Numerical Analysis Project Report

1- Karim M. Elsayad - 6023
2- Hagar Omar Abouroumia - 5943
3- Marwan Medhat El Nahas - 5365
4- Youssef Ahmed Walid - 5491
5- Ahmed M. Abdelrehim - 5684

Contents:

1 - Code Organization

Functions:

Bisection Function:

```
function ret = Bisection(app)
            xl, xu, funcm maxIter are given

         if func(xl) * func(xu) > 0
                  No solution!

    else

        for i=1:1:MaxIter

            % compute midpoin xr
            xr = (xu + xl)/2;

            % compute test = f(xl) * f(xr)
            test = func(xl) * func(xr);

            if test < 0
                xr_old = xu;
                xu = xr;
            elseif test > 0
                xr_old = xl;
                xl = xr;


            if test == 0
                ea = 0;
            else
                % approximate relative error
                ea = abs((xr - xr_old)/xr) * 100;

            if ea < Precision
                break;
```

```
ret = [xr, i];
```

Bisection Notes:

After choosing Bisection in the GUI, initialize xr_old=0,we take the input of the user in xl (x lower) & xu (x upper) , Then we take the function input (function expression) and get the value of Function(xl) & Function(xu) If Function(xl) * Function(xu) > 0 display No Solution Because they have the same sign.

Else we loop from i=1 to the Maximum Number of iterations taken from the input of the user in GUI or break if The Error = (absolute of (xr-xr_old) * 100/xr) is < Precision.

At each iteration xr (new root) = (xl+xu)/2, Then get

Function(xl) * Function(xr) if (< 0 xr_old = xu ,xu=xr ,iterate once again) else if (> 0 xr_old = xl , xl=xr,iterate once again) else(= 0 break)

Returns the xr,Number of Iterations of the Final Iteration which is the desired Root.

---

1) False Position (Regula Falsi) Function:

```
function ret = FalsePosition(app)
         xl = XlEditField_False.Value;
         xu = XuEditField_False.Value;
         xr_old=0;
         % on solution if both starting points have same sign
         if func(xl)*func(xu) > 0
             disp("No Solution!");
             xr = 0;
             i = 0;
             ErrorLabel.Text = "No solution in range";
         else
             for i=1:1:MaxIter
                 % compute midpoin xr
                 % minimize functon calls
                 fu = func(xu);
                 fl = func(xl);
                 xr = (xl * fu - xu * fl)/(fu - fl);

                 % compute test = f(xl) * f(xr)
                 test = fl * func(xr);
                 if test < 0
                     xr_old = xu;
                     xu = xr;
                 elseif test > 0
                     xr_old = xl;
                     xl = xr;
                 end
                 if test == 0
                     ea = 0;
                 else
                     % approximate relative error
                     ea = abs((xr - xr_old)/xr) * 100;
                 end
                 if ea < Precision
                     break;

ret = [xr, i];
```

False Position (Regula Falsi) Notes:

After choosing False Position in the GUI, initialize xr_old=0,we take the input of the user in xl (x lower) & xu (x upper) , Then we take the function input (function expression) and get the value of Function(xl) & Function(xu) If Function(xl) * Function(xu) > 0 display No Solution Because they have the same sign.

Else we loop from i=1 to the Maximum Number of iterations taken from the input of the user in GUI or break if The Absolute Error = (absolute of (xr-xr_old) * 100/xr) is < Precision.

At each iteration xr (new root) =(xl F(xu)-xu F(xl))/(F(xu)-F(xl)),

Then get Function(xl) * Function(xr) if (< 0 xr_old = xu ,xu=xr ,iterate- once again) else if (> 0 xr_old = xl , xl=xr,iterate once again) else(= 0 break)

Returns the xr,Number of Iterations of the Final Iteration which is the desired Root.

---

2) Fixed Point Function:

```
function ret = Fixedpoint(app)
            x0 = X0EditField_Fixed.Value;
            syms gFunc(x)
            gFunc(x) = str2sym(gxEditField.Value);
            xr = x0;
            for i=1:1:MaxIter
                xr_old = xr;
                xr = gFunc(xr_old);
                ea = abs((xr - xr_old)/xr) * 100;

                if (ea < Precision)
                    break
                end
            end
            ret = [xr, i];
        end
```

Fixed Point Notes:

After choosing Fixed Point in the GUI, we take the input of the user for x0 & gFunc(x), initialize xr = x0, we loop from i=1 to Maximum Number of Iteration taken from the input of the user in GUI.

