

State University of New York at Buffalo

Mechanical and Aerospace Engineering Department

MAE 550: OPTIMIZATION IN ENGINEERING DESIGN

HOMEWORK I

NAME: BABAK FARROKH
SEUNG KOOK JUN
LENG FENG LEE

DATE: 8 SEPT 2002.

PROBLEM STATEMENT:

Develop a 1-D search method. Efficiency will not be graded but only if you can find an accurate solution. The method should be creative. Can be coded in any programming language. Finally, the method should be tested on at least 4 examples from textbooks.

ABSTRACT:

The 1-D search method that we have created in this assignment is limited for a unimodal function that known to have one minimum value. We identify that, if the efficiency is not an issue to consider in the method, the main concern in the any method will be to identify the step size of our search of optimal point. The step size of the search can be chosen arbitrarily for most method. However, in our method, we determine the step size by first divided the range into two pieces and calculating the area under the two pieces (two steps) as the initial area, then again divided the area into three parts (three steps), calculating the area cover by this three parts, and compare this new area with the initial area. If the difference of the two areas is less than an allowable error, in our method, we assigned this error as 0.0000001; we then use the corresponding step size. If the value is larger then the error, we will further increase the steps and thus the step size will have a smaller value until we reach the allowable error. After this, an iteration process is then use to calculate the optimal point. We used C++ programming language to code our method.

ASSUMPTIONS:

Our 1-D search method is useful with the following assumptions:

1. The function given to find the minimum is a unimodal function. That is, the function has only one minimum in the region of our search. In this case, we can guarantee that the minimum that we found is the true minimum.
2. Second, the functions that we use for our method is limited to functions that are continuous and they have continuous first and second derivatives. [Garret 34]
3. We also assume that this is a constraint problem that the range that bound the minimum point is know and given.
4. The efficiency of this numerical optimization process is not in our concern. We are more interesting in accuracy in which we have some control over the error of the optimization process.

BACK GROUND OF THE IDEA.

We have discussed few methods before the final method is being used. Our main concept is that how do we determine the step size of the value of the variable in the function such that it gives an accurate result in our numerical optimization method. We can choose the step size arbitrarily with a very small value. However, this is not what we want. We used the idea of integration that divided an area under a curve into many pieces to calculate the area that it covered for a given range. Since the more pieces we use in the calculation, the more accurate result we will get compare to the true area. In this way, we can have some

control over the accuracy of the result by setting a allowable error in the comparison of our area calculation with the true area. When the pieces, that is, the number of steps that we use is sufficient enough, the area will close to the true value of the area. This is a fundamental thought behind the integration method used in calculus, which called Trapezoid method. This is how the idea of our method comes from.

THEORY AND SEARCH METHOD:

Our method consists of two steps. The first part of the method is finding the step size and the second part of the method is substitute the value start from the lower bound and calculate the function value with a increment of the step size that we have determine in the first step using a iteration process until the it reach the upper bound. The minimum value of the function is then stored and displayed in the output. The value of the step size, the value of variable at which the optimal value is obtain is also displayed.

The first problem that we need to figure out was to decide the step size. As we know, the value of the step size can be arbitrarily, we can choose a very small value as the step size. In any numerical optimization process, the processing time of the process is directly related to the step size of the variable. An inadequacy step size may lead to a answer that is far away from the actual optimal point or, the optimal point lied in between the step size. A demonstration of this problem can be shown in the following graph.

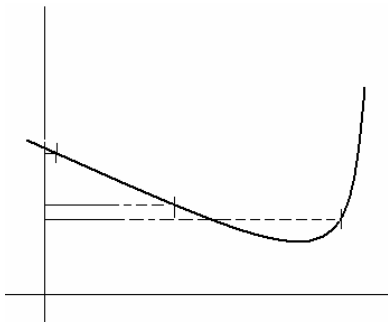


Figure 1. Example of inappropriate step size.
In this case, It in impossible to find minimum point

So our first task is to find a way to know function shape accurately. By this reason, we use trapezoidal method to decide optimal step size. If we divide step size enough to fit to the function, we can use this step size to calculate minimum point because it can be an optimal step size to describe function shape well.

