Mapua University

School of Electrical, Electronics

and

Computer Engineering

COE60/C1

Machine Problem 2

User Manual

Sardina, Kent Johnric M.

2014150748

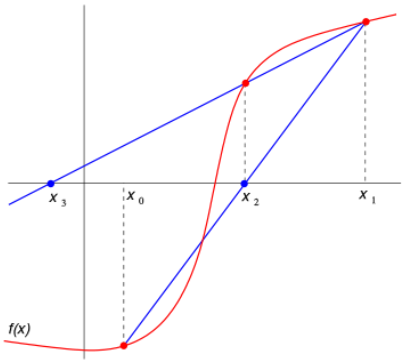
COE60/C1

Prof. Carlos V. Hortinela IV

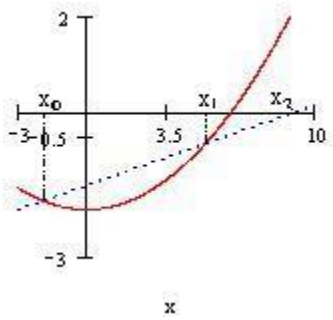
Secant Method

In numerical methods, the secant method is a root-finding algorithm that uses a succession of roots of secant lines to better approximate a root of a function f. The secant method can be thought of as a finite difference approximation of Newton's method. However, the method was developed independently of Newton's method, and predates it by over 3,000 years.

The Newton-Raphson algorithm requires the evaluation of two functions (the function and its derivative) per each iteration. If they are complicated expressions it will take considerable amount of effort to do hand calculations or large amount of CPU time for machine calculations. Hence it is desirable to have a method that converges (please see the section order of the numerical methods for theoretical details) as fast as Newton's method yet involves only the evaluation of the function.



Let x0 and x1 are two initial approximations for the root 's' of f(x) = 0 and f(x0) & f(x1) respectively, are their function values. If x 2 is the point of intersection of x-axis and the line-joining the points (x0, f(x0)) and (x1, f(x1)) then x2 is closer to 's' than x0 and x1. The equation relating x0, x1 and x2 is found by considering the slope 'm'.



$$\mathbf{m} = \frac{f(x1) - f(x0)}{x1 - x0} = \frac{f(x2) - f(x1)}{x2 - x1} = \frac{0 - f(x1)}{x2 - x1}$$

$$\mathbf{x2 - x1} = \frac{-f(x1) * (x1 - x0)}{f(x1) - f(x0)}$$

$$\mathbf{x2} = \mathbf{x1} - \frac{f(x1) * (x1 - x0)}{f(x1) - f(x0)}$$

Or in general the iterative process can be written as

$$\mathbf{Xi+1} = \mathbf{Xi} - \frac{\boldsymbol{f(xi)} * \boldsymbol{(xi - xi - 1)}}{\boldsymbol{f(xi) - f(xi - 1)}}$$

I=1,2,3,4…….

This formula is similar to Regula-falsi scheme of root bracketing methods but differs in the implementation. The Regula-falsi method begins with the two initial approximations 'a' and 'b' such that a < s < b where s is the root of f(x) = 0. It proceeds to the next iteration by calculating c(x2) using the above formula and then chooses one of the interval (a,c) or (c,h) depending on f(a) * f(c) < 0 or > 0 respectively. On the other hand secant method starts with two initial approximation x0 and x1 (they may not bracket the root) and then calculates the x2 by the same formula as in Regula-falsi method but proceeds to the next iteration without bothering about any root bracketing.

Parts of the program

I. Numerical method selection for MP2 (Main Window)

Figure 1. Main Window

This depicts the available numerical method to be used by the user. For Machine Problem 2, Secant Method is available.