Each Iteration make x_old = xr , calculate xr = gFunc(xr_old) ,

calculate Absolute Error = (absolute of (xr-xr_old) * 100/xr)

if (< Precision) break.

Returns the xr,Number of Iterations of the Final Iteration which is the desired Root.

Newton Raphson Function:

```
function ret = NewtonRaphson(app)
          xr_old = X0EditField_Newton.Value;
          for i=1:1:MaxIter
              xr = xr_old - ((func(xr_old))/(funcDiff(xr_old)));
   ea = abs((xr - xr_old)/xr) * 100;
              if (ea < Precision)
                  break
              end
          end
        ret = [xr, i];
      end
```

Newton Raphson Notes:

After choosing Fixed Point in the GUI, we take the input of the user for xr_old we loop from i=1 to Maximum Number of Iteration taken from the input of the user in GUI.

Each iteration xr = xr_old-(Function(xr_old)/Derivative(Func(xr_old)),

calculate Absolute Error = (absolute of (xr-xr_old) * 100/xr)

if (< Precision) break.

Returns the xr,Number of Iterations of the Final Iteration which is the desired Root.

3) Secant Function:

```
function ret = Secant(app)
               xi_old = X0EditField_Secant.Value;
            xi = X1EditField_Secant.Value;
            for i=1:1:MaxIter
                xiplus = xi - (func(xi)*(xi_old -
xi))/(func(xi_old)-func(xi));
                ea = abs((xiplus-xi)/xiplus)*100;
                if (ea < Precision)
                    break
                end
                xi_old = xi;
                xi = xiplus;
            end
              ret = [xiplus, i];
        end
```

-> Secant Notes:

After choosing Fixed Point in the GUI, we take the input of the user for xi & xi_old we loop from i=1 to Maximum Number of Iteration taken from the input of the user in GUI.

Each iteration xiplus = xi-(Func(xi)*(xi_old - xi))/(Func(xi_old)-Func(xi))

calculate Absolute Error = (absolute of (xr-xr_old) * 100/xr)

if (< Precision) break.

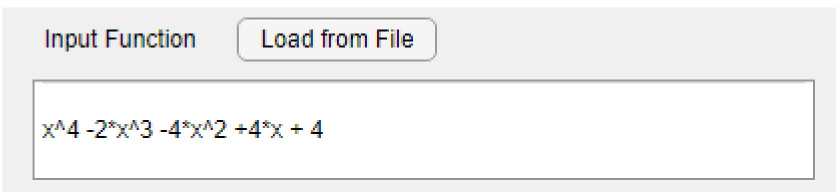
Else : xi_old = xi & xi=xiplus.

Returns the xiplus,Number of Iterations of the Final Iteration which is the desired Root.

How to use

- No implicit multiplication. Use `*`. (wrong: `2x`, correct: `2 * x). It's only a text box that doesn't provide advance parsing capabilities.
- Only enter the Left-hand side of the equation, RHS is assumed to be zero. Adding an equal sign results in an error.
- The buttons under the Input box tell what functions are known to work.
- You can use "load from file" button to read a function from a file, provided it's written in the same format mentioned above