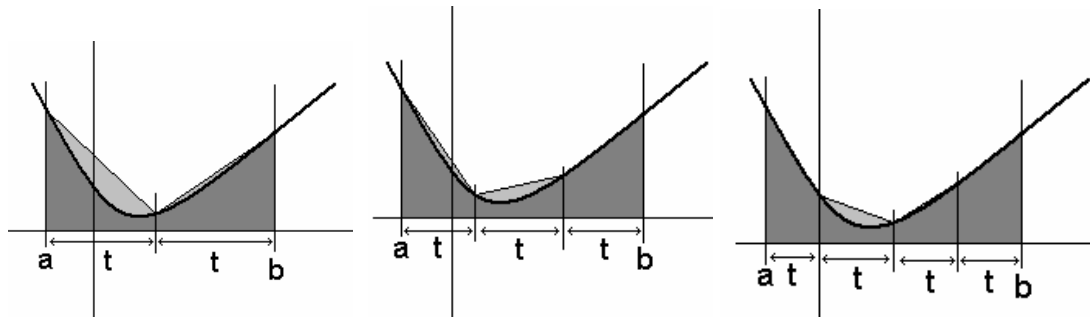


Figure 2. Trapezoid Method

Trapezoidal method uses trapezoid to get the integral. It was the basis of the integration method. So the smaller step size will guarantee more accurate answer. We use this property to decide step size. To get optimal step size our program keeps increasing number of "t" and checks that integration is converged or not.

Once the difference between old integration and current integration is smaller than a certain error limit – we set the limit as 0.0000001- the program stops iteration and starts to check minimum point with the step size from iteration.

THE ALGORITHM OF THE PROGRAM.