II. Secant Method Calculator Window

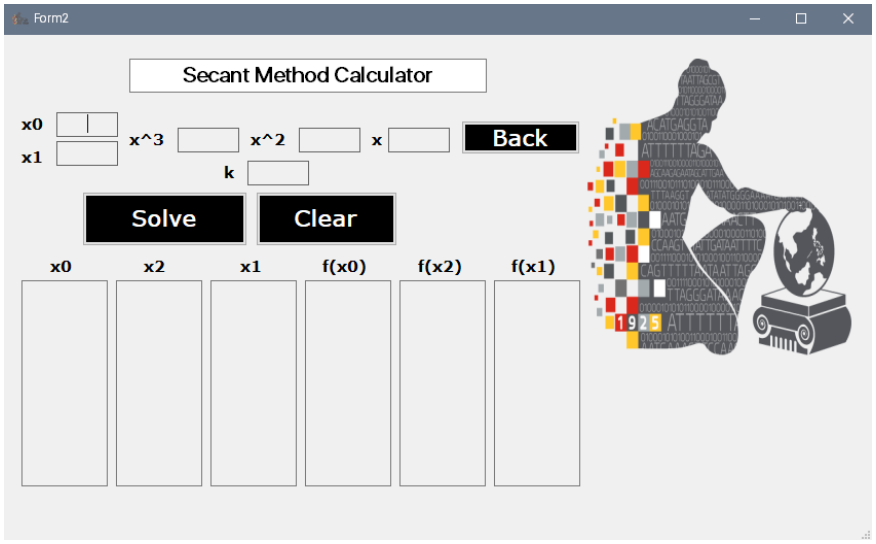
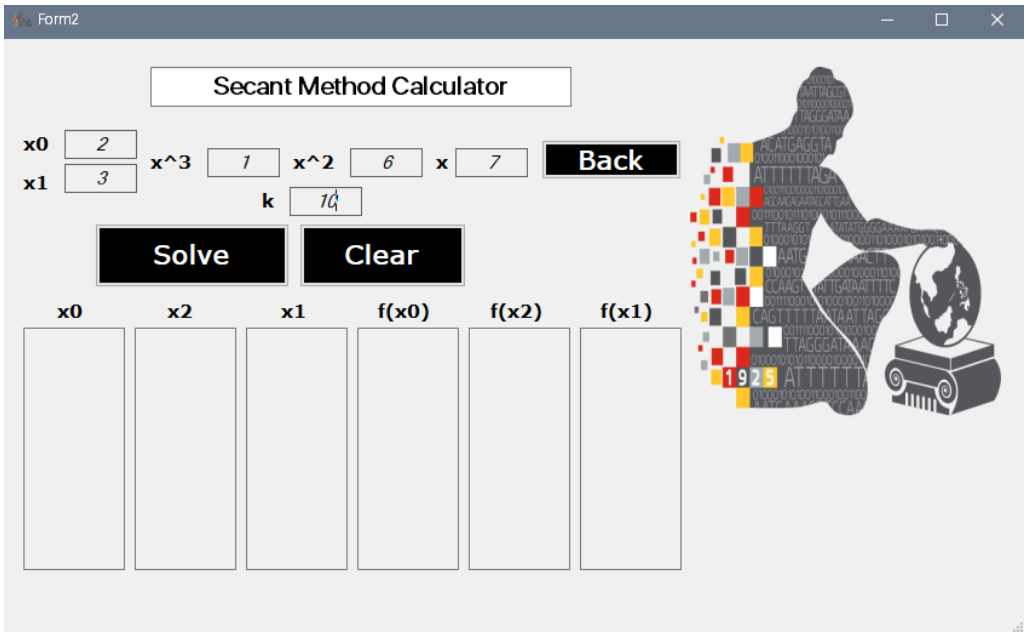


