equation:

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

fixedpoint

X₀: 2

Iterations

-	16	0.56707	0.06398%
-	17	0.56718	0.037007%
-	18	0.56712	0.020998%
-	19	0.56716	0.011001%
-	20	0.56713	0.0059998%
-	21	0.56715	0.0040001%
-	22	0.56714	0.003%
-	23	0.56715	0.002%

Time of execution:

Solution: Maximum Number of iterations:

Precision: Epsilon:

- In Newton Raphson

part1_Gui

Equation Solver

Enter The Equation:

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

newtonRaphs

X₀: 2

Iterations

Number of iteration	X _i	Ea%
1	0.50028	99.8001%
2	0.56632	11.6614%
3	0.56715	0.14515%
4	0.56714	0.0011831%
5	0.56714	0.00058019%

Time of execution:

Solution: Maximum Number of iterations:

Precision: Epsilon:

- In Newton Raphson Modified 1

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM1

X₀: -2 M: 1

Iterations

Number of iteration	X _i	Ea%
1	-0.88067	127.1004%
2	0.084375	1143.7571%
3	0.51935	83.7539%
4	0.56673	8.3597%
5	0.56714	0.07287%
6	0.56715	0.00058019%
7	0.56714	0.0011831%
8	0.56714	0.00058010%

Time of execution: 0.200625

Solution: 0.56714

Maximum Number of iterations: 25

Precision: 8.04647

Epsilon: 0.0000000001

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM1

X₀: -2 M: 1

Iterations

Number of iteration	X _i	Ea%
9	0.56715	0.00058019%
10	0.56714	0.0011831%
11	0.56714	0.00058019%
12	0.56715	0.00058019%
13	0.56714	0.0011831%
14	0.56714	0.00058019%
15	0.56715	0.00058019%
16	0.56714	0.0011831%
17	0.56714	0.00058010%

Time of execution: 0.200625

Solution: 0.56714

Maximum Number of iterations: 25

Precision: 8.04647

Epsilon: 0.0000000001

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM1

X₀: M:

Iterations:

	X _i	E _a %
17	0.56714	0.00058019%
18	0.56715	0.00058019%
19	0.56714	0.0011831%
20	0.56714	0.00058019%
21	0.56715	0.00058019%
22	0.56714	0.0011831%
23	0.56714	0.00058019%
24	0.56715	0.00058019%
25	0.56714	0.0011831%

Time of execution: 0.200625

Solution: 0.56714 Maximum Number of iterations: 25

Precision: 8.04647 Epsilon: 0.0000000001

- In Newton Raphson Modified 2

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM2

X₀: M:

Iterations:

Number of iteration	X _i	E _a %
1	0.66629	99.8499%
2	0.56875	17.1494%
3	0.56714	0.28323%
4	0.56715	0.00058019%
5	0.56714	0.0011831%
6	0.56714	0.00058019%
7	0.56715	0.00058019%
8	0.56714	0.0011831%

Time of execution: 0.455799

Solution: 0.567145 Maximum Number of iterations: 25

Precision: 8.759 Epsilon: 0.00000001

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM2

X ₀	.001	2																											
Iterations	<table border="1"> <tr><td>9</td><td>0.56714</td><td>0.0005802%</td></tr> <tr><td>10</td><td>0.56715</td><td>0.00058019%</td></tr> <tr><td>11</td><td>0.56714</td><td>0.0011831%</td></tr> <tr><td>12</td><td>0.56714</td><td>0.0005802%</td></tr> <tr><td>13</td><td>0.56715</td><td>0.00058019%</td></tr> <tr><td>14</td><td>0.56714</td><td>0.0011831%</td></tr> <tr><td>15</td><td>0.56714</td><td>0.0005802%</td></tr> <tr><td>16</td><td>0.56715</td><td>0.00058019%</td></tr> <tr><td>17</td><td>0.56714</td><td>0.0011831%</td></tr> </table>		9	0.56714	0.0005802%	10	0.56715	0.00058019%	11	0.56714	0.0011831%	12	0.56714	0.0005802%	13	0.56715	0.00058019%	14	0.56714	0.0011831%	15	0.56714	0.0005802%	16	0.56715	0.00058019%	17	0.56714	0.0011831%
9	0.56714	0.0005802%																											
10	0.56715	0.00058019%																											
11	0.56714	0.0011831%																											
12	0.56714	0.0005802%																											
13	0.56715	0.00058019%																											
14	0.56714	0.0011831%																											
15	0.56714	0.0005802%																											
16	0.56715	0.00058019%																											
17	0.56714	0.0011831%																											

Time of execution	0.455799	
Solution	0.567145	
Precision	8.759	
Maximum Number of iterations	25	
Epsilon	0.0000001	

part1_Gui

Equation Solver

Enter The Equation:

Button Group:

