Some things they don’t tell you about least squares fitting

“A mathematical procedure for finding the best-fitting curve to a given set of points by minimizing the sum of the squares of the offsets ("the residuals") of the points from the curve.”

Mathworld

Luis Valcárcel, McGill University

HEP Graduate Student Meeting

October 19, 2005
Overview

- Linear Least Squares Fitting (review)
- Non Linear Least Squares Fitting
- Why do we minimize the chi-square?
  - Connection with Maximum Likelihood principle
  - Vertical vs Perpendicular offsets
  - Robust estimation
- What about errors in the inputs?
  - Weighting errors in y
  - What to do with the errors in x?
- What about errors in the outputs?
  - How to calculate them?
  - How to interpret them?
- Program comparisons
Linear Least Squares Fitting (review)
Line Fitting

\[ R^2 (\alpha, \beta) = \sum_{i=1}^{n} [y_i - (\alpha + \beta x_i)]^2 \]

\[ \frac{\partial (R^2)}{\partial \alpha} = -2 \sum_{i=1}^{n} [y_i - (\alpha + \beta x_i)] = 0 \]

\[ \frac{\partial (R^2)}{\partial \beta} = -2 \sum_{i=1}^{n} [y_i - (\alpha + \beta x_i)] x_i = 0. \]

\[
\begin{bmatrix}
\sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \\
\sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2
\end{bmatrix}
\begin{bmatrix}
\alpha \\
\beta
\end{bmatrix}
=
\begin{bmatrix}
\sum_{i=1}^{n} y_i \\
\sum_{i=1}^{n} x_i y_i
\end{bmatrix}
\]

\[ \alpha = \frac{\sum_{i=1}^{n} x_i^2 y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sum_{i=1}^{n} x_i^2 - n \bar{x}^2} \]

\[ \beta = \frac{(\sum_{i=1}^{n} x_i y_i) - n \bar{x} \bar{y}}{\sum_{i=1}^{n} x_i^2 - n \bar{x}^2} \]

- Exact solution
- Implemented in scientific calculators
- Can even easily get the errors on the parameters
Polynomial Fitting

$$R^2 = \sum_{i=1}^{n} \left[ y_i - (\alpha_0 + \alpha_1 x_i + \ldots + \alpha_k x_i^k) \right]^2.$$  

$$\frac{\partial (R^2)}{\partial \alpha_k} = -2 \sum_{i=1}^{n} \left[ y_i - (\alpha_0 + \alpha_1 x_i + \ldots + \alpha_k x_i^k) \right] x_i^k = 0.$$  

\[
\begin{bmatrix}
\sum_{i=1}^{n} x_i^k & \sum_{i=1}^{n} x_i^{k-1} & \ldots & \sum_{i=1}^{n} x_i^0 \\
\sum_{i=1}^{n} x_i^k & \sum_{i=1}^{n} x_i^{k-2} & \ldots & \sum_{i=1}^{n} x_i^0 \\
\vdots & \vdots & \ddots & \vdots \\
\sum_{i=1}^{n} x_i^k & \sum_{i=1}^{n} x_i^{k+1} & \ldots & \sum_{i=1}^{n} x_i^0
\end{bmatrix}
\begin{bmatrix}
\alpha_0 \\
\alpha_1 \\
\vdots \\
\alpha_k
\end{bmatrix}
= 
\begin{bmatrix}
\sum_{i=1}^{n} y_i \\
\sum_{i=1}^{n} x_i y_i \\
\vdots \\
\sum_{i=1}^{n} x_i^k y_i
\end{bmatrix}.
\]

- Really just a generalization of the previous case
- Exact solution
- Just big matrices
General Linear Fitting

\[ y(x) = \sum_{k=1}^{M} a_k X_k(x) \]

\[ \chi^2 = \sum_{i=1}^{N} \left[ \frac{y_i - \sum_{k=1}^{M} a_k X_k(x_i)}{\sigma_i} \right]^2 \]

