

# Least Squares Data Fitting

## Linear LSQ

*A common method for finding the best straight line fit to a series of measured points. (linear regression)*

Suppose we have  $n$  pairs of measurements  $(x_1, y_1)$ ,  $(x_2, y_2)$ ,  $\dots$ ,  $(x_n, y_n)$  and that the errors are entirely in the values of  $y$ . We want to obtain the best fit to the linear function

$$y = mx + b . \quad \left( y = \sum_{k=0}^p a_k x^k \right)$$

This problem is “equivalent” to find  $m$  and  $b$  that minimizes the following function

$$\chi^2 = \sum_{i=1}^n (y_i - mx_i - b)^2 .$$

$$\Rightarrow \frac{\partial \chi^2}{\partial m} = -2 \sum x_i (y_i - mx_i - b) = 0 ,$$

$$\frac{\partial \chi^2}{\partial b} = -2 \sum (y_i - mx_i - b) = 0 .$$

*Linear equation:*

$$m \sum x_i^2 + b \sum x_i = \sum x_i y_i$$

$$m \sum x_i + b \sum 1 = \sum y_i .$$

Define


$$\begin{aligned}\bar{x} &= \frac{1}{n} \sum_{i=1}^n x_i & \bar{y} &= \frac{1}{n} \sum_{i=1}^n y_i \\ \overline{x^2} &= \frac{1}{n} \sum_{i=1}^n x_i^2 & \overline{xy} &= \frac{1}{n} \sum_{i=1}^n x_i y_i ,\end{aligned}$$

we obtain

$$\begin{aligned}m &= \frac{\overline{xy} - \bar{x}\bar{y}}{(\Delta x)^2} & (\Delta x)^2 &= \overline{x^2} - \bar{x}^2 \\ b &= \bar{y} - m\bar{x}\end{aligned}$$

### Weighted LSQ

More generally, we minimize the quantity

$$\chi^2 = \sum_{i=1}^n w_i (y_i - mx_i - b)^2 .$$


We need only to replace the average by

$$\bar{f} = \frac{1}{n\bar{w}} \sum_{i=1}^n w_i f_i \qquad \bar{w} = \frac{1}{n} \sum_{i=1}^n w_i .$$

## Estimate Uncertainties in $m$ and $b$

If there is error associated only with the measurement of  $y$ , but not  $x$ , then we should consider the so called *Chi-Square Fitting* by minimize the following quantity

$$\chi^2 \equiv \sum_i \left( \frac{y_i - mx_i - b}{\sigma_i} \right)^2 ,$$

where  $\sigma_i$  is the standard deviation for each measurement. In particular,

$$\sigma_i^2 = \sigma_y^2 = \frac{1}{n-2} \sum_{i=1}^n (y_i - mx_i - b)^2 .$$

Similarly, we have

$$\begin{aligned} \frac{\partial \chi^2}{\partial m} &= -2 \sum \frac{x_i (y_i - mx_i - b)}{\sigma_i^2} = 0 , \\ \frac{\partial \chi^2}{\partial b} &= -2 \sum \frac{(y_i - mx_i - b)}{\sigma_i^2} = 0 . \end{aligned}$$

Define

$$\begin{aligned} S &\equiv \sum_{i=1}^n \frac{1}{\sigma_i^2} & S_x &\equiv \sum_{i=1}^n \frac{x_i}{\sigma_i^2} & S_y &\equiv \sum_{i=1}^n \frac{y_i}{\sigma_i^2} \\ S_{xx} &\equiv \sum_{i=1}^n \frac{x_i^2}{\sigma_i^2} & S_{xy} &\equiv \sum_{i=1}^n \frac{x_i y_i}{\sigma_i^2} , \end{aligned}$$

we have the *Linear equation*:

$$\begin{aligned} mS_{xx} + bS_x &= S_{xy} \\ mS_x + bS &= S_y , \end{aligned}$$

and the solution ( $\Delta = SS_{xx} - S_x^2$ )

$$m = \frac{SS_{xy} - S_x S_y}{\Delta} \quad b = \frac{S_{xx} S_y - S_x S_{xy}}{\Delta}.$$

Propagation of errors in  $m$  is given by

$$\sigma_m^2 = \sum_{i=1}^n \sigma_i^2 \left( \frac{\partial m}{\partial y_i} \right)^2 \quad \frac{\partial m}{\partial y_i} = \frac{S x_i - S_x}{\sigma_i^2 \Delta}.$$

So the estimated uncertainty (the most probable error) in  $m$  is

$$\sigma_m^2 = \frac{S}{\Delta}.$$

Similarly, the estimated uncertainty in  $b$  is

$$\sigma_b^2 = \frac{S_{xx}}{\Delta}.$$

## Errors in Both Coordinates

The  $\chi^2$  function to consider is

$$\chi^2 \equiv \sum_i \frac{(y_i - m x_i - b)^2}{\sigma_{yi}^2 + m^2 \sigma_{xi}^2},$$

where  $\sigma_{xi}$  and  $\sigma_{yi}$  are, respectively, the  $x$  and  $y$  standard deviations for the  $i$ th data. This problem is considerably harder, for the resulting equations become *nonlinear*.

