Heaven's Light is Our Guide

Rajshahi University of Engineering & Technology

Department of Computer Science & Engineering

Lab Report

Topic : Least-squares Curve Fitting Procedures

Course Title : Sessional Based on CSE 2103
Course Code : CSE 2104

Submitted By :
Md. Tanzid Hasan
Section : A
Roll No : 1603054

Submitted To :
Shyla Afroge
Assistant Professor
Dept. of Computer Science & Engineering

Date : 02-04-2018

Experiment No : 01

Experiment Name : Fitting a straight line

Theory :

Let, $Y = a_0 + a_1x$ be the straight line to be fitted to the given data. Here, error, $e_i = y_i - f(x_i)$. If we write,

$$\begin{aligned} S &= e_1^2 + e_2^2 + \ldots\ldots + e_m^2 \\ &= [y_1 - f(x_1)]^2 + [y_2 - f(x_2)]^2 + \ldots\ldots + [y_m - f(x_m)]^2 \\ &= [y_1 - (a_0 + a_1x_1)]^2 + [y_2 - (a_0 + a_1x_2)]^2 + \ldots\ldots + [y_m - (a_0 + a_1x_m)]^2 \quad \ldots(1) \end{aligned}$$

For S to be minimum, we have

$$\frac{\partial S}{\partial a_0} = 0 = -2[y_1 - (a_0 + a_1x_1)] - 2[y_2 - (a_0 + a_1x_2)] - \cdots - 2[y_m - (a_0 + a_1x_m)] \quad \ldots(2)$$

and

$$\frac{\partial S}{\partial a_1} = 0 = -2x_1[y_1 - (a_0 + a_1x_1)] - 2x_2[y_2 - (a_0 + a_1x_2)] - \cdots - 2x_m[y_m - (a_0 + a_1x_m)]. \quad \ldots\ldots\ldots(3)$$

The equation is simplify to

$$ma_0 + a_1(x_1 + x_2 + \ldots\ldots + x_m) = y_1 + y_2 + \ldots\ldots + y_m \quad \ldots\ldots\ldots(4)$$

and

$$a_0(x_1 + x_2 + \ldots. + x_m) + a_1(x_1^2 + x_2^2 + \ldots\ldots + x_m^2) = x_1y_1 + x_2y_2 + ... + x_3y_3 \quad ...(5)$$

or, more compactly to

$$ma_0 + a_1\sum_{i=1}^{m} x_i = \sum_{i=1}^{m} y_i \quad .....(6)$$

and

$$a_0 + \sum_{i=1}^{m} x_i + a_1\sum_{i=1}^{m} x_i^2 = \sum_{i=1}^{m} x_iy_i. \quad \ldots\ldots(7)$$

Since the x_I and y_i are known quantities, Eqs. (4),(5) or (6),(7) called the normal equations, can be solved for the two unknown a_0 and a_1.

Differentiating Eqs. (2) and (3) with respect to a_0 to a_1 respectively, we find that $\partial^2 S/\partial a_0^2$ and $\partial^2 S/\partial a_1^2$ will both be positive at the points a_0 and a_1. Hence these values provide a minimum of S.

Further, dividing Eqs. (6) throughout by m, we obtain

$$a_0 + a_1\bar{x} = \bar{y}$$

where $(\bar{x},\bar{y})$ is the centroid of the given data points. It follows that the fitted straight line passes through the centroid of the data points. The following example demonstrates the working of this method.