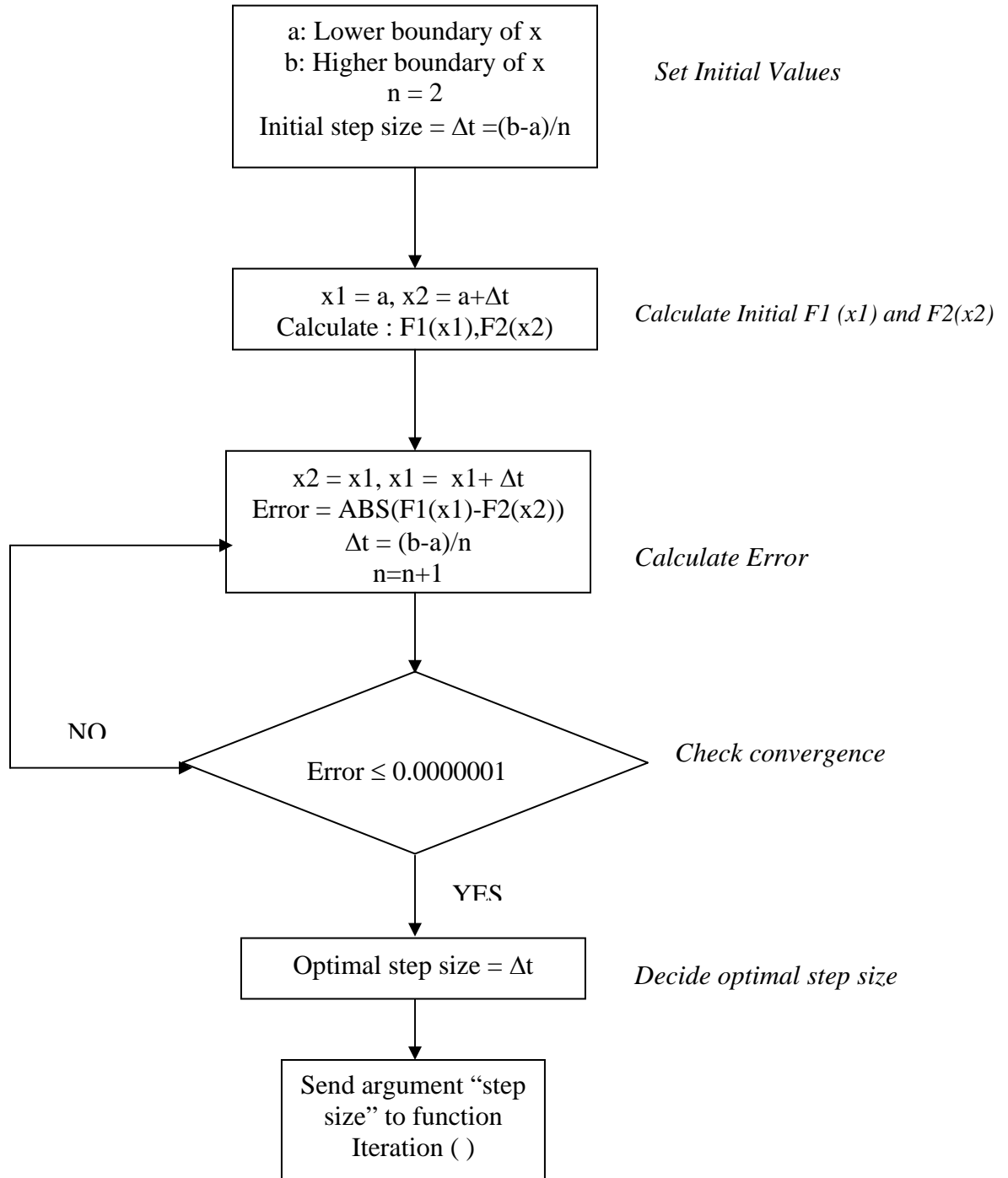


Figure 3. Flow chart of the main () in the program.