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

*normal equations* of the least squares problem

Can be put in matrix form and solved
Exponential Fitting

\[ y = A e^{Bx}, \]

\[ \ln y = \ln A + Bx. \quad \text{Linearize the equation and apply the fit to a straig} \]

\[ A \equiv \exp (\alpha) \]

\[ B \equiv b \]

\[ \alpha = \frac{\sum_{i=1}^{n} \ln y_i \cdot \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i \cdot \sum_{i=1}^{n} x_i \cdot \ln y_i}{n \cdot \sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2} \]

\[ b = \frac{n \cdot \sum_{i=1}^{n} x_i \ln y_i - \sum_{i=1}^{n} x_i \cdot \sum \ln y_i}{n \cdot \sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}, \]
Logarithmic Fitting

\[ y = \alpha + \hat{\beta} \ln x, \]

\[ \hat{\beta} = \frac{n \sum_{i=1}^{n} (y_i \ln x_i) - \sum_{i=1}^{n} y_i \sum_{i=1}^{n} \ln x_i}{n \sum_{i=1}^{n} (\ln x_i)^2 - (\sum_{i=1}^{n} \ln x_i)^2} \]

\[ \alpha = \frac{\sum_{i=1}^{n} y_i - \hat{\beta} \sum_{i=1}^{n} (\ln x_i)}{n}. \]
Power Law Fitting

\[ y = A x^B, \]

\[ A = e^a \]

\[ B = \hat{b} \]

\[ \hat{b} = \frac{n \sum_{i=1}^{n} (\ln x_i \ln y_i) - \sum_{i=1}^{n} (\ln x_i) \sum_{i=1}^{n} (\ln y_i)}{n \sum_{i=1}^{n} (\ln x_i)^2 - (\sum_{i=1}^{n} \ln x_i)^2} \]

\[ \alpha = \frac{\sum_{i=1}^{n} (\ln y_i) - \hat{b} \sum_{i=1}^{n} (\ln x_i)}{n} \]
Summary of Linear least squares fitting

• “The linear least squares fitting technique is the simplest and most commonly applied form of linear regression and provides a solution to the problem of finding the best fitting straight line through a set of points. In fact, if the functional relationship between the two quantities being graphed is known to within additive or multiplicative constants, it is common practice to transform the data in such a way that the resulting line is a straight line. For this reason, standard forms for exponential, logarithmic, and power laws are often explicitly computed. The formulas for linear least squares fitting were independently derived by Gauss and Legendre.” Mathworld
Non Linear Least Squares Fitting
Non linear fitting

• “For nonlinear least squares fitting to a number of unknown parameters, linear least squares fitting may be applied iteratively to a linearized form of the function until convergence is achieved. However, it is often also possible to linearize a nonlinear function at the outset and still use linear methods for determining fit parameters without resorting to iterative procedures. This approach does commonly violate the implicit assumption that the distribution of errors is normal, but often still gives acceptable results using normal equations, a pseudoinverse, etc. Depending on the type of fit and initial parameters chosen, the nonlinear fit may have good or poor convergence properties.” Mathworld.

• “We use the same approach as in previous sections, namely to define a $\chi^2$ merit function and determine best-fit parameters by its minimization. With nonlinear dependences, however, the minimization must proceed iteratively. Given trial values for the parameters, we develop a procedure that improves the trial solution. The procedure is then repeated until $\chi^2$ stops (or effectively stops) decreasing.” Numerical Recipes
\[ \chi^2 = \sum_i (y_i - y(x_i))^2 \]

\[ \frac{\partial \chi^2}{\partial a_j} = -2 \sum_i \left\{ (y_i - y(x_i)) \frac{\partial y(x_i)}{\partial a_j} \right\} = 0 \]

- Treat chi-squared as a continuous fct of the m parameters and search the m-dimensional space for the appropriate minimum value of chi-squared
- Apply to the m equations approximation methods for finding roots of coupled, nonlinear equations
- Use a combination of both methods

FIGURE 8.2
Chi-square hypersurface as a function of two parameters.
• Grid Search: Vary each parameter in turn, minimizing chi-squared with respect to each parameter independently. Many successive iterations are required to locate the minimum of chi-squared unless the parameters are independent.