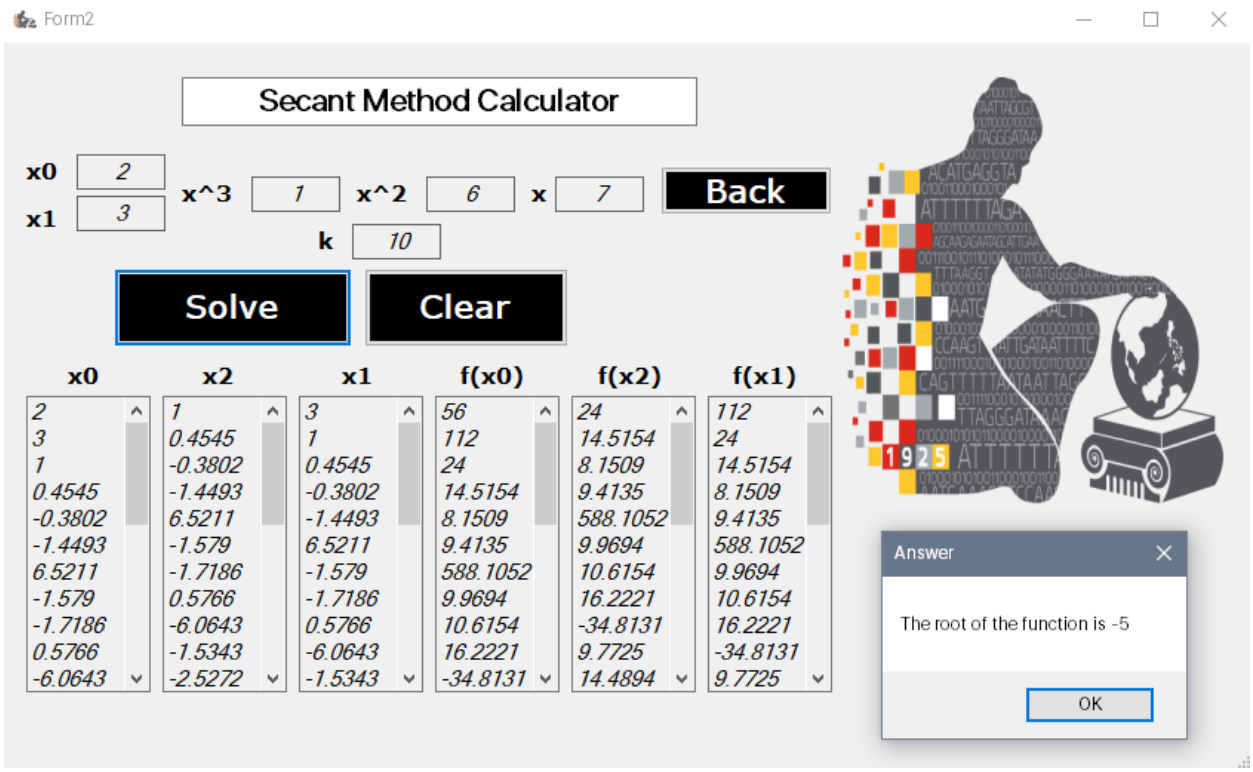
Figure 2. Window for Secant Method Program

The method available is shown in Fig. 2 is known as the Secant method. This is an open method which requires 2 initial guesses for the root like the Regula-Falsi method. This constructs an approximating straight line connecting the two functions of the initial values to estimate a root value. If the functions of x when computed are not equal, the 2 guesses are considered valid. The iterative formula is used from the previously found value until the terminating condition is met.

Steps to use the Program

1. Type in your chosen values denoted as x0 and x1.
2. Fill in the box for your coefficient of you function at x3, x2, x as well at the constant k.
3. Click solve.





4. The six tables below show how the iteration works towards the process.
5. A dialog box appears that displays the root of the function given.
6. You can click Clear to remove all the data inputted and Back to return to the main window.