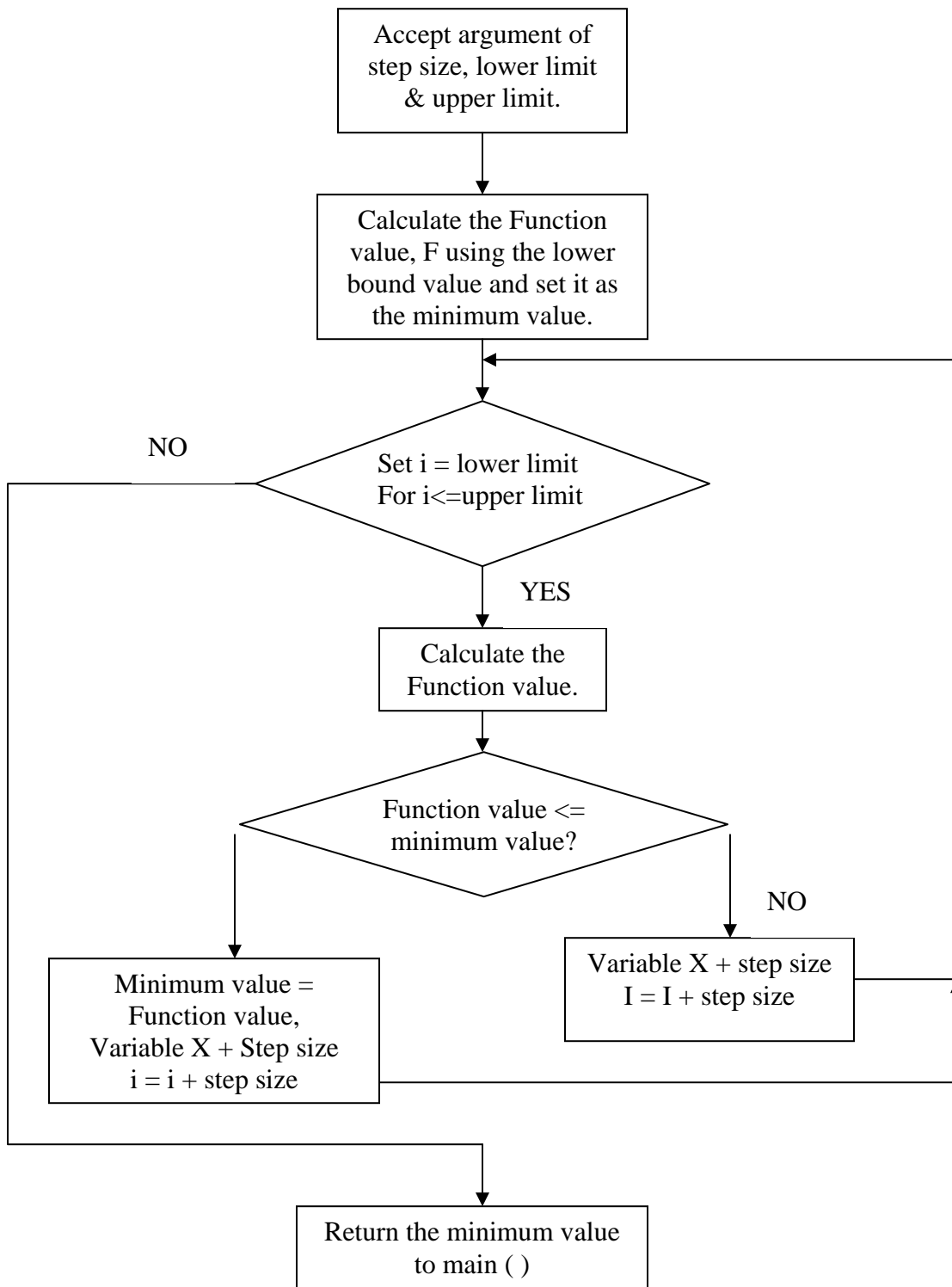


Figure 4. The flow chart showing the operation of the function Iteration().

CODING IN C++ PROGRAMMING LANGUAGE:

After we have decided the flow of the program, we coded the using C++ language. We then test our program for four examples as required in the question. To test each different function, we need only to change the equation in the program, the basic steps and algorithm is the same for each function. The following is a example of our program.

The following is a sample of our program. We test the program using 4 different equations. To do this, we need to change the equation in the program for each type of the functions. The following is a example of the code that we use in our search method with a equation of:

$$F = (x - 7)^2 + 2$$

```
//This is a program of the optimization process created for Assigment 1 of Courses MAE550,  
//Optimization in Engineering Design.  
//Copyright of Sk and LengFeng  
//Created on 7 Step 2002  
//  
//This program first determine the step size of the iteration by first dividing the  
//range given into two and calculate the area covered by the three point, then divided the  
//range into 4 points(4 steps) and calculate the area covered by the 5 points. The new area  
//calculated was then compared with the old calculate area with a lower points/steps.  
//If the difference of the two area is less then an assign error, e=0.0000001, the process  
//stop and using the final step size that we have now, we can start our iteration process  
//and calculating the minimum value of the Function F=(n-7)^2+2
```

```
#include<iostream.h>  
#include<math.h>
```

```
double Iteration(double stepsize1, double lowerlimit, double upperlimit);
```

```
void main() {
```

```
    double stepsize; // step size delta t  
    double a; // Left boundary of range  
    double b; // Right Boundary of range  
    double f1; // First value of function at x1  
    double f2; // Second value of function at x2  
    double area; // Area of trapezoid  
    double totarea; // Sume of areas - integral  
    double oldarea; // Old area  
    double deno; // Denominator to decide step size  
    int i; // Counter variable  
    double e; // Limitation of iteration error  
    double x1; // Independent variable of function  
    double x2; // Independent variable of function  
    double error; // Error between oldarea and current area  
    double result; // Final optimization value after the optimization process
```

```
    a=-5; // Set initial variable  
    b=5;  
    deno=2;  
    x1=a;
```

```

    totarea=0;
    oldarea=0;
    e=1;
    error = 0.000001;

    while(e>=error) {
        stepsize = (b-a)/deno;
        totarea=0;

        for (i=1;i<=deno;i++) {
            x2=x1+stepsize;

            f1=20-10*x1+x1*x1*x1*x1;
            f2=20-10*x2+x2*x2*x2*x2;

            area=((f1+f2)*stepsize)/2;

            x1=x1+stepsize;
            totarea=totarea+area;
        }
        if(totarea-oldarea <0)
        {
            e= -1*(totarea-oldarea);
        }
        else{
            e=totarea-oldarea;
        }

        x1=a;
        oldarea=totarea;
        deno=deno+1;
    }

    cout << "The error, e =" << e << "\n" << "Stepsize=" << stepsize << "\n";
    result =Iteration(stepsize, a, b);
    cout << "The value of the minimum using this optimization process is = "
        <<result <<"\n\n\a";

    return;
}

double Iteration(double stepsize1, double LowerLimit, double UpperLimit)
{
    double F;
    double X,Min;
    double i,minX;

    X=LowerLimit;           //we want to start calculating F from the lower bound
    minX=X;
    Min=20-10*X+X*X*X*X;    //first let min be the value of F using the lower bound as x

    for (i=LowerLimit;i<=UpperLimit;i+i+stepsize1) //calculating step by step of Function F
                                                //with a increment of the step size
    {
        F=20-10*X+X*X*X*X;    //calculate the value of Function F