• Gradient Search: Vary all parameters simultaneously, adjusting relative magnitudes of the variations so that the direction of propagation in parameter space is along the direction of steepest descent of chi-squared.

• Expansion Methods: Find an approximate analytical function that describes the chi-squared hypersurface and use this function to locate the minimum, with methods developed for linear least-squares fitting. Number of computed points is less, but computations are considerably more complicated.

• Marquardt Method: Gradient-Expansion combination

From Bevington and Robinson
• “What Minuit is intended to do.

Minuit is conceived as a tool to find the minimum value of a multi-parameter function and analyze the shape of the function around the minimum. The principal application is foreseen for statistical analysis, working on chi-square or log-likelihood functions, to compute the best-fit parameter values and uncertainties, including correlations between the parameters. It is especially suited to handle difficult problems, including those which may require guidance in order to find the correct solution.

• What Minuit is not intended to do

Although Minuit will of course solve easy problems faster than complicated ones, it is not intended for the repeated solution of identically parametrized problems (such as track fitting in a detector) where a specialized program will in general be much more efficient.

MINUIT documentation

• Careful with error estimation using MINUIT: Read their documentation.
• Also see “How to perform a linear fit” in ROOT documentation
Why do we minimize the chi-square?
Other minimization schemes

• Merit function: A function that measures the agreement between data and the fitting model for a particular choice of the parameters. By convention, this is small when agreement is good.

• MinMax problem: \[ \max \{ |y_i - (ax_i + b)| \} \]
  Requires advanced techniques

• Absolute deviation: \[ \sum |y_i - (ax_i + b)| \]
  Absolute value function not differentiable at zero! “Although the unsquared sum of distances might seem a more appropriate quantity to minimize, use of the absolute value results in discontinuous derivatives which cannot be treated analytically.” Mathworld

• Least squares: \[ \sum [y_i - (ax_i + b)]^2 \]
  Most convenient. “This allows the merit function to be treated as a continuous differentiable quantity. However, because squares of the offsets are used, outlying points can have a disproportionate effect on the fit, a property which may or may not be desirable depending on the problem at hand.” Mathworld.

• Least median squares, Maple
Connection with Maximum Likelihood principle

• “Given a particular set of parameters, what is the probability that this data set could have occurred?”
  • Intuition tells us that the data set should not be too improbable for the correct choice of parameters.
  • Identify the probability of the data given the parameters (which is a mathematically computable number), as the likelihood of the parameters given the data.
  • Find those values that maximize the likelihood
  • least-squares fitting is a maximum likelihood estimation of the fitted parameters if the measurement errors are independent and normally distributed with constant standard deviation.

\[
P \propto \prod_{i=1}^{N} \left\{ \exp \left[ -\frac{1}{2} \left( \frac{y_i - y(x_i)}{\sigma} \right)^2 \right] \Delta y \right\}
\]

\[
\left[ \sum_{i=1}^{N} \frac{(y_i - y(x_i))^2}{2\sigma^2} \right] - N \log \Delta y
\]

minimize over \(a_1 \ldots a_M\): \[
\sum_{i=1}^{N} [y_i - y(x_i; a_1 \ldots a_M)]^2
\]

\[
\chi^2 \equiv \sum_{i=1}^{N} \left( \frac{y_i - y(x_i; a_1 \ldots a_M)}{\sigma_i} \right)^2
\]
Vertical vs Perpendicular offsets

\[ R_\perp^2 = \sum_{i=1}^{n} \frac{[y_i - (\alpha + \beta x_i)]^2}{1 + \beta^2} \]

- “In practice, the *vertical* offsets from a line (polynomial, surface, hyperplane, etc.) are almost always minimized instead of the perpendicular offsets.
- This provides a much simpler analytic form for the fitting parameters.
- Minimizing \( R^2_{\perp} \) for a second- or higher-order polynomial leads to polynomial equations having *higher* order, so this formulation cannot be extended.
- In any case, for a reasonable number of noisy data points, the difference between vertical and perpendicular fits is quite small.”