## Polynomial LSQ

$$p(x) = c_0 + c_1x + c_2x^2 + \cdots + c_mx^m$$

$$\chi^2 = \sum_{j=1}^n (p(x_j) - y_j)^2 \quad \text{or} \quad = \sum_{j=1}^n \frac{(p(x_j) - y_j)^2}{\sigma_j^2}$$

$$\frac{\partial \chi^2}{\partial c_k} = \sum_{j=1}^n 2x_j^k (c_0 + c_1x_j + \cdots + c_mx_j^m - y_j) = 0 .$$

$\Rightarrow m+1$  ( $k = 0, 1, \dots, m$ ) linear equations.

$$\begin{pmatrix} \Sigma(1) & \Sigma x_j & \Sigma x_j^2 & \cdots & \Sigma x_j^m \\ \Sigma x_j & \Sigma x_j^2 & \Sigma x_j^3 & \cdots & \Sigma x_j^{m+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \Sigma x_j^m & \Sigma x_j^{m+1} & \Sigma x_j^{m+2} & \cdots & \Sigma x_j^{m+m} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_m \end{pmatrix} = \begin{pmatrix} \Sigma y_j \\ \Sigma x_j y_j \\ \vdots \\ \Sigma x_j^m y_j \end{pmatrix}$$

In a more compact form, we write it as

$$\sum_{k=0}^m c_k \left[ \sum_{j=1}^n x_j^{i+k-1} \right] = \sum_{j=1}^n x_j^{i-1} y_j, \quad i = 1, 2, \dots, m$$

This is known as *normal equations* (symmetric), its determinant is nonzero if all  $x_j$ s are distinct.

*It is not true that the higher order  $m$ , the better fit!*

The system becomes increasingly ill-conditioned (numerically unstable) as  $m$  increases ( $m = 4$ ).

Example: the Hilbert matrix.

$$\begin{pmatrix} 1 & 1/2 & 1/3 & \cdots \\ 1/2 & 1/3 & 1/4 & \cdots \\ 1/3 & 1/4 & 1/5 & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

## Orthogonal Polynomials LSQ

Take a more general form

$$f(x) = c_0\phi_0(x) + c_1\phi_1(x) + \cdots + c_m\phi_m(x),$$

minimization of  $\chi^2$  leads to,  $i = 0, 1, \cdots, m$ ,

$$\sum_{k=0}^m c_k \left[ \sum_{j=1}^n \phi_k(x_j) \phi_i(x_j) \right] = \sum_{j=1}^n \phi_i(x_j) y_j, \quad (1)$$

where  $\phi_0(x), \cdots, \phi_m(x)$  are so chosen that the matrix of coefficients in Eq. (1) is not ill-conditioned.

There are optimal choices of these functions  $\phi_j(x)$ , with degree  $j - 1$  and with the coefficients matrix becoming *diagonal*.

That is,

$$\int_a^b dx \phi_i(x) \phi_j(x) = \delta_{ij}, \text{ (orthogonal polynomials)}$$

where we have changed summation into integration.

More generally, with weighting function  $w(x)$ ,

$$\int_a^b dx w(x) \phi_i(x) \phi_j(x) = \delta_{ij}.$$

$[a, b]$	$w(x)$	Symbol	Name
$[-1, 1]$	1	$P_n(x)$	Legendre
$[-1, 1]$	$(1 - x^2)^{-1/2}$	$T_n(x)$	Chebyshev I
$[-1, 1]$	$(1 - x^2)^{+1/2}$	$U_n(x)$	Chebyshev II
$[0, \infty)$	$e^{-x}$	$L_n(x)$	Laguerre
$(-\infty, \infty)$	$e^{-x^2}$	$H_n(x)$	Hermite

## General LSQ

Let  $y(x) = \sum_{k=1}^M a_k X_k(x)$

where  $X_1(x), \dots, X_M(x)$  are arbitrary fixed functions of  $x$ , called the *basis functions*.