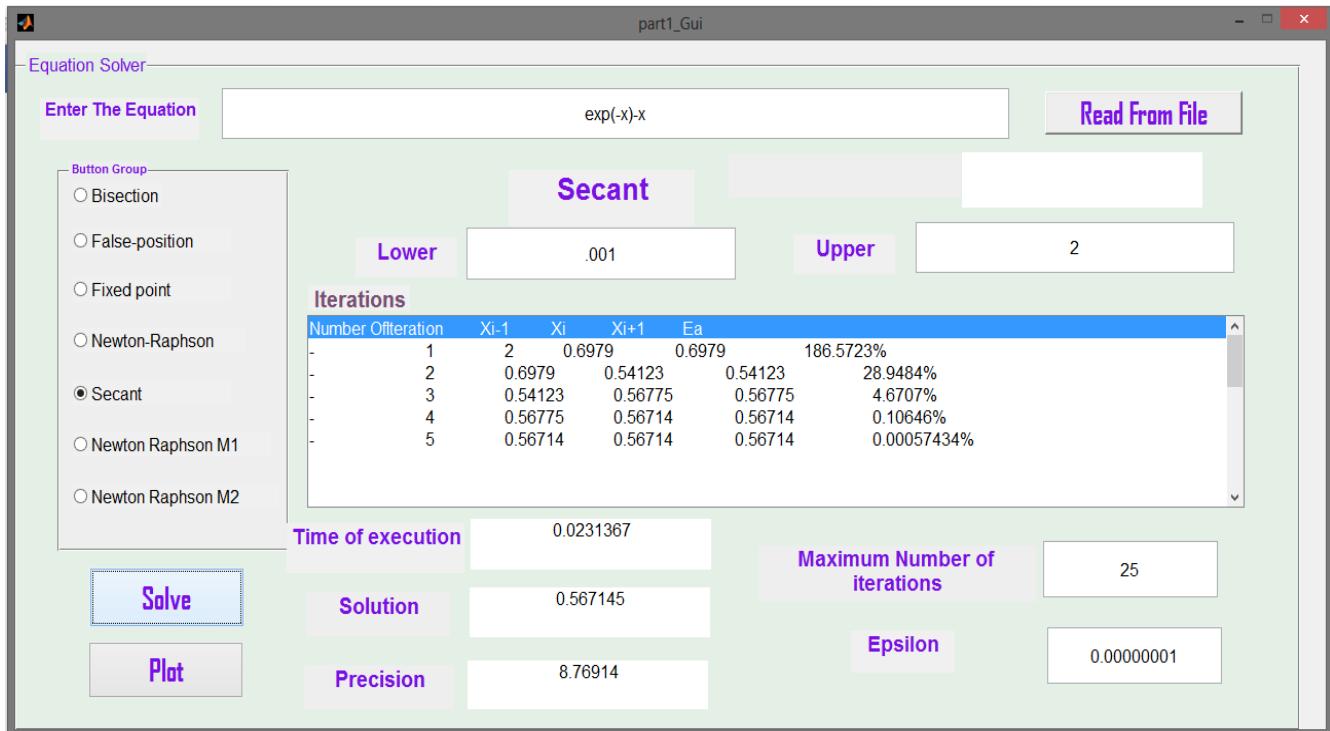
- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

NewtonRaph sonM2

X ₀	.001	2																											
Iterations	<table border="1"> <tr><td>17</td><td>0.56714</td><td>0.0011831%</td></tr> <tr><td>18</td><td>0.56714</td><td>0.0005802%</td></tr> <tr><td>19</td><td>0.56715</td><td>0.00058019%</td></tr> <tr><td>20</td><td>0.56714</td><td>0.0011831%</td></tr> <tr><td>21</td><td>0.56714</td><td>0.0005802%</td></tr> <tr><td>22</td><td>0.56715</td><td>0.00058019%</td></tr> <tr><td>23</td><td>0.56714</td><td>0.0011831%</td></tr> <tr><td>24</td><td>0.56714</td><td>0.0005802%</td></tr> <tr><td>25</td><td>0.56714</td><td>0.00058019%</td></tr> </table>		17	0.56714	0.0011831%	18	0.56714	0.0005802%	19	0.56715	0.00058019%	20	0.56714	0.0011831%	21	0.56714	0.0005802%	22	0.56715	0.00058019%	23	0.56714	0.0011831%	24	0.56714	0.0005802%	25	0.56714	0.00058019%
17	0.56714	0.0011831%																											
18	0.56714	0.0005802%																											
19	0.56715	0.00058019%																											
20	0.56714	0.0011831%																											
21	0.56714	0.0005802%																											
22	0.56715	0.00058019%																											
23	0.56714	0.0011831%																											
24	0.56714	0.0005802%																											
25	0.56714	0.00058019%																											

Time of execution	0.455799	
Solution	0.567145	
Precision	8.759	
Maximum Number of iterations	25	
Epsilon	0.0000001	

- In Secant



Comparison and analysis of all methods:

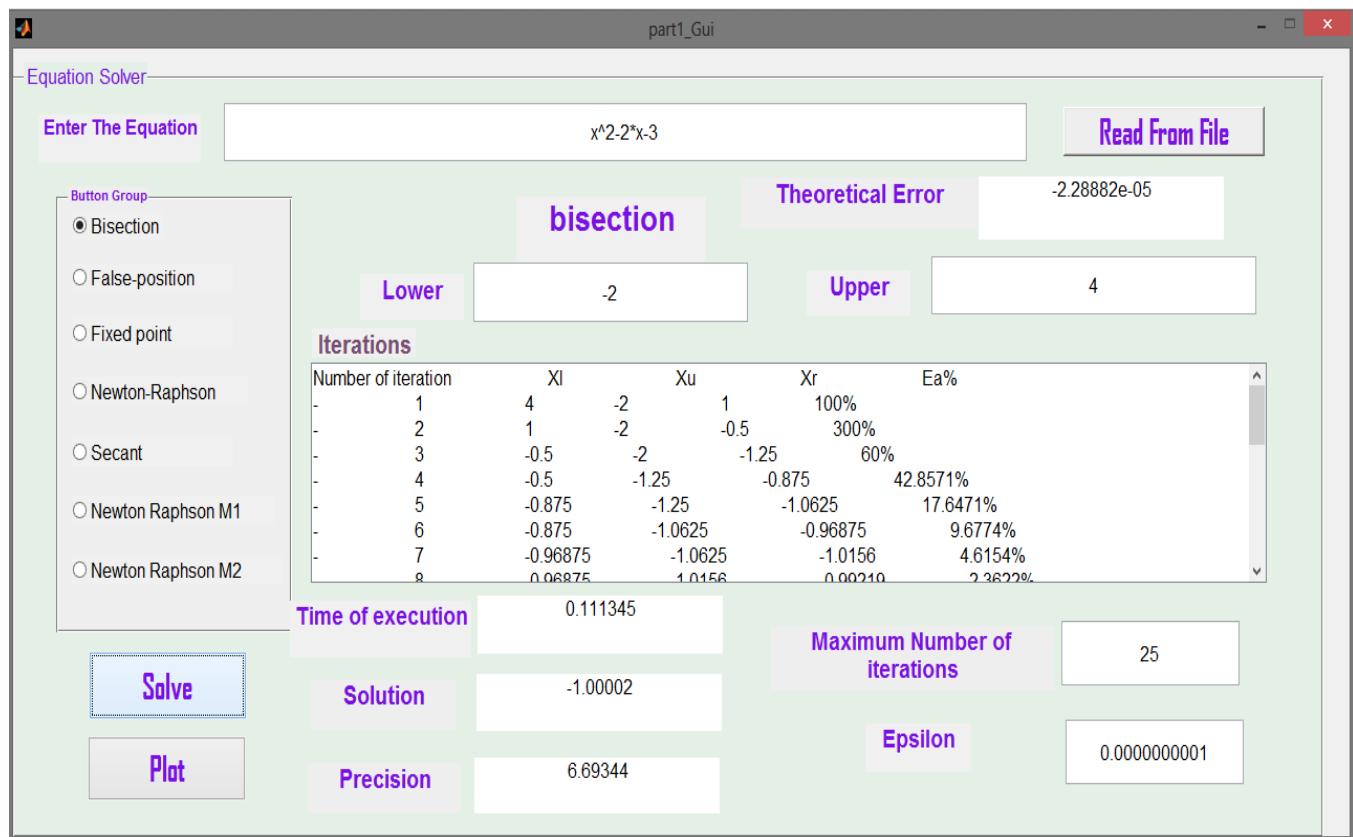
Method	Time	Root	Precision	Iteration	Relative Error
Bisection	0.112694	0.567142	8.61169	19	0.0006722%
False Position	0.0116607	0.567145	8.76914	5	0.00057434%
Fixed Point	0.092562	0.567145	8.21461	23	0.002%
Newton Raphson	0.0593299	0.567144	8.759	5	0.00058019%
Newton Raphson M 1	0.200625	0.56714	8.04647	25	0.0011831%
Newton Raphson M2	0.455799	0.567145	8.759	25	0.00058019%
Secant	0.0231367	0.567145	8.76914	5	0.00057434%

We can see that **false Position** take the least time of execution and it got the accurate root , **false Position ,Secant and Newton Raphson** take the least number of iteration and precisions are almost equal in every method so we can say that **false position** is the accurate and the fastest method in this example.

Example 4:

Equation = “ $x^2-2*x-3$ ”:

- In Bisection method



part1_Gui

Equation Solver

Enter The Equation:

bisection

	Lower	Upper	Theoretical Error
	-2	4	-2.28882e-05

Iterations

Iteration	Lower	Upper	Theoretical Error (%)
9	-0.99219	-1.0156	1.1673%
10	-0.99219	-1.0039	0.58708%
11	-0.99805	-1.0039	0.29268%
12	-0.99805	-1.001	0.14656%
13	-0.99951	-1.001	0.073224%
14	-0.99951	-1.0002	0.036626%
15	-0.99988	-1.0002	0.018309%
16	-0.99988	-1.0001	0.0091556%
17	0.00007	1.0001	0.0015776%

Time of execution: 0.111345

Solution: -1.00002

Maximum Number of iterations: 25

Precision: 6.69344

Epsilon: 0.0000000001

part1_Gui

Equation Solver

Enter The Equation:

bisection

	Lower	Upper	Theoretical Error
	-2	4	-2.28882e-05

Iterations

Iteration	Lower	Upper	Theoretical Error (%)
11	-0.99805	-1.0039	-1.001 0.29268%
12	-0.99805	-1.001	-0.99951 0.14656%
13	-0.99951	-1.001	-1.0002 0.073224%
14	-0.99951	-1.0002	-0.99988 0.036626%
15	-0.99988	-1.0002	-1.0001 0.018309%
16	-0.99988	-1.0001	-0.99997 0.0091556%
17	-0.99997	-1.0001	-1 0.0045776%

Time of execution: 0.111345

Solution: -1.00002

Maximum Number of iterations: 25

Precision: 6.69344

Epsilon: 0.0000000001

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

- False Position

part1_Gui

Equation Solver

Enter The Equation: $x^2-2*x-3$

falseposition

Iterations:

	Lower	Upper	Precision	Convergence (%)
8	2.9999	4	2.9999	0.0026668%
9	2.9999	4	3	0.0026665%

Time of execution: 0.0191213

Solution: 3.00003

Maximum Number of iterations: 25

Precision: 7.23384

Epsilon: 0.0000000001

Buttons: Solve, Plot

- In fixed point

part1_Gui

Equation Solver

Enter The Equation: $\sqrt{2*x+3}$

fixedpoint

Iterations:

	X ₀	X ₁	Precision	Convergence (%)
8	0.001	2.9993	Inf	0.13026%
9	2.9998	2.9998	Inf	0.043362%
10	2.9999	2.9999	Inf	0.014448%
11	3	3	Inf	0.005556%
12	3	3	Inf	0.00111111%
13	3	3	Inf	0.00111111%

Time of execution: 0.0766798

Solution: 3

Maximum Number of iterations: 25

Precision: Inf

Epsilon: 0.0000000001

Buttons: Solve, Plot

- In Newton Raphson

part1_Gui

Equation Solver

Enter The Equation: $x^2-2*x-3$

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

newtonRaphson

X₀: .001

Iterations

Number of iteration	X _i	Ea%
1	-1.5015	100.0666%
2	-1.0503	42.9631%
3	-1.0006	4.965%
4	-0.99999	0.05999%