Mathworld
Exponential Fitting Revisited

• Linearizing the equation like we did previously gives too much weight to small y values

• This is not the least squares approximation of the original problem

• Better to minimize another function or to treat the exact least squares problem nonlinearly

\[ \sum_{i=1}^{n} y_i (\ln y_i - \alpha - \beta x_i)^2. \]

\[ \alpha = \frac{\sum_{i=1}^{n} (x_i^2 y_i) \sum_{i=1}^{n} (y_i \ln y_i) - \sum_{i=1}^{n} (x_i y_i) \sum_{i=1}^{n} (x_i y_i \ln y_i)}{\sum_{i=1}^{n} y_i \sum_{i=1}^{n} (x_i^2 y_i) - (\sum_{i=1}^{n} x_i y_i)^2} \]

\[ \beta = \frac{\sum_{i=1}^{n} y_i \sum_{i=1}^{n} (x_i y_i \ln y_i) - \sum_{i=1}^{n} (x_i y_i) \sum_{i=1}^{n} (y_i \ln y_i)}{\sum_{i=1}^{n} y_i \sum_{i=1}^{n} (x_i^2 y_i) - (\sum_{i=1}^{n} x_i y_i)^2}. \]
Robust Estimation

- “Insensitive to small departures from the idealized assumptions for which the estimator is optimized.”
- Fractionally large departures for a small number of data points
- Can occur when measurement errors are not normally distributed
- General idea is that the weight given individual points should first increase with deviation, then decrease
- decide which estimate you want, that is, \( \rho \)
- Ex: if the errors are distributed as a double or two-sided exponential, namely

\[
P = \prod_{i=1}^{N} \left\{ \exp \left[ -\rho(y_i, y \{x_i; a\}) \right] \Delta y \right\}
\]

minimize over \( a \)

\[
\sum_{i=1}^{N} \rho \left( \frac{y_i - y(x_i; a)}{\sigma_i} \right)
\]

\[
\text{Prob} \{ y_i - y(x_i) \} \sim \exp \left( -\frac{y_i - y(x_i)}{\sigma_i} \right)
\]

\[
y(x; a, b) = a + bx
\]

\[
\sum_{i=1}^{N} |y_i - a - bx_i|
\]

\[
a = \text{median} \{ y_i - a - bx_i \}
\]

\[
0 = \sum_{i=1}^{N} x_i \text{sgn}(y_i - a - bx_i)
\]
What about errors in the inputs?
Weighting errors in $y$

- If the uncertainties are known, weight the distances with them:

$$\chi^2 \equiv \sum_{i=1}^{N} \left( \frac{y_i - y(x_i; a_1 \ldots a_M)}{\sigma_i} \right)^2$$

- What if the uncertainties are unknown? Use the chi-square to estimate them. But then, can’t use the chi-square to estimate goodness of fit:

$$\sigma^2 = \sum_{i=1}^{N} [y_i - y(x_i)]^2 / (N - M)$$
What to do with the errors in $x$?

- Trick: switch the $x$- and $y$-axis when the $x$ errors are bigger than the $y$ errors.
- Numerical Recipes:

$$\chi^2(a, b) = \sum_{i=1}^{N} \frac{(y_i - a - bx_i)^2}{\sigma_{y_i}^2 + b^2\sigma_{x_i}^2}$$

$$\text{Var}(y_i - a - bx_i) = \text{Var}(y_i) + b^2\text{Var}(x_i) = \sigma_{y_i}^2 + b^2\sigma_{x_i}^2 \equiv 1/w_i$$

$$a = \left[ \sum_i w_i(y_i - bx_i) \right] / \sum_i w_i \quad \frac{\partial \chi^2}{\partial a} = 0, \text{ is still linear} \quad \frac{\partial \chi^2}{\partial b} = 0 \text{ is nonlinear.}$$

- ROOT

$$\chi^2 = \frac{(y - f(x))^2}{\sigma_y^2 + ((f(x + \sigma_x) - f(x - \sigma_x))/2)^2}$$

- LSM program

$$\sum_i \left[ \frac{(x_i - x_{oi})^2}{\sigma_{x_i}^2} + \frac{(y_i - y_{oi})^2}{\sigma_{y_i}^2} \right]$$
What about errors in the outputs?
How to calculate them?