Again, we define

$$\chi^2 = \sum_{i=1}^N \frac{(y_i - y(x_i))^2}{\sigma_i^2}$$

where  $\sigma_i$  is the measurement error (standard deviation) of the  $i$ th data point, presumed to be known. If the measurement errors are not known, they may all be set to the constant value  $\sigma = 1$ .

Define a  $N \times M$  matrix, called the *design matrix* of the fitting problem

$$A_{ij} = \frac{X_j(x_i)}{\sigma_i} \quad N \geq M$$

Also define vector  $\mathbf{b}$  (length  $N$ ) and vector  $\mathbf{a}$  (length  $M$ )

$$b_i = \frac{y_i}{\sigma_i} \quad \mathbf{a} = (a_1, a_2, \dots, a_M)^T$$

Minimization of  $\chi^2$  leads to ( $k = 1, \dots, M$ )

$$0 = \sum_{i=1}^N \frac{1}{\sigma_i^2} \left[ y_i - \sum_{j=1}^M a_j X_j(x_i) \right] X_k(x_i)$$

This is so called the *normal equations* of LSQ.

It could be written into a matrix form as

$$\sum_{j=1}^M \alpha_{kj} a_j = \beta_k \quad \Leftrightarrow \quad [\alpha] \cdot \mathbf{a} = [\beta]$$

where

$$\alpha_{kj} = \sum_{i=1}^N \frac{X_j(x_i) X_k(x_i)}{\sigma_i^2} \quad \Leftrightarrow \quad [\alpha] = \mathbf{A}^T \cdot \mathbf{A}$$

$$\beta_k = \sum_{i=1}^N \frac{y_i X_k(x_i)}{\sigma_i^2} \quad \Leftrightarrow \quad [\beta] = \mathbf{A}^T \cdot \mathbf{b}$$

Define the inverse matrix  $C_{jk} \equiv [\alpha]_{jk}^{-1}$ , we have

$$a_j = \sum_{k=1}^M C_{jk} \beta_k = \sum_{k=1}^M C_{jk} \sum_{i=1}^N \frac{y_i X_k(x_i)}{\sigma_i^2}$$

and that the variance associated with the estimate  $a_j$  is

$$\sigma^2(a_j) = \sum_{i=1}^N \sigma_i^2 \left( \frac{\partial a_j}{\partial y_i} \right)^2$$

with

$$\frac{\partial a_j}{\partial y_i} = \sum_{k=1}^M C_{jk} X_k(x_i) / \sigma_i^2$$

Consequently

$$\sigma^2(a_j) = \sum_{k=1}^M \sum_{l=1}^M C_{jk} C_{jl} \left[ \sum_{i=1}^N \frac{X_k(x_i) X_l(x_i)}{\sigma_i^2} \right] = C_{jj}$$



TABLE 3.2 Orthogonal Polynomials

$[a, b]$	$w(x)$	<i>Symbol</i>	<i>Name</i>
$[-1, 1]$	1	$P_n(x)$	Legendre
$[-1, 1]$	$(1 - x^2)^{-1/2}$	$T_n(x)$	Chebyshev I
$[-1, 1]$	$(1 - x^2)^{+1/2}$	$U_n(x)$	Chebyshev II
$[0, \infty)$	$e^{-x}$	$L_n(x)$	Laguerre
$(-\infty, \infty)$	$e^{-x^2}$	$H_n(x)$	Hermite

## Nonlinear Least Squares

Not all the problems that confront us are linear. Imagine that you have some experimental data that you believe should fit a particular theoretical model. For example, atoms in a gas emit light in a range of wavelengths described by the Lorentzian lineshape function

$$I(\lambda) = I_0 \frac{1}{1 + 4(\lambda - \lambda_0)^2 / \Gamma^2}, \quad (3.142)$$