```

```

    if (F<= Min)                                //if the F value calculated is less than the min
    {
        Min=F;                                  //then F will assigned to the min value
        minX=X;
        X=X+stepsize1;                          //increment of x for one stepsize
    }

    else
        X=X+stepsize1;                          //if F value is less then the min, do nothing
                                                //calculate another value of F with increment of X
}

cout << "\nUsing the calculated step size of " <<stepsize1 <<"," <<"\n";
cout << "The Minimum value occur at X= " <<minX <<"\n";

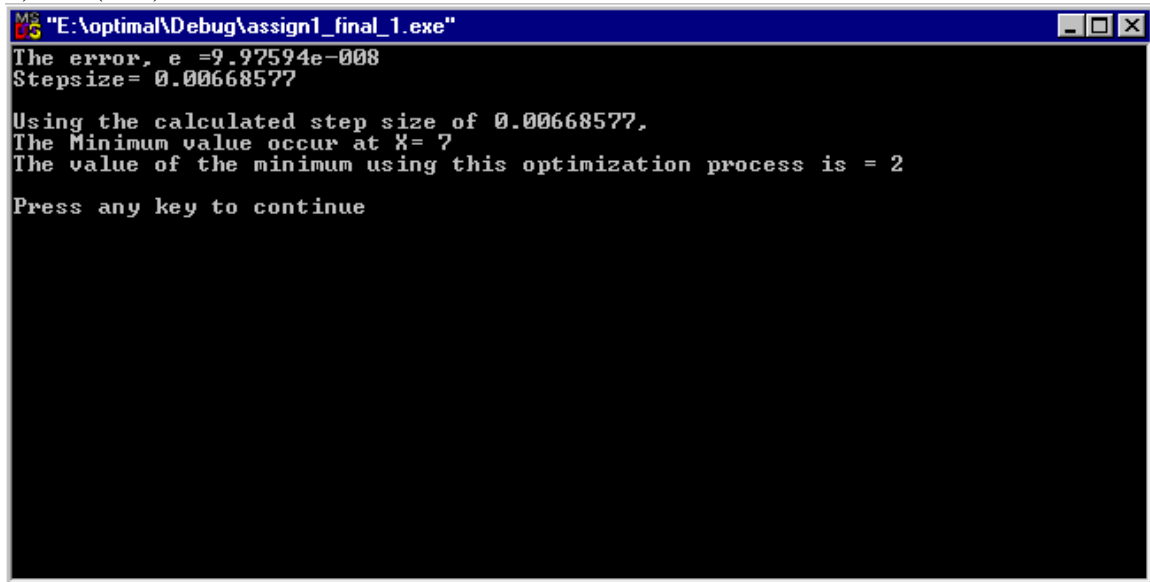
return Min;                                    //return the value to main ()
}

```

Figure 5. Sample coding for the search method.

RESULT AND TESTING.