If we knew the true distribution, we would know everything that there is to know about the quantitative uncertainties in our experimental measurement $a_{(0)}$. So the name of the game is to find some way of estimating or approximating this probability distribution without knowing $a_{\text{true}}$ and without having available to us an infinite universe of hypothetical data sets.

Figure 15.6.1. A statistical universe of data sets from an underlying model. True parameters $a_{\text{true}}$ are realized in a data set, from which fitted (observed) parameters $a_0$ are obtained. If the experiment were repeated many times, new data sets and new values of the fitted parameters would be obtained.
Let us assume — that the shape of the probability distribution \( a_{(i)} - a_{(0)} \) in the fictitious world is the same, or very nearly the same, as the shape of the probability distribution \( a_{(i)} - a_{\text{true}} \) in the real world.

Figure 13.6.2. Monte Carlo simulation of an experiment. The fitted parameters from an actual experiment are used as surrogates for the true parameters. Computer-generated random numbers are used to simulate many synthetic data sets. Each of these is analyzed to obtain its fitted parameters. The distribution of these fitted parameters around the (known) surrogate true parameters is thus studied.
How to interpret them?

• “Rather than present all details of the probability distribution of errors in parameter estimation, it is common practice to summarize the distribution in the form of confidence limits.

• A confidence region (or confidence interval) is just a region of that \( M \)-dimensional space (hopefully a small region) that contains a certain (hopefully large) percentage of the total probability distribution.

• The experimenter, get to pick both the confidence level (99 percent in the above example), and the shape of the confidence region. The only requirement is that your region does include the stated percentage of probability.”, Numerical Recipes
• “When the method used to estimate the parameters $a_{(0)}$ is chi-square minimization then there is a natural choice for the shape of confidence intervals.
• The region within which $\chi^2$ increases by no more than a set amount $\Delta \chi^2$ defines some $M$-dimensional confidence region around $a(0)$”

Numerical Recipes
The formal covariance matrix that comes out of a $\chi^2$ minimization has a clear quantitative interpretation only if (or to the extent that) the measurement errors actually are normally distributed. In the case of non-normal errors, you are “allowed”

- to fit for parameters by minimizing $\chi^2$
- to use a contour of constant $\Delta \chi^2$ as the boundary of your confidence region
- to use Monte Carlo simulation or detailed analytic calculation in determining which contour $\Delta \chi^2$ is the correct one for your desired confidence level
- to give the covariance matrix $C_{ij}$ as the “formal covariance matrix of the fit.”

You are not allowed

- to use formulas that we now give for the case of normal errors, which establish quantitative relationships among $\Delta \chi^2$, $C_{ij}$, and the confidence level.

$$\delta a_1 = \pm \sqrt{\Delta \chi^2} \sqrt{C_{11}}$$
Program comparisons

- Maple
- Matlab & MFIT
- Root
- Origin
- LSM
- Kaleidagraph
- Excel
Things I didn’t talk about

- Testing the fit
- Singular Value Decomposition for the general linear least squares fitting
- Covariance matrix
- Maximum likelihood method
- The method of Moments
- Is there a package that uses perpendicular offsets and uses errors in all dimensions?
- Fit to non fcts.
Interesting readings and references

- McGill University, *Lab Notes*
- Burden and Faires, *Numerical Analysis*
- Taylor, *An Introduction to Error Analysis The Study of Uncertainties in Physical Measurements*
- Bevington and Robinson, *Data Reduction and Error Analysis for the Physical Sciences*
- Frodesen, Skjeggestad and Tøfte, *Probability and Statistics in Particle Physics*
- *Numerical Recipes in C: The Art of Scientific Computing*