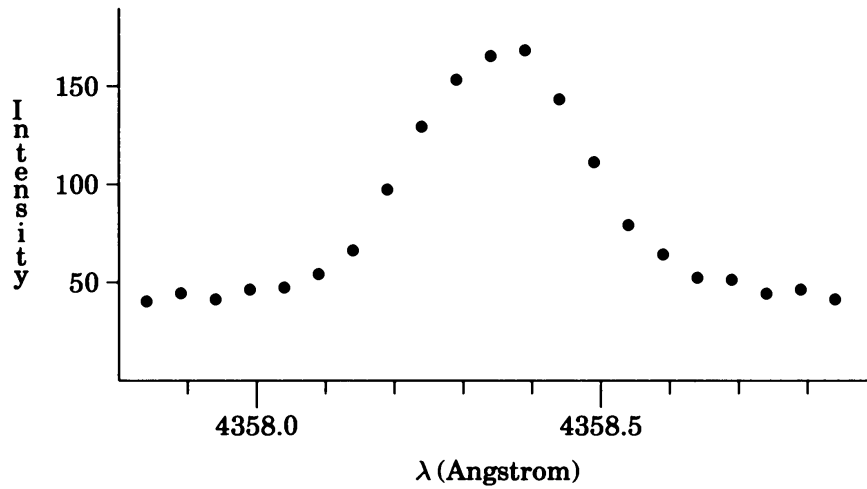
where  $\lambda$  is the wavelength of the light emitted,  $\lambda_0$  is the resonant wavelength,  $\Gamma$  is the width of the curve (full width at half maximum), and  $I_0$  is the intensity of the light at  $\lambda = \lambda_0$ . From measurements of  $I(\lambda)$ , which necessarily contain experimental noise, we're to determine  $\lambda_0$ ,  $\Gamma$ , and  $I_0$ . (Actually, the intensity is often measured in arbitrary units, and so is unimportant. The position and width of the curve, however, are both significant quantities. The position depends upon the atom emitting the light and hence serves to identify the atom, while the width conveys information concerning atom's environment, e.g., the pressure of the gas. )

Sample data of ambient room light, obtained by an optical multichannel analyzer, are presented in Figure 3.7. A diffraction grating is used to spread the light, which then falls upon a linear array of very sensitive electro-optical elements, so that each element is exposed to some (small) range of wavelengths. Each element then responds according to the intensity of the light falling upon it, yielding an entire spectrum at one time. Note that the data is not falling to zero away from the peak as would be suggested by our theoretical lineshape. This is typical of experimental data — a baseline must

also be established. We thus define our error as

$$S = \sum_{j=1}^N (I_j - B - I(\lambda_j))^2, \quad (3.143)$$

where  $I_j$  are the measured intensities. To obtain the normal equations, we minimize  $S$  with respect to  $B$ , the baseline, and the lineshape parameters  $\lambda_0$ ,  $\Gamma$ , and  $I_0$ , as before. However,  $I(\lambda)$  is not a linear function of  $\lambda_0$  and  $\Gamma$ , so that the normal equations we derive will be *nonlinear* equations, much more difficult than the linear ones we had earlier.



**FIGURE 3.7** A spectrum of ambient room light, in arbitrary units of intensity. (Data courtesy L.W. Downes, Miami University.)

Rather than pursue this approach, let's look at the problem from a different perspective. First, let's write the least squares error as

$$S(a_1, a_2, \dots, a_m) = \sum_{k=1}^N (y_k - Y(x_k; a_1, a_2, \dots, a_m))^2, \quad (3.144)$$

where  $y_k$  are the data at  $x_k$  and  $Y(x_k; a_1, a_2, \dots, a_m)$  is the proposed function evaluated at  $x_k$  and parameterized by the coefficients  $a_1, a_2, \dots, a_m$ . Our goal is to minimize  $S$  — why don't we simply vary the parameters  $a_i$  until we find a minimum? If we already have initial guesses  $a_1^0, a_2^0, \dots, a_m^0$  that lie near a minimum, then  $S$  will be approximately quadratic. Let's evaluate  $S$  at  $a_1^0 + h_1$ ,  $a_1^0$ , and  $a_1^0 - h_1$ , holding all the other parameters fixed. The minimum

of a quadratic interpolating polynomial through these points then gives us a better approximation to the minimum of  $S$ ,

$$a_1^1 = a_1^0 - \frac{h_1}{2} \frac{S(a_1^0 + h_1, \dots) - S(a_1^0 - h_1, \dots)}{S(a_1^0 + h_1, \dots) - 2S(a_1^0, \dots) + S(a_1^0 - h_1, \dots)}. \quad (3.145)$$

The process is then repeated for  $a_2, a_3, \dots, a_m$ . Admittedly, this is a crude procedure, but it works! (Sort of.)

We want to keep the programming as general as possible, so that we don't have to rewrite the routine to solve a different problem. In particular, the routine doesn't need to know much about  $S$  — only how many parameters are used, and the value of  $S$  at some specific locations. Crucial portions of the code might look something like the following:

```

Subroutine Minimize
double precision a(6),h(6)
integer M
*
* Get initial guesses for the parameters.
*
call init(M,a,h)

call crude(M,a,h)

end
*
*-----
*
Subroutine Init(m,a,h)
double precision a(6),h(6)
integer m,max
parameter ( max = 6)
*
* We need four parameters to determine the lineshape.
*
m = 4
*
* Check if properly dimensioned. If not, will need to
* modify the following array bounds:
*
In routine:      modify arrays:
*
minimize      a, h

```