- Example of an input function



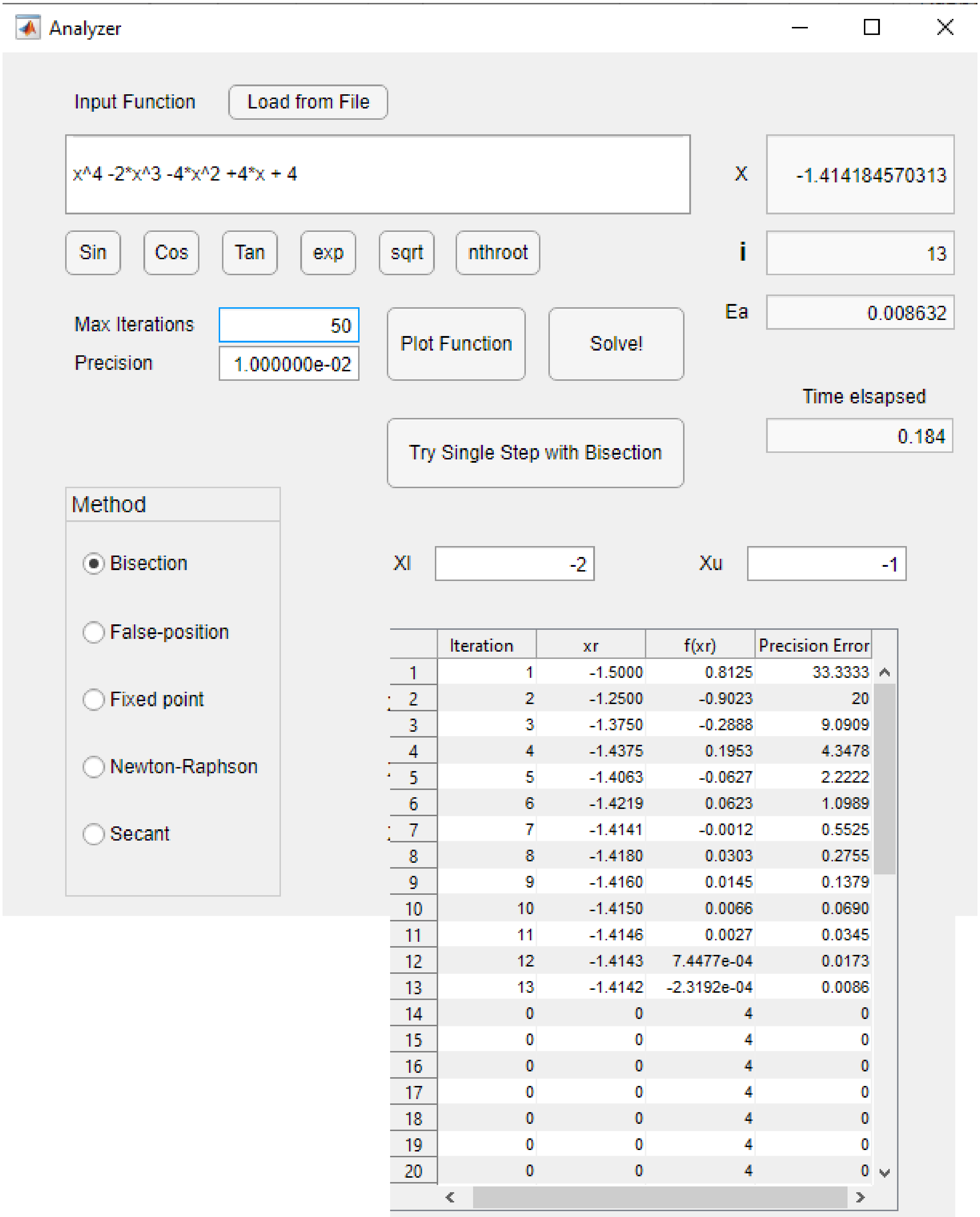
Installation Guide:

As this app is made using app designer, it requires version 2016 or later to run.