Time of execution: 0.0522097

Solution: -1

Maximum Number of iterations: 20

Precision: 4.12043

Epsilon: 0.00001

- In Secant

part1_Gui

Equation Solver

Enter The Equation: $x^2-2*x-3$

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

Secant

Lower: .001 Upper: 4

Iterations

Number Of Iteration	X _{i-1}	X _i	X _{i+1}	Ea
1	4	1.5012	1.5012	166.4447%
2	1.5012	2.5719	2.5719	41.6298%
3	2.5719	3.3096	3.3096	22.2871%
4	3.3096	2.9658	2.9658	11.5886%
5	2.9658	2.9976	2.9976	1.058%
6	2.9976	3	3	0.080474%

Time of execution: 0.0238375

Solution: 2.99998

Maximum Number of iterations: 20

Precision: 3.82668

Epsilon: 0.00001

- In Newton Raphson Modified1

part1_Gui

Equation Solver

Enter The Equation Read From File

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Newton Raphson M1
- Secant
- Newton Raphson M2

NewtonRaphsonM1

X₀ M

Iterations

Number of iteration	X _i	Ea%
1	-1.1667	71.4286%
2	-1.0064	15.926%
3	-1	0%

Time of execution

Solution Maximum Number of iterations

Precision Epsilon

Solve **Plot**

- In Newton Raphson Modified2

part1_Gui

Equation Solver

Enter The Equation Read From File

Button Group

- Bisection
- False-position
- Fixed point
- Newton-Raphson
- Secant
- Newton Raphson M1
- Newton Raphson M2

NewtonRaphsonM2

X₀ M

Iterations

Number of iteration	X _i	Ea%
1	-0.84615	136.3636%
2	-0.99357	14.8373%
3	-0.99999	0.64192%
4	-1	0.00099997%

Time of execution

Solution Maximum Number of iterations

Precision Epsilon

Solve **Plot**

Comparison and analysis of all methods:

Method	Time	Root	Precision	Iteration	Relative Error
Bisection	0.111345	- 1.00002	6.69344	17	0.0045776%
False Position	0.019123	3.00003	7.23384	9	0.002665%
Fixed Point	0.0766798	3	inf	13	0.0011111%
Newton Raphson	0.0606904	-1	4.12043	4	0.05999%
Newton Raphson M 1	0.0323055	-1	-1.4611	3	0%
Newton Raphson M2	0.070612	-1	8.21464	4	0%
Secant	0.0260523	2.99998	3.82668	6	0.080474%

We can see that false Position take the least time of execution ,Newton Raphson Modified 1 and 2 got the accurate root , Newton Raphson Modified 1 take the least number of iteration and the least precision is in Newton Raphson Modified 1 method so we can say that Newton Raphson Modified 1 is the accurate and the fastest method in this example.

Conclusion:

For Bracketing Method:

- They always converge but bisection is slower but the false position converge slowly when the function is too sloppy like $x^{10} - 1$.
- The bisection converges slowly and get wrong results when $f(x_l)$ is close to zero when x_l is close to the exact root.

For open methods:

- Open method converge faster than bracketing methods but these methods can diverge such as Newton Raphson diverge when the equation is sloppy like $x^{10}-1$ but it converge faster when x_0 is closer to the exact root.
- Newton Raphson converges quadratically unless there is multiple roots.
- Newton Raphson takes less iteration than fixed point.
- Secant method converges faster than False Position method.

PART TWO

Objectives:

The aim of this assignment is to compare and analyze the behavior of numerical methods studied in class {Gaussian-elimination, LU decomposition, and Gaussian- Jordan}.

Specification:

The program must contain the following features:

- An interactive GUI that enables the user to enter set of linear equations. Reading from files must be available as well.
- A way to choose a method to solve the given set of linear equations.
- A way to enter the needed parameters if any;
- The answer for the chosen method indicating the execution time, solution, precision, and the iterations (if applicable).

DESIGN SPECIFICATIONS

There are two ways to enter the equations

Manual way :

- Enter the number of equations.
- Press 'ok' button.
- Enter the coefficients and the result of every equation.
- Press 'Enter' button.
- Enter the next equation by the same way in the same edit text then Press 'Enter' and so on.
- Press 'Solve' button to solve the system of the equation.
- Your solutions will appear in the list box and you can get the execution time.

Read File way:

- Press 'ReadFile' button.
 - The number of equation appears in '#equation' edit text.
 - Press 'Get from file' to get the coefficients and the result from the file.
 - The data appears in its edit text.
 - Press 'Solve' button to solve the system of the equations.
 - Your solutions appear in the list box and you can get the execution time.
-

DATA STRUCTURE USED AND HOW HELPFUL WAS YOUR CHOICE.

1- We have used arrays as a data structure:

We have used two dimensional arrays to solve the system of linear equations, these two dimensional arrays are augmented arrays. So, their entries are the coefficients of the equations and the last column in each of them is the resulted coefficient.

2- We have also used Strings as another data structure to handle Parsing and validating operations,

The program takes the user input as a string to facilitate different validation and other operations.

3- We have also used Cell Array as another data structures, we have used it to hold the solution for the system, we have also used it in reading from file operation as every cell of the array holds a line from the input file.

ANALYSIS FOR THE BEHAVIOR OF DIFFERENT EXAMPLES USING THE ANALYSIS TEMPLATE, AND YOUR CONCLUSION ABOUT THE BEHAVIOR OF EACH METHOD (AT LEAST THREE EXAMPLES).