Source Code:

SecantMain.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MachineProblem2
{
   public partial class SecantWindow : Form
   {
      public SecantWindow()
      {
         InitializeComponent();
      }

      private void SecantButton_Click(object sender, EventArgs e)
      {
         MachineProblem2.frmSecant form = new MachineProblem2.frmSecant();
         form.ShowDialog();
      }
   }
}
```

SecantCalc.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MachineProblem2
{
   public partial class frmSecant : Form
   {
      int i = 0, count = 0, bandila = 0;

      double eps = 0.0001;

      double[] coeff = new double[10];
      double x0, x1, x2 = 0;
      double fx0 = 0, fx1 = 0, fx2 = 0, temp = 0;

      private void btnSecant_back_Click(object sender, EventArgs e)
      {
         this.Close();
      }

      private void btnSecant_clear_Click(object sender, EventArgs e)
```

```
{
    txtBoxSecant_int1.Text = "";
    txtBoxSecant_int2.Text = "";
    txtBoxSecant_x3.Text = "";
    txtBoxSecant_x2.Text = "";
    txtBoxSecant_x1.Text = "";
    txtBoxSecant_k.Text = "";
    lBoxSecant_fx1.Items.Clear();
    lBoxSecant_fx0.Items.Clear();
    lBoxSecant_fx2.Items.Clear();
    lBoxSecant_x1.Items.Clear();
    lBoxSecant_x2.Items.Clear();
    lBoxSecant_x0.Items.Clear();
}

int test = 0;

public frmSecant()
{
    InitializeComponent();
}

public int check()
{

    x0 = double.Parse(txtBoxSecant_int1.Text);
    x1 = double.Parse(txtBoxSecant_int2.Text);

    fx0 = fx1 = fx2 = 0;
```

```
    for (i = 3; i >= 1; i--)
    {

        fx0 += coeff[i] * (Math.Pow(x0, i));
        fx1 += coeff[i] * (Math.Pow(x1, i));

    }

    fx0 += coeff[0];
    fx1 += coeff[0];

    if (fx0 == fx1)
    {

        MessageBox.Show("Initial Values are not meant to be!!");
        return (1);

    }
    return (0);

}

private void btnSecant_solve_Click(object sender, EventArgs e)
{
    coeff[0] = double.Parse(txtBoxSecant_k.Text);
    coeff[1] = double.Parse(txtBoxSecant_x1.Text);
    coeff[2] = double.Parse(txtBoxSecant_x2.Text);
    coeff[3] = double.Parse(txtBoxSecant_x3.Text);
```

```
do
{
   test = check();
   if (test != 0)
   {
      Close();
      break;
   }
} while (check() != 0);
bandila = 1;

if (bandila == 1)
{

   do
   {

      count++;
      fx0 = fx1 = fx2 = 0;

      lBoxSecant_x0.Items.Add(Math.Round(x0, 4));
      lBoxSecant_x1.Items.Add(Math.Round(x1, 4));

      for (i = 3; i >= 1; i--)
      {

         fx0 += coeff[i] * (Math.Pow(x0, i));
         fx1 += coeff[i] * (Math.Pow(x1, i));
```

```
    }

    fx0 += coeff[0];
    lBoxSecant_fx0.Items.Add(Math.Round(fx0, 4));

    fx1 += coeff[0];
    lBoxSecant_fx1.Items.Add(Math.Round(fx1, 4));

    temp = x2;

    x2 = x0 - ((fx0) * ((x1 - x0) / (fx1 - fx0)));
    lBoxSecant_x2.Items.Add(Math.Round(x2, 4));

    for (i = 3; i >= 1; i--)
    {

       fx2 += coeff[i] * (Math.Pow(x2, i));

    }
    fx2 += coeff[0];
    lBoxSecant_fx2.Items.Add(Math.Round(fx2, 4));

    x0 = x1;
    x1 = x2;
    fx2 = 0;

} while ((Math.Abs(temp - x2)) >= eps);
```

```
                MessageBox.Show("The root of the function is " + Math.Round(x2, 4), "Answer");
            }
        }
    }
}
```

References:

https://en.wikipedia.org/wiki/Secant_method

https://mat.iitm.ac.in/home/sryedida/public_html/caimna/transcendental/iteration%20methods/secant/secant.html