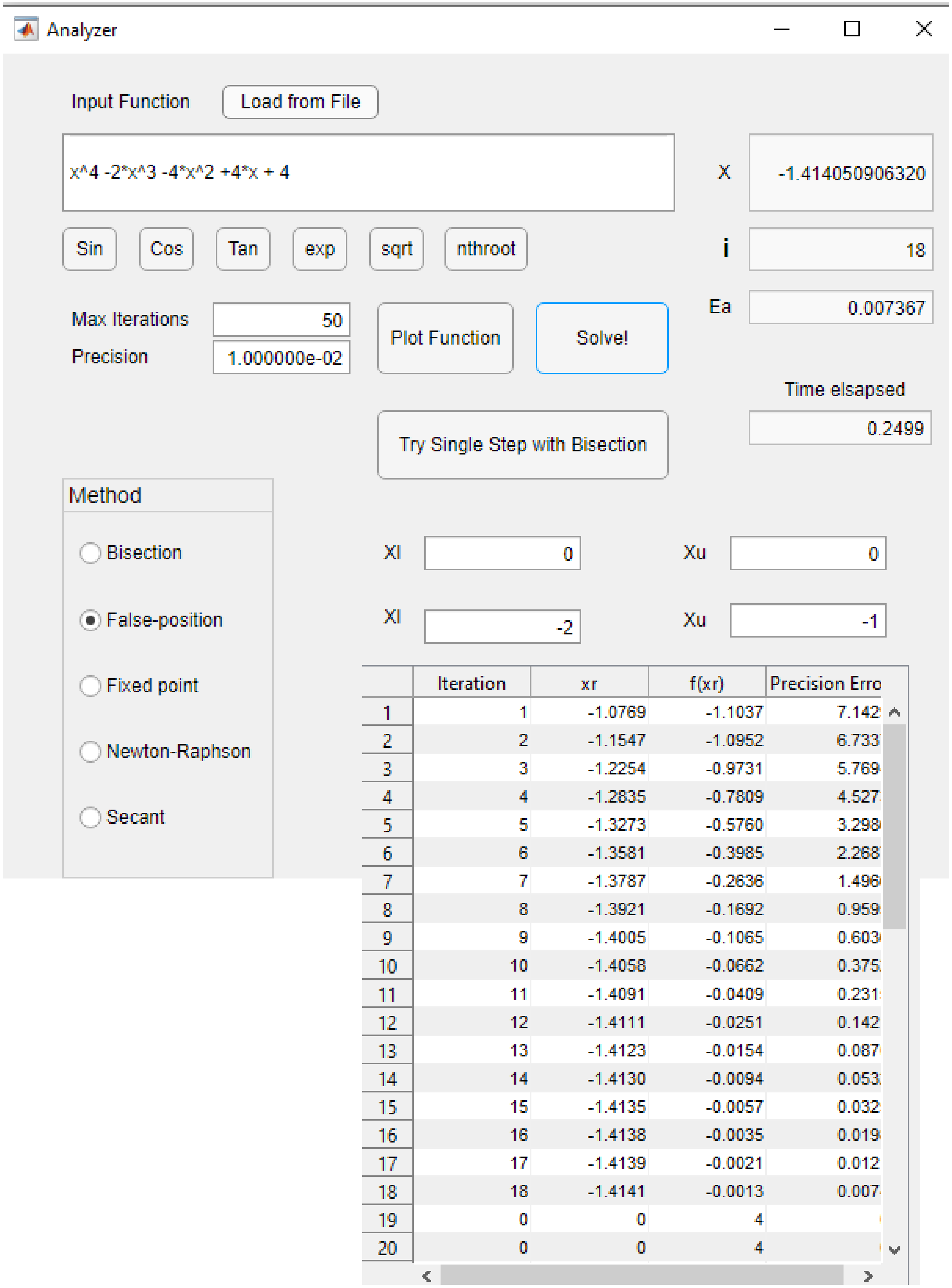
- Put the three provided files (analyze, single_step_mode, plotFunction) into a folder. Alternatively, use the matlab function `appinfo = matlab.apputil.install(appfile)` to install using the provided `Analyzer_1.mlappinstall` file.
- After files are placed somewhere, add that place to path, or change current directory to that place. Ultimately, you want the files to be visible for matlab search, as it's the case with all functions.
- Run `analyze` through the matlab console/cli.