Method	Five Equations	Four Equations	Three Equations	Two Equations
Gauss Elimination	0.00683474	0.00444223	0.00540778	0.00355428
LU_Decomposition	0.00678178	0.00446645	0.00514176	0.00343039
Gaussian_Jordan	0.00679123	0.00478132	0.00549071	0.00388434
Gaussian_Elimination (Partial Pivoting)	0.00675756	0.00425339	0.00504652	0.00353088
LU_Decomposition (Partial Pivoting)	0.00673499	0.00345247	0.00501901	0.00308218

If the input is a system of five linear equations:

The execution time of each method:

Gauss Elimination: it takes 0.00683474

Gauss Elimination (Partial Pivoting): it takes 0.00675756

Gaussian_Jordan: it takes 0.00679123

LU_Decomposition: it takes 0.00678178

LU_Decomposition (Partial Pivoting): it takes 0.00673499

If the input is a system of four linear equations:

The execution time of each method:

Gauss Elimination: it takes 0.00444223

Gauss Elimination (Partial Pivoting): it takes 0.00425339

Gaussian_Jordan: it takes 0.00478132

LU_Decomposition: it takes 0.00467992

LU_Decomposition (Partial Pivoting): it takes 0.00345247

If the input is a system of three linear equations:

The execution time of each method:

Gauss Elimination: it takes 0.00540778

Gauss Elimination (Partial Pivoting): it takes 0.00504652

Gaussian_Jordan: it takes 0.0054907

LU_Decomposition: it takes 0.00514176

LU_Decomposition (Partial Pivoting): it takes 0.00501901

If the input is a system of two linear equations:

The execution time of each method:

Gauss Elimination: it takes 0.00355428

Gauss Elimination (Partial Pivoting): it takes 0.00353088

Gaussian_Jordan: it takes 0.00388434

LU_Decomposition: it takes 0.0034303

LU_Decomposition (Partial Pivoting): it takes 0.00308218

Conclusion:

Gaussian-Jordan Takes Time more than other methods because it repeats gauss elimination n times so the order of this method is $O(n^3)$.

NUMERICAL METHODS

Gauss Elimination:

It's a numerical method to solve a system of linear equations with any number of coefficients and variables (Solve $Ax = b$).

It consists of two techniques:

- Forward elimination

Reduces $Ax = b$ to an upper triangular system $Tx = b'$

- Back substitution

Can then solve $Tx = b'$ for x

Pseudo Code:

```
start
do for k <- 1 to n-1
  do for i <- k to n-1
    m <- B(i+1,k)/B(k,k)
    do for j <- 1 to t
      B(i+1,j) <- B(i+1,j) - m * B(k,j)
    End for
  End for
End for
i <- n
x(i,1) <- B(i,t)/B(i,i)
do for i <- n-1 to 1
  s <- 0
  for k <- n to i+1
    s <- s + B(i, k) * x(k, 1)
  end for
```

```
x(i,1) <- (B(i,t)-s) / B(i,i)  
end for  
end
```

Problematic functions and the reason for their misbehavior and your suggestions (if exists).

1- Division by zero

It is possible that during both elimination and back-substitution phases a division by zero can occur.

2- Large Round-off Errors

Solution: **Pivoting**

So, Gaussian Elimination with Partial Pivoting

And LU_Decomposition with Partial Pivoting can solve that problem by switching rows according to the largest pivot.

- Avoids division by zero.
- Reduces round off error.

Gauss Elimination (Partial Pivoting):

Similar to Gauss Elimination but it solves Gaussian's pitfalls as it:

- Avoids division by zero.
- Reduces round off error.

Pseudo Code:

```
start
do for u <- 1 to n-1
    maximum <- abs(B(u, u))
    r <- u
    do for i <= u+1 to n
        if maximum < abs(B(i, u))
            maximum <- abs(B(i,u));
            r <- i
        end if
    end for
    row1(1,:) <- B(r,:)
    row2(1,:) <- B(u,:)
    B(u,:) <- row1
    B(r,:) <- row2

    do for k <- u+1: n
        m <- B(k,u) / B(u,u)
        do for j <- u to t
            B(k,j) <- B(k,j) - m * B(u,j);
        End for
    End for
    i <-n
    x(i,1) <- B(i,t) / B(i,i);
    do for i <- n-1 to 1
```

```
s <- 0
do for k <- n to i+1
  s <- s + B(i, k) * x(k)

end for
x(i) <- (B(i,t)-s) / B(i,i);
end for
end
```

LU-Decomposition:

Description:

Gauss Elimination can be used to decompose [A] into [L] and [U].

Therefore, it requires the same total FLOPs as for Gauss elimination:

In the order of (proportional to) N^3 where N is the # of unknowns.

l_{ij} values (the factors generated during the elimination step) can be stored in the lower part of the matrix to save storage. This can be done because these are converted to zeros anyway and unnecessary for the future operations.

Saves computing time by separating time-consuming elimination step from the manipulations of the right hand side.

Provides efficient means to compute the matrix inverse.

Pseudo Code:

```
function x = LUdecomposition(B)
    [n,t] = size(B)
    for k = 1 -> n-1
        for i = k -> n-1
            m = B(i+1,k) / B(k,k)
            for j = 1 -> t
                B(i+1,j) = B(i+1,j) - m * B(k,j)
            endfor
            B(i+1,k) = m
        endfor
    endfor
    if B(i,i) == 0
        disp('This system has no solutions!!')
        return
    endif
    i = n
    x(i,1) = B(i,t) / B(i,i)
    for i = 1 -> n-1
        s = 0
        for k = 1 -> i-1
            s = s + B(i, k) * x(k, 1)
        endfor
        if B(i,i) == 0
            disp('This system has no solutions!!')
            return
        endif
        x(i,1) = (B(i,t)-s) / B(i,i)
    endfor
```

```
endfor  
for i = n-1 -> 1  
s = 0  
for k = n -> i+1  
s = s + B(i, k) * x(k, 1)  
endfor  
x(i,1) = (B(i,t)-s) / B(i,i)  
endfor  
endfor
```

Problematic functions and the reason for their misbehavior and your suggestions (if exists).

1- Division by zero

It is possible that during both elimination and back-substitution phases a division by zero can occur.

