LEAST SQUARE FITTING

Introduction

The method of least squares is a standard approach to the approximate solution of over-determined systems, i.e. sets of equations in which there are more equations than unknowns.

"Least squares" means that the overall solution minimizes the sum of the squares of the errors made in solving every single equation.

Least squares problems fall into two categories: linear or non-linear least squares.

**Example Linear Fit**

The best fit in the least-squares sense minimizes the sum of squared residuals defined by:

$$ r_i = y_i - \hat{y}_i $$

The summed square of residuals is given by

$$ S = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 $$

where $n$ is the number of data points included in the fit and $S$ is the sum of squares error estimate.
Linear Least Square Method

Consider we want to fit the data \((x_i, y_i)\) to a function

\[ y = ax + b \]

then the square sum of the residuals is:

\[
S = \sum_{i=1}^{n} (y_i - ax_i - b)^2
\]

The aim is to get parameters \(a\) and \(b\) by minimizing \(S\). To do that, we should solve the following equations simultaneously:

\[
\frac{\partial S}{\partial a} = 0 \quad \frac{\partial S}{\partial b} = 0
\]

Solutions are:

\[
a = \frac{n \sum x_i y_i - (\sum x_i)(\sum y_i)}{n \sum x_i^2 - (\sum x_i)^2}
\]
\[
b = \frac{(\sum x_i^2)(\sum y_i) - (\sum x_i)(\sum x_i y_i)}{n \sum x_i^2 - (\sum x_i)^2}
\]
Deriving the errors is a bit more tedious, you end up with

**ADD ERROR EQUATIONS HERE**

Finally, to test the goodness of the fit, we can define

\[
    r^2 = \frac{S - S_i}{S_i}
\]

where

\[
    S = \sum_{i=1}^{n} (y_i - ax_i - b)^2 \quad \text{and} \quad S_i = \sum_{i=1}^{n} (y_i - y_m)^2 \quad \text{with} \quad y_m = \frac{1}{n} \sum_{i=1}^{n} y_i
\]

**EXAMPLE**

The distance \( (d) \) required to stop an automobile is a function of its speed \( (v) \). The data is collected to get this relationship.

Fit the data to a linear function and compute the goodness of the fit.

```
// Computer program -----------------------------
int main(){
    const int n = 6;
    double x[n] = {24, 32, 40, 48, 64, 80};
    double y[n] = {4.8, 6.0, 10.2, 12.0, 18.0, 27.0};
    double sx=0,sy=0,sxx=0,sxy=0,s=0,st=0,r2,a,b;
    for(int i=0; i<n; i++){
        sx += x[i]; sxx += x[i]*x[i];
        sy += y[i]; sxy += x[i]*y[i];
    }
    a = (n*sxy-sx*sy)/(n*sxx-sx*sx);
    b = (sxx*sy-sx*sxy)/(n*sxx-sx*sx);
    for(int i=0; i<n; i++){
        s += pow(y[i]-a*x[i]-b,2.0);
        st += pow(y[i]-sx/n,2.0);
    }
    r2 = (st-s)/st;
    cout << a << "\n" << b << endl;
    cout << s << "\n" << r2 << endl;
} // main
```

\[ S \rightarrow 0 \]

\[ r^2 \rightarrow 1 \]
Weighted Linear Least Square Method

Weighted least squares regression minimizes the error estimate:

$$S = \sum_{i=1}^{n} w_i (y_i - \hat{y}_i)^2$$

where $w_i$ are the weights which determine how much each response value influences the final parameter estimates. If you know the variances of your data, then the weights are given by:

$$w_i = \frac{1}{\sigma_i^2}$$

**EXAMPLE**

The distance ($d$) required to stop an automobile is a function of its speed ($v$). The data is collected to get this relationship.

The \(+-\) value represents the measurement error (one standard deviation). Fit the data to a linear function and compute the goodness of the fit.

<table>
<thead>
<tr>
<th>$v$ (km/h)</th>
<th>$d$ (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>4.8 +- 0.3</td>
</tr>
<tr>
<td>32</td>
<td>6.0 +- 0.4</td>
</tr>
<tr>
<td>40</td>
<td>10.2 +- 1.0</td>
</tr>
<tr>
<td>48</td>
<td>12.0 +- 1.1</td>
</tr>
<tr>
<td>64</td>
<td>18.0 +- 1.4</td>
</tr>
<tr>
<td>80</td>
<td>27.0 +- 1.5</td>
</tr>
</tbody>
</table>
Non-Linear Fit

In general, error estimate (S) can be written as:

$$S(a) = \sum_{i=1}^{n} w_i (y_i - f(x_i, a))^2 = \sum_{i=1}^{n} \left( \frac{y_i - f(x_i, a)}{\sigma_i} \right)^2$$

where $a$ is a vector of coefficients: $a = \{a_1, a_2, \ldots, a_m\}$.

To minimize $S$, we should solve the following $m$ equations simultaneously:

$$\frac{\partial S}{\partial a_j} = 0 \quad j = 1, 2, \ldots, m$$

To get the solution, one can apply the following iterative method:

$$a_{i+1} = a_i - H_i^{-1} \nabla S_i$$

where

$$a = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}, \quad H = \begin{pmatrix} \frac{\partial^2 S}{\partial a_1^2} & \frac{\partial^2 S}{\partial a_1 \partial a_2} & \cdots & \frac{\partial^2 S}{\partial a_1 \partial a_m} \\ \frac{\partial^2 S}{\partial a_2 \partial a_1} & \frac{\partial^2 S}{\partial a_2^2} & \cdots & \frac{\partial^2 S}{\partial a_2 \partial a_m} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 S}{\partial a_m \partial a_1} & \frac{\partial^2 S}{\partial a_m \partial a_2} & \cdots & \frac{\partial^2 S}{\partial a_m^2} \end{pmatrix}, \quad \nabla S = \begin{pmatrix} \frac{\partial S}{\partial a_1} \\ \frac{\partial S}{\partial a_2} \\ \vdots \\ \frac{\partial S}{\partial a_m} \end{pmatrix}$$