Sample Runs

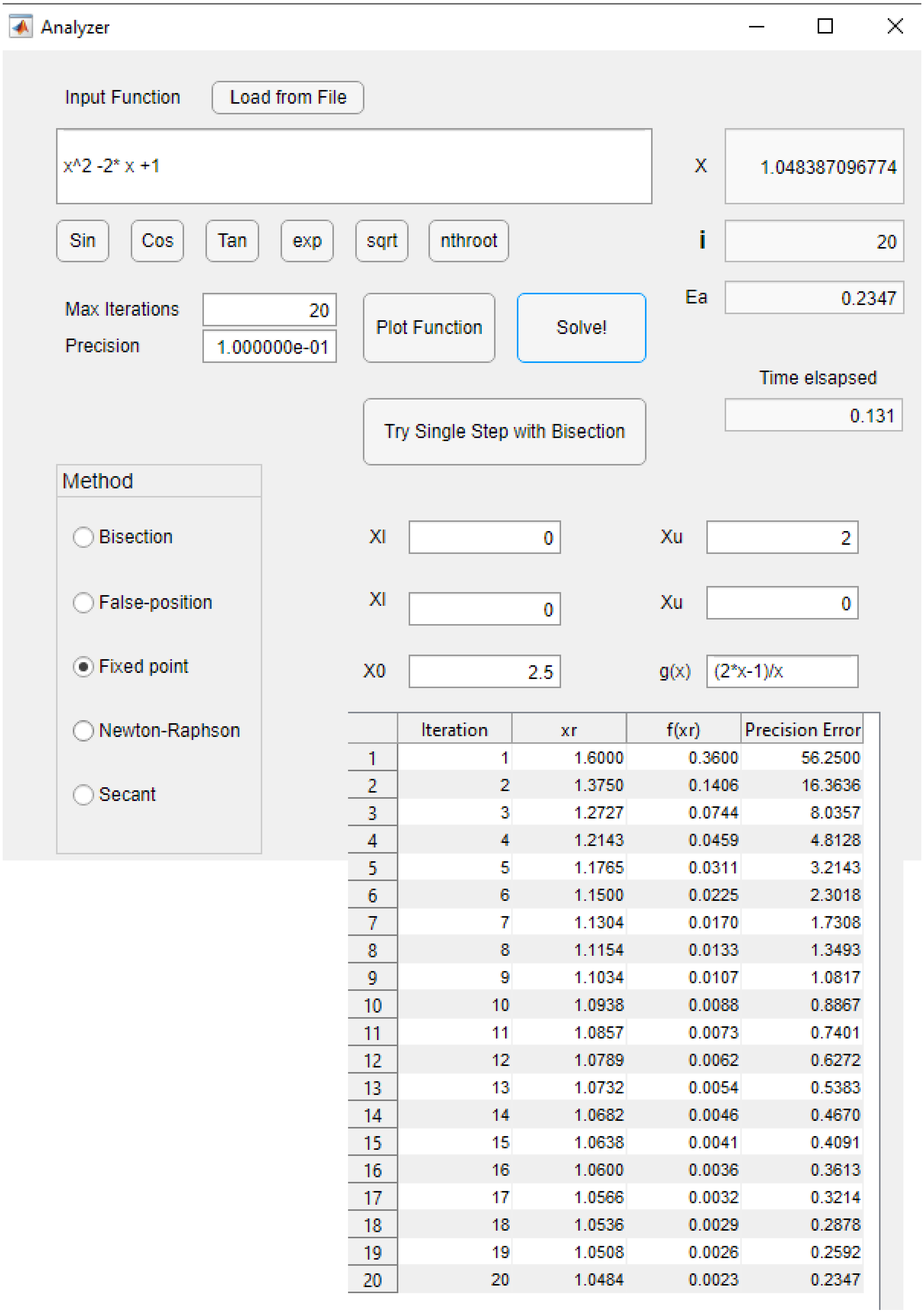
Bisection



False position

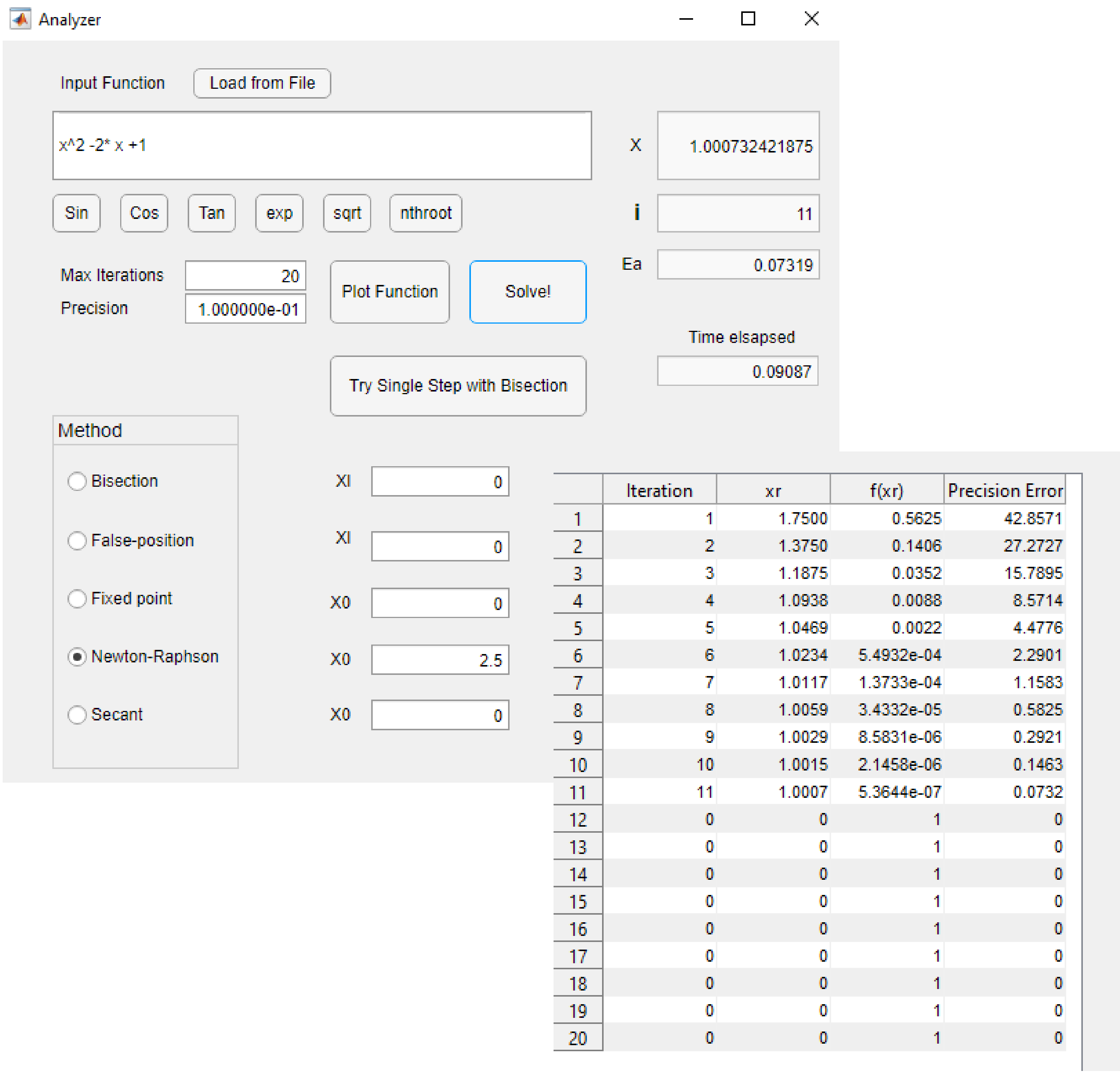


Fixed point



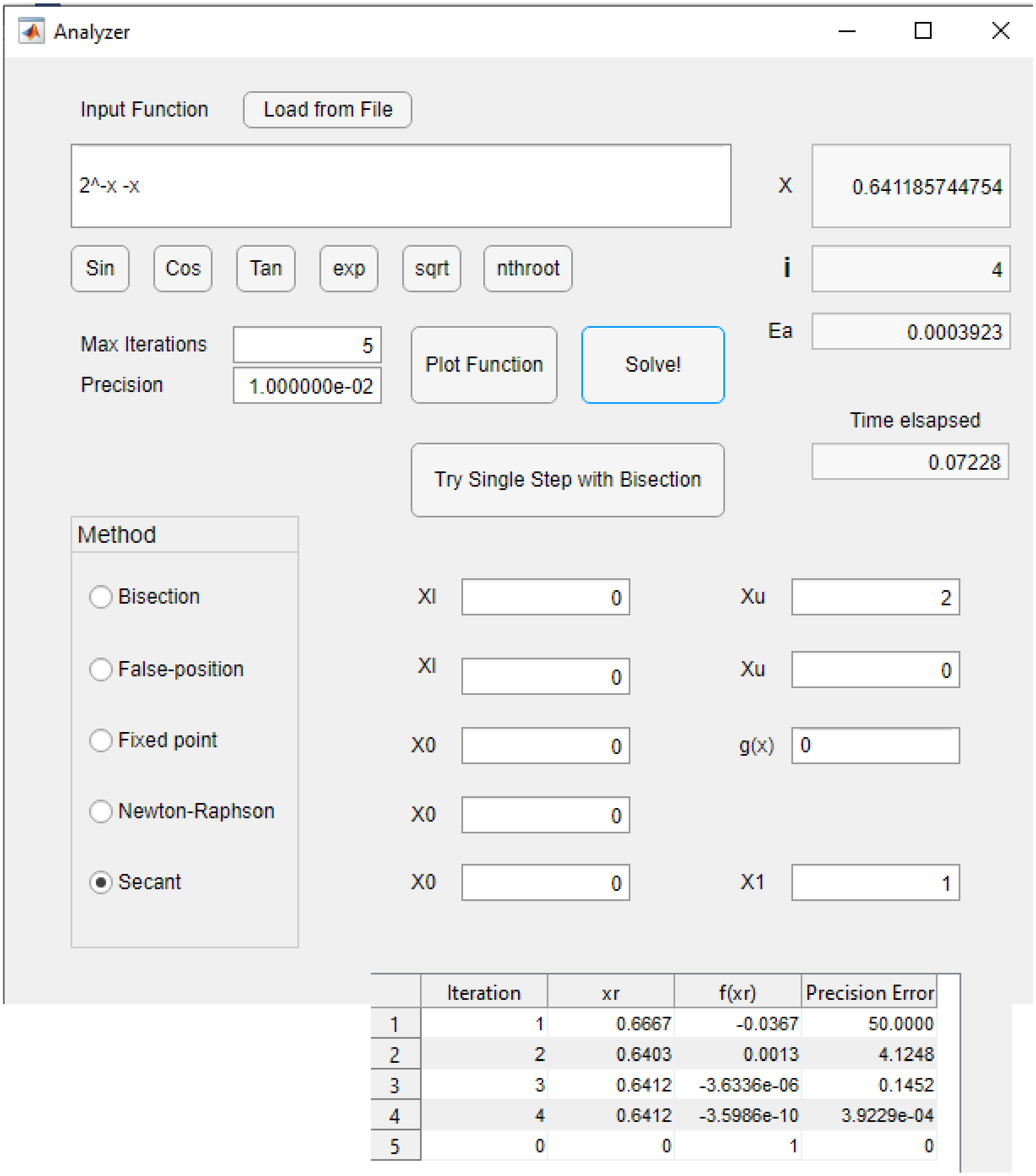
	Iteration	xr	f(xr)	Precision Error
1	1	1.6000	0.3600	56.2500
2	2	1.3750	0.1406	16.3636
3	3	1.2727	0.0744	8.0357
4	4	1.2143	0.0459	4.8128
5	5	1.1765	0.0311	3.2143
6	6	1.1500	0.0225	2.3018
7	7	1.1304	0.0170	1.7308
8	8	1.1154	0.0133	1.3493