2- Large Round-off Errors

Solution: Pivoting

So, LU_Decomposition with Partial Pivoting can solve that problem by switching rows according to the largest pivot.

- Avoids division by zero.
- Reduces round off error.

LU-Decomposition (Partial Pivoting):

Description:

- Avoids division by zero.
- Reduces round off error.

Pseudo Code:

```
function x = LUPartialPivoting(A)
[n, t] = size(A);
B = A;

for u = 1: n-1
maximum = abs(B(u, u));
r = u;
for i = u+1: n
if maximum < abs(B(i, u))
maximum =abs(B(i,u));
r = i;
end
end

row1(1,:) = B(r,:);
row2(1,:) = B(u,:);
B(u,:)= row1;
B(r,:)= row2;

for k = u+1: n;
m = B(k,u)/B(u,u);
for j = u: t;
B(k,j) = B(k,j) - m * B(u,j);
end
B(k,u)=m; %
end
```

```
i = n; x(i,1) = B(i,t) / B(i,i);
```

```
for i = 1:1:n-1;  
s = 0;  
for k = 1: 1: i-1;  
s = s + B(i, k) * x(k);  
end  
x(i,1)=(B(i,t)-s)/B(i,i);  
end;
```

```
for i = n-1: -1: 1;  
s = 0;  
for k = n: -1: i+1;  
s = s + B(i, k) * x(k);  
end  
x(i) = (B(i,t)-s) / B(i,i);  
end;
```

```
end
```

Gauss-Jordan

Description:

- It is a variation of Gauss elimination. The major differences are:
 - When an unknown is eliminated, it is eliminated from all other equations rather than just the subsequent ones.
 - All rows are normalized by dividing them by their pivot elements.
 - Elimination step results in an identity matrix.
 - Consequently, it is not necessary to employ back substitution to obtain solution.

Pseudo Code:

```
function x = GaussJordan(B)
[n,t] = size(B)
for k = 1 -> n
    for i = 1 -> n
        if i != k
            item = B(i,k)
            scale = B(k,k)
            for j = k ->t
                %scalling
                if scale == 0
                    disp('This system has no solutions!!')
                    return
                endif
            endfor
        endfor
    endfor
endfunction
```

```

B(k,j) = B(k,j) / scale
%change row
B(i,j) = B(i,j) - item * B(k,j)
endfor
endif
endfor
endfor
for u = 1 -> n
x(u) = B(u,t)
endfor
endfor

```

Problematic functions and the reason for their misbehavior and your suggestions (if exists).

If the pivot element =0, so I can't scale or normalize its row by dividing it by this pivot.

Solution:

We can replace the row which have zero pivot with another one in order to have non zero pivot and can normalize the row correctly.

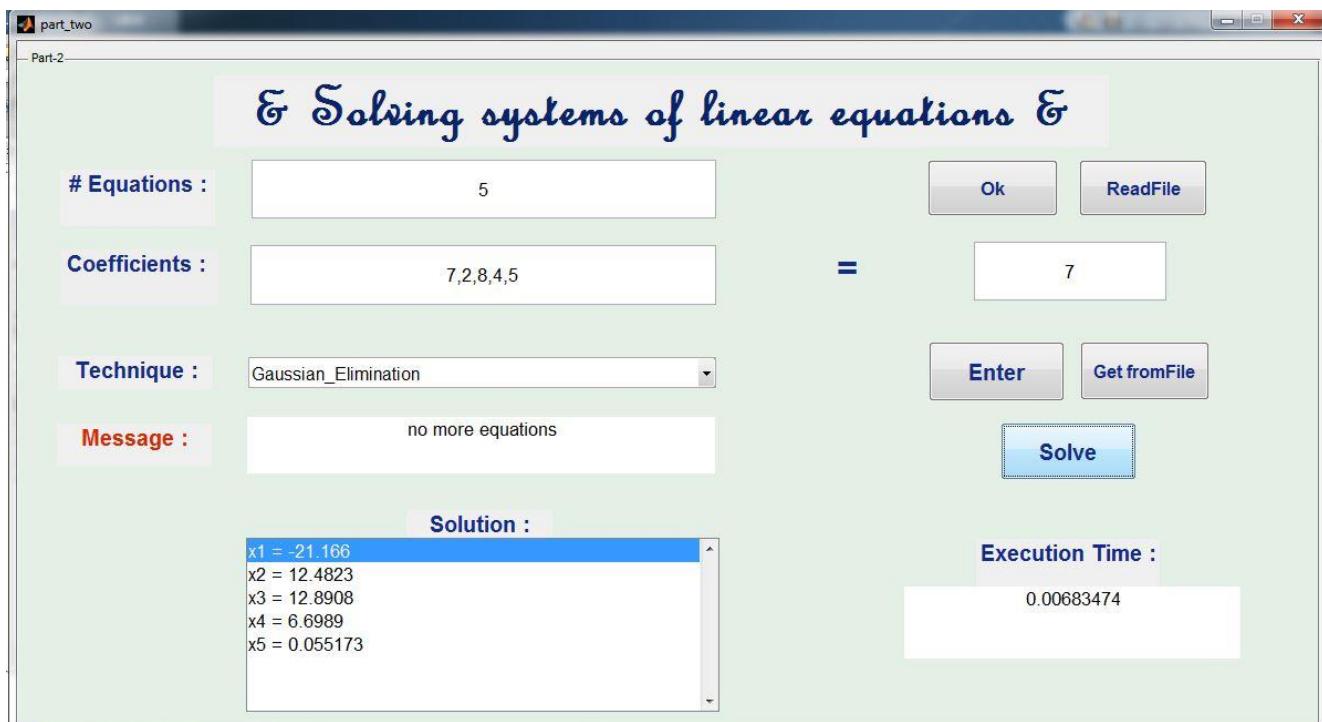
SAMPLE RUNS AND SNAPSHOTS FROM YOUR GUI.