1) $F = (X-7)^2 + 2$

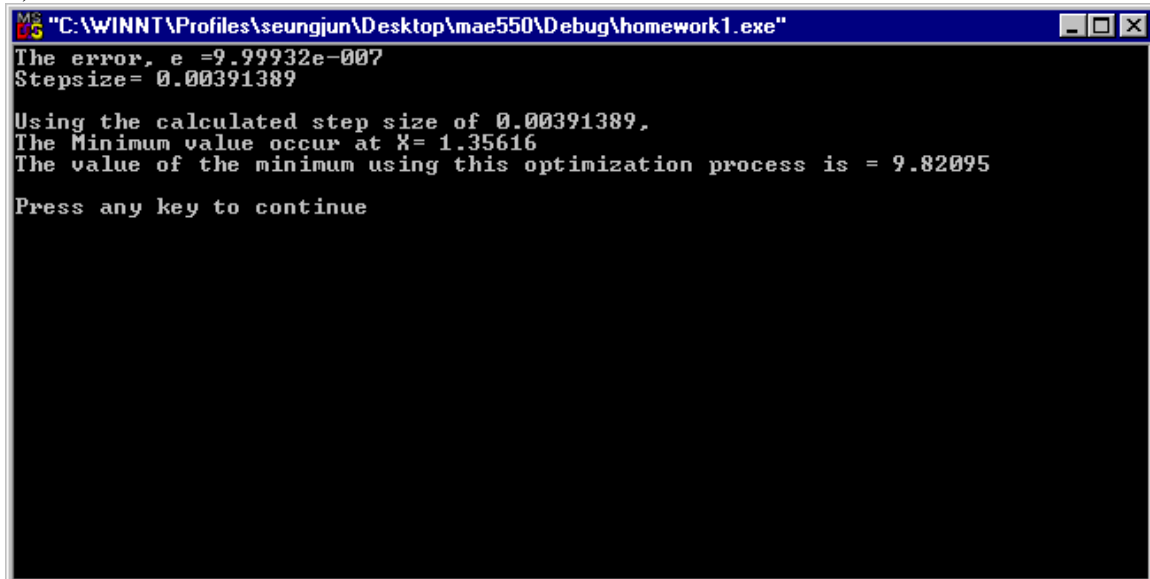


```
MS-DOS "E:\optimal\Debug\assign1_final_1.exe"
The error, e =9.97594e-008
Stepsize= 0.00668577

Using the calculated step size of 0.00668577,
The Minimum value occur at X= 7
The value of the minimum using this optimization process is = 2

Press any key to continue
```