Code :

```
#include <bits/stdc++.h>
using namespace std;
double f(double b0,double b1,double x)
{
   double y=b0+(b1*x);
   return y;
}
int main()
{
   int n;
   double sumx=0,sumy=0,sumxx=0,sumxy=0;
   cout<<"How many element you want to input : ";
   cin>>n;
   double x[n];
   double y[n];
   for(int i=1;i<=n;i++)
   {
      cout<<"x1 : ";
      cin>>x[i];
      sumx=sumx+x[i];
      sumxx=sumxx+(x[i]*x[i]);
```

```
        cout<<"y1 : ";
        cin>>y[i];
        cout<<endl;
        sumy=sumy+y[i];
        sumxy=sumxy+(x[i]*y[i]);
    }
    double d=((n*sumxx)-(sumx*sumx));
    double a0=((sumxx*sumy)-(sumx*sumxy))/d;
    double a1=((sumxy*n)-(sumx*sumy))/d;
    cout<<"a0 : "<<a0<<ends<<ends<<"a1 : "<<a1<<endl;
    double err[n];
    cout<<endl<<"Error : "<<endl;
    for(int i=1;i<=n;i++)
    {
        err[i]=y[i]-f(a0,a1,x[i]);
        cout<<"For "<<x[i]<<", "<<"error : "<<err[i]<<endl;
    }
    return 0;
}
```

Input :

How many element you want to input : 6

x1 : 20

y1 : 800.3

x1 : 30

y1 : 800.4

x1 : 40

y1 : 800.6

x1 : 50

y1 : 800.7

x1 : 60

y1 : 800.9

x1 : 70

y1 : 801.0

Output :

a0 : 799.994 a1 : 0.0145714

Error :

For 20, error : 0.0142857

For 30, error : -0.0314286

For 40, error : 0.0228571

For 50, error : -0.0228571

For 60, error : 0.0314286

For 70, error : -0.0142857

Discussion : In this program, the temperatures were taken as x and corresponding pressure was taken as y. Then the values were calculated as the algorithm. The errors were calculated from difference of y – f(x).

Experiment No : 02

Experiment Name : Fitting an exponential curve

Theory :

Let, the curve

$$y = a_0 e^{a_1 x} \quad \text{.....(1)}$$

Be fitted to the given data. Then, as before, taking logarithms of both sides of Eqs. (1), we get,

$$\log y = \log a_0 + a_1 x \quad \text{.....(2)}$$

which can be written in the form

$$Z = A + Bx,$$

where $Z = \log y$, $A = \log a_0$ and $B = a_1$. The problem therefore reduces to finding a least-squares straight line through the given data.

Code :

```
#include <bits/stdc++.h>
using namespace std;
double f(double b0,double b1,double x)
{
   double y=exp(b0+(b1*x));
   return y;
}
int main()
{
   int n;
   double sumx=0,sumy=0,sumxx=0,sumxy=0;
   cout<<"How many element you want to input : ";
   cin>>n;
   double x[n];
   double y[n];
   for(int i=1;i<=n;i++)
```

```
    {
        cout<<"x1 : ";
        cin>>x[i];
        sumx=sumx+x[i];
        sumxx=sumxx+(x[i]*x[i]);
        cout<<"y1 : ";
        cin>>y[i];
        cout<<endl;
        sumy=sumy+log(y[i]);
        sumxy=sumxy+(x[i]*log(y[i]));
    }
    double d=((n*sumxx)-(sumx*sumx));
    double a0=exp(((sumxx*sumy)-(sumx*sumxy))/d);
    double a1=((sumxy*n)-(sumx*sumy))/d;
    cout<<"a0 : "<<a0<<ends<<ends<<"a1 : "<<a1<<endl;
    double err[n];
    cout<<endl<<"Errors : "<<endl;
    for(int i=1;i<=n;i++)
    {
        err[i]=y[i]-f(log(a0),a1,x[i]);
        cout<<"For "<<x[i]<<", "<<"error : "<<err[i]<<endl;
    }
    return 0;
}
```

Input :

How many element you want to input : 5

x1 : 2

y1 : 4.077

x1 : 4

y1 : 11.084

x1 : 6

y1 : 30.128

x1 : 8

y1 : 81.897

x1 : 10

y1 : 222.62

Output :

a0 : 1.4999 a1 : 0.500008

Errors :

For 2, error : -0.000221058

For 4, error : 0.000776285

For 6, error : 0.000163863

For 8, error : -0.000337236

For 10, error : -0.00381593

Discussion : In this program, x and y were taken by the method of least squares such that $y = ae^{bx}$. Then the values were calculated as the algorithm. The errors were calculated from difference of y – f(x).