In Gaussian Elimination Method:

- In case of system of five linear equations as an input:

Input file:

```
5
4,6,2,-2,7
3
2,0,5,-2,5
9
-4,-3,-5,4,8
10
8,18,-2,3,6
50
7,2,8,4,5
7|
```



- In case of system of four linear equations as an input:

Input file:

```
4
4,6,2,-2
8
2,0,5,-2
4
-4,-3,-5,4
1
8,18,-2,3
40
```

part_two

Part-2

& Solving systems of linear equations &

# Equations :	<input type="text" value="4"/>	<input type="button" value="Ok"/> <input type="button" value="ReadFile"/>
Coefficients :	<input type="text" value="8,18,-2,3"/>	=
Technique :	<input type="text" value="Gaussian_Elimination"/>	<input type="button" value="Enter"/> <input type="button" value="Get fromFile"/>
Message :	<input type="text" value="no more equations"/>	
	<input type="button" value="Solve"/>	
Solution :	Execution Time : <input type="text" value="0.00444223"/>	
<pre>x1 = -13.5 x2 = 8.66667 x3 = 7 x4 = 2</pre>		

- In case of system of three linear equations as an input:

Input file:

```
3
-1,1,2
1
3,-1,1
1
-1,3,4
1
```

part_two

— Part-2 —

& Solving systems of linear equations &

Equations :

Coefficients : =

Technique :

Message :

Solution :

Execution Time :

The screenshot shows a Windows application window titled "part_two". The title bar also displays "— Part-2 —". The main area contains a heading "Solving systems of linear equations". Below it are several input fields and buttons:

- "# Equations :" field with value "3", accompanied by "Ok" and "ReadFile" buttons.
- "Coefficients :" field with value "-1,3,4", followed by an equals sign and a field with value "1", with "Enter" and "Get fromFile" buttons.
- "Technique :" dropdown menu set to "Gaussian_Elimination".
- "Message :" field with value "no more equations", with a "Solve" button.
- "Solution :" scrollable text area displaying the solution: "x1 = -0.2", "x2 = -0.8", and "x3 = 0.8".
- "Execution Time :" field with value "0.00540778".

- In case of system of two linear equations as an input:

Input file:

```

2
1, 5
7
-2, -7
-5

```

part_two

Part-2

& Solving systems of linear equations &

# Equations :	<input type="text" value="2"/>	<input type="button" value="Ok"/>	<input type="button" value="ReadFile"/>
Coefficients :	<input type="text" value="-2,-7"/>	=	<input type="text" value="-5"/>
Technique :	<input type="button" value="Gaussian_Elimination"/>	<input type="button" value="Enter"/>	<input type="button" value="Get fromFile"/>
Message :	<input type="text" value="no more equations"/>		
	<input type="button" value="Solve"/>		
Solution :	<pre>x1 = -8 x2 = 3</pre>		
Execution Time :	<input type="text" value="0.00355428"/>		

In Gaussian Elimination (Partial Pivoting) Method:

- In case of system of five linear equations as an input:

The screenshot shows a software interface titled "Part-2". The main title is "& Solving systems of linear equations &". The input fields are:
Equations : 5
Coefficients : 7,2,8,4,5 = 7
Technique : Gaussian_Elimination (Partial Pivoting)
Message : no more equations
Solution :
x1 = -21.166
x2 = 12.4823
x3 = 12.8908
x4 = 6.6989
x5 = 0.055173
Execution Time : 0.00675756

- In case of system of four linear equations as an input:

The screenshot shows a software interface titled "Part-2". The main title is "& Solving systems of linear equations &". The input fields are:
Equations : 4
Coefficients : 8,18,-2,3 = 40
Technique : Gaussian_Elimination (Partial Pivoting)
Message : no more equations
Solution :
x1 = -13.5
x2 = 8.6667
x3 = 7
x4 = 2
Execution Time : 0.00425339

- In case of system of three linear equations as an input:

& Solving systems of linear equations &

# Equations :	<input type="text" value="3"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-1,3,4"/>	=	<input type="text" value="1"/>
Technique :	<input type="button" value="Enter"/>		
Message :	<input type="button" value="Get fromFile"/>		
Solution :		Solve	
$x_1 = -0.2$ $x_2 = -0.8$ $x_3 = 0.8$		Execution Time : 0.00504652	

- In case of system of two linear equations as an input:

& Solving systems of linear equations &

# Equations :	<input type="text" value="2"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-2,-7"/>	=	<input type="text" value="-5"/>
Technique :	<input type="button" value="Enter"/>		
Message :	<input type="button" value="Get fromFile"/>		
Solution :		Solve	
$x_1 = -8$ $x_2 = 3$		Execution Time : 0.00353088	

In LU-Decomposition Method:

- In case of system of five linear equations as an input:

The screenshot shows a software interface titled "part_two" with the sub-section "Part-2". The main title is "& Solving systems of linear equations &".

Input fields:
Equations : 5
Coefficients : 7,2,8,4,5 = 7
Technique : LU_Decomposition
Message : no more equations

Solution :
x1 = -21.166
x2 = 12.4823
x3 = 12.8908
x4 = 6.6989
x5 = 0.055173

Execution Time : 0.00678178

Buttons: Ok, ReadFile, Enter, Get from File, Solve.

- In case of system of four linear equations as an input:

The screenshot shows a software interface titled "part_two" with the sub-section "Part-2". The main title is "& Solving systems of linear equations &".

Input fields:
Equations : 4
Coefficients : 8,18,-2,3 = 40
Technique : LU_Decomposition
Message : no more equations

Solution :
x1 = -13.5
x2 = 8.66667
x3 = 7
x4 = 2

Execution Time : 0.00446645

Buttons: Ok, ReadFile, Enter, Get from File, Solve.