9	9	1.1034	0.0107	1.0817
10	10	1.0938	0.0088	0.8867
11	11	1.0857	0.0073	0.7401
12	12	1.0789	0.0062	0.6272
13	13	1.0732	0.0054	0.5383
14	14	1.0682	0.0046	0.4670
15	15	1.0638	0.0041	0.4091
16	16	1.0600	0.0036	0.3613
17	17	1.0566	0.0032	0.3214
18	18	1.0536	0.0029	0.2878
19	19	1.0508	0.0026	0.2592
20	20	1.0484	0.0023	0.2347

Newton Raphson



	Iteration	xr	f(xr)	Precision Error
1	1	1.7500	0.5625	42.8571
2	2	1.3750	0.1406	27.2727
3	3	1.1875	0.0352	15.7895
4	4	1.0938	0.0088	8.5714
5	5	1.0469	0.0022	4.4776
6	6	1.0234	5.4932e-04	2.2901
7	7	1.0117	1.3733e-04	1.1583
8	8	1.0059	3.4332e-05	0.5825
9	9	1.0029	8.5831e-06	0.2921
10	10	1.0015	2.1458e-06	0.1463
11	11	1.0007	5.3644e-07	0.0732
12	0	0	1	0
13	0	0	1	0
14	0	0	1	0
15	0	0	1	0
16	0	0	1	0
17	0	0	1	0
18	0	0	1	0
19	0	0	1	0
20	0	0	1	0

Secant Method



Single Step Mode