2) $F = 20 - 10X + X^4$

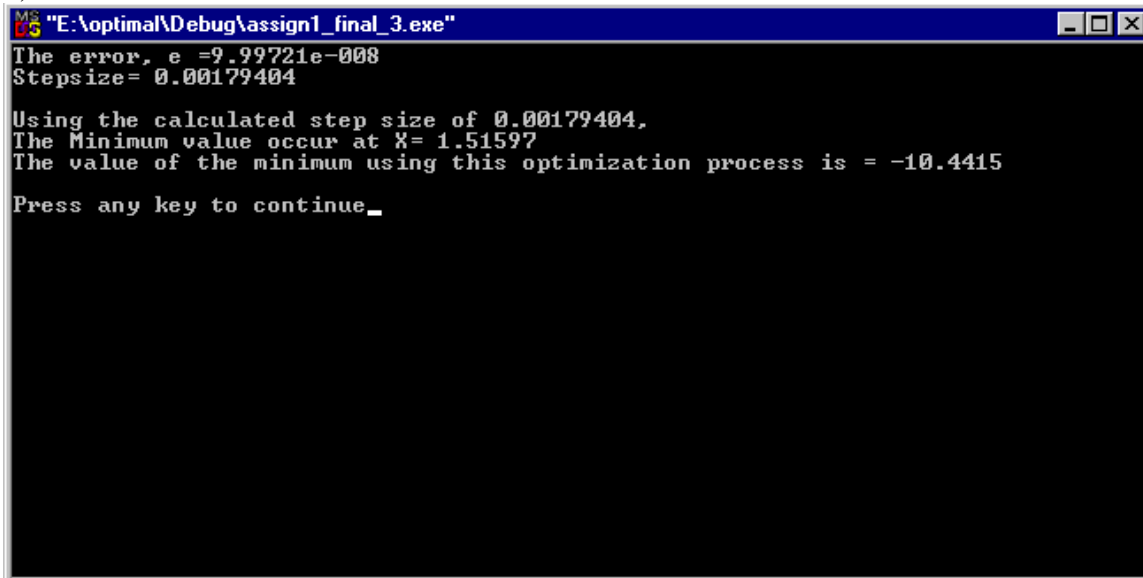


```
MS-DOS "C:\WINNT\Profiles\seungjun\Desktop\mae550\Debug\homework1.exe"
The error, e =9.99932e-007
Stepsize= 0.00391389

Using the calculated step size of 0.00391389,
The Minimum value occur at X= 1.35616
The value of the minimum using this optimization process is = 9.82095

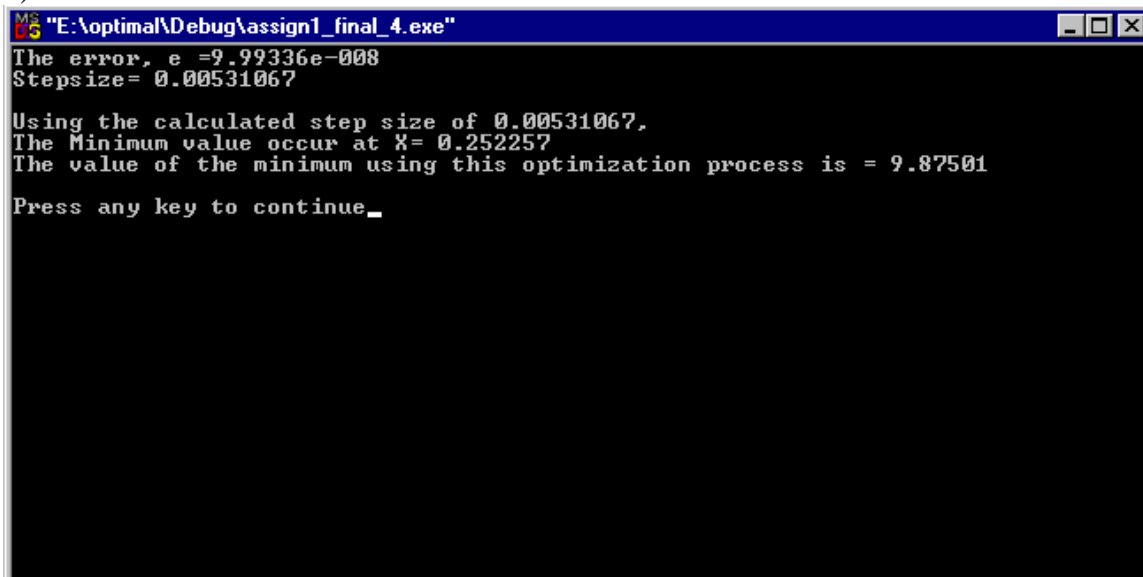
Press any key to continue
```

3) $F = 10 - 20X + 2X^2 + X^4$



```
"E:\optimal\Debug\assign1_final_3.exe"  
The error, e =9.99721e-008  
Stepsize= 0.00179404  
Using the calculated step size of 0.00179404.  
The Minimum value occur at X= 1.51597  
The value of the minimum using this optimization process is = -10.4415  
Press any key to continue_
```

4) $F = 10 - X + 2X^2$



```
"E:\optimal\Debug\assign1_final_4.exe"  
The error, e =9.99336e-008  
Stepsize= 0.00531067  
Using the calculated step size of 0.00531067.  
The Minimum value occur at X= 0.252257  
The value of the minimum using this optimization process is = 9.87501  
Press any key to continue_
```

Conclusion

According to results, this method found minimum points with enough accuracy – we compared results with answers from calculator which has minimum finding function.

Although optimal step size is decided by this method, we must set the error limit to decide step size but it is totally different from choose step size arbitrarily. The step size which came from this method is based on the curve fitting of trapezoidal method so this can be more trustful.