- In case of system of three linear equations as an input:

& Solving systems of linear equations &

# Equations :	<input type="text" value="3"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-1,3,4"/>	=	<input type="text" value="1"/>
Technique :	<input type="text" value="LU_Decomposition"/>	Enter	Get fromFile
Message :	<input type="text" value="no more equations"/>		
Solution :		Execution Time :	
<code>x1 = -0.2 x2 = -0.8 x3 = 0.8</code>		0.00514176	
		Solve	

- In case of system of two linear equations as an input:

& Solving systems of linear equations &

# Equations :	<input type="text" value="2"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-2,-7"/>	=	<input type="text" value="-5"/>
Technique :	<input type="text" value="LU_Decomposition"/>	Enter	Get fromFile
Message :	<input type="text" value="no more equations"/>		
Solution :		Execution Time :	
<code>x1 = -8 x2 = 3</code>		0.0034303	
		Solve	

In LU-Decomposition (Partial Pivoting) Method:

- In case of system of five linear equations as an input:

The screenshot shows a software window titled "part_two" with the sub-tittle "Part-2". The main title is "& Solving systems of linear equations &".
Fields:
Equations : 5
Coefficients : 7,2,8,4,5
Technique : LU_Decomposition(Partial Pivoting)
Message : no more equations
Solution :
x1 = -21.166
x2 = 12.4823
x3 = 12.8908
x4 = 6.6989
x5 = 0.055173
Execution Time : 0.00673499

- In case of system of four linear equations as an input:

The screenshot shows a software window titled "part_two" with the sub-tittle "Part-2". The main title is "& Solving systems of linear equations &".
Fields:
Equations : 4
Coefficients : 8,18,-2,3
Technique : LU_Decomposition(Partial Pivoting)
Message : no more equations
Solution :
x1 = -13.5
x2 = 8.6667
x3 = 7
x4 = 2
Execution Time : 0.00345247

- In case of system of three linear equations as an input:

& Solving systems of linear equations &

# Equations :	<input type="text" value="3"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-1,3,4"/>	=	<input type="text" value="1"/>
Technique :	LU_Decomposition(Partial Pivoting)		
Message :	no more equations		
Solution :		Execution Time :	
<code>x1 = -0.2 x2 = -0.8 x3 = 0.8</code>		0.00501901	
		Solve	

- In case of system of two linear equations as an input:

& Solving systems of linear equations &

# Equations :	<input type="text" value="2"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-2,-7"/>	=	<input type="text" value="-5"/>
Technique :	LU_Decomposition(Partial Pivoting)		
Message :	no more equations		
Solution :		Execution Time :	
<code>x1 = -8 x2 = 3</code>		0.00308218	
		Solve	

In Gaussian Jordan Method:

- In case of system of five linear equations as an input:

The screenshot shows a software window titled "part_two" with the sub-tittle "Part-2". The main title of the interface is "& Solving systems of linear equations &".

Input fields and their values:

- # Equations : 5
- Coefficients : 7,2,8,4,5
- = 7
- Technique : Gaussian_Jordan
- Message : no more equations

Buttons:

- Ok
- ReadFile
- Enter
- Get fromFile
- Solve

Solution output:

x1 = -21.166
x2 = 12.4823
x3 = 12.8908
x4 = 6.6989
x5 = 0.055173

Execution Time :

0.00679123

- In case of system of four linear equations as an input:

The screenshot shows a software window titled "part_two" with the sub-tittle "Part-2". The main title of the interface is "& Solving systems of linear equations &".

Input fields and their values:

- # Equations : 4
- Coefficients : 8,18,-2,3
- = 40
- Technique : Gaussian_Jordan
- Message : no more equations

Buttons:

- Ok
- ReadFile
- Enter
- Get fromFile
- Solve

Solution output:

x1 = -13.5
x2 = 8.6667
x3 = 7
x4 = 2

Execution Time :

0.00478132

- In case of system of three linear equations as an input:

part_two

Part-2

& Solving systems of linear equations &

# Equations :	<input type="text" value="3"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-1,3,4"/>	=	<input type="text" value="1"/>
Technique :	<input type="text" value="Gaussian_Jordan"/>	Enter	Get fromFile
Message :	<input type="text" value="no more equations"/>		
Solution :	$x_1 = -0.2$ $x_2 = -0.8$ $x_3 = 0.8$		
Execution Time :	<input type="text" value="0.0054907"/>		

Solve

- In case of system of two linear equations as an input:

part_two

Part-2

& Solving systems of linear equations &

# Equations :	<input type="text" value="2"/>	Ok	ReadFile
Coefficients :	<input type="text" value="-2,-7"/>	=	<input type="text" value="-5"/>
Technique :	<input type="text" value="Gaussian_Jordan"/>	Enter	Get fromFile
Message :	<input type="text" value="no more equations"/>		
Solution :	$x_1 = -8$ $x_2 = 3$		
Execution Time :	<input type="text" value="0.00388434"/>		

Solve

Special Cases:

```
5
4,6,2,-2
8
2,0,5,-2
4
-4,-3,-5,4
1
8,18,-2,3
40
```

Part-2

& Solving systems of linear equations &

# Equations :	<input type="text" value="5"/>	<input type="button" value="Ok"/>	<input type="button" value="ReadFile"/>
Coefficients :	<input type="text"/>	=	<input type="text"/>
Technique :	<input type="button" value="Gaussian_Elimination"/>	<input type="button" value="Enter"/>	<input type="button" value="Get fromFile"/>
Message :	Not enough Equations!!		
Solution :	<input type="text"/>		
Execution Time :	<input type="text"/>		

```
2
2,3
4
2,3
5
```

part_two

Part-2

& Solving systems of linear equations &

Equations :

Coefficients : =

Technique :

Message : no more equations

Solution :
This system has no solutions!!

Execution Time :
0.00160349