Least-squares method for data/model fitting, motivating it, applying for correlated data and non-linear models

Indrek Mandre <indrek(at)mare.ee>

14 May 2012

Abstract

Where does the method of least squares come from? How does one motivate it (somewhat rigorously)? As an amateur in statistics I couldn't really find many answers even from the wikipedia (as of 11 May 2012), or the numerical recipes in c++ book (that doesn't even mention correlated data/errors), and the Internet is full of hand-waving and scripts for following by rote (for accountants who run our economy, I guess). Here I try to write what I've found out so far. As I'm an amateur, be careful of my derivations and conclusions (this is the disclaimer;). But I hope someone finds this useful.

1 Model and data

Let us have a series of measurements (or Monte-Carlo simulation results) $y_1, \ldots, y_n$, taken at points $x_1, \ldots, x_n$.

We try to fit this data to the model

$$Y = Y\left(x|a_1 \ldots a_m\right), \tag{1}$$

where a_k are the model parameters. That is we try to find the most likely parameters (or we can just say the most likely model) given the data from our measurements.

We assume that the measurements y_i are of normal distribution, and they can be correlated, with correlations given by the covariance matrix

$$\mathbf{\Sigma} = \begin{pmatrix} \operatorname{cov}(y_1, y_1) & \operatorname{cov}(y_1, y_2) & \cdots & \operatorname{cov}(y_1, y_n) \\ \operatorname{cov}(y_2, y_1) & \operatorname{cov}(y_2, y_2) & \cdots & \operatorname{cov}(y_1, y_n) \\ \vdots & \ddots & \ddots & \vdots \\ \operatorname{cov}(y_n, y_1) & \cdots & \cdots & \operatorname{cov}(y_n, y_n) \end{pmatrix}. \tag{2}$$

Note that the diagonals of this matrix are simply the variances $\sigma_i^2 = \operatorname{cov}(y_i, y_i)$ of the measurements.

In real life measurements, the covariances matrix is often unavailable. Then it needs to be estimated. In case of Monte-Carlo simulations, however, it is usually easy to calculate.

In case of no correlations in the measurements (which is usually, and sometimes wrongly assumed), matrix $\mathbf{\Sigma}$ will be diagonal.

2 Optimizing the parameters

For the following, we define these vectors:

$$\mathbf{Y} = \left(Y\left(x_1|a_1 \dots a_m\right), \dots, Y\left(x_n|a_1 \dots a_m\right)\right)^T, \quad (3)$$

$$\mathbf{y} = \left(y_1, \dots, y_n\right)^T. \quad (4)$$

Assuming also multivariate normal distribution for the vector $\mathbf{y}$ (in case of uncorrelated measurements this is automatically given), the probability of the data given the model $P\left(\mathbf{y}|a_1 \dots a_m\right)$ is

$$P\left(\mathbf{y}|a_1 \dots a_m\right) \propto \frac{1}{\left(2\pi\right)^{n/2} \left|\mathbf{\Sigma}\right|^{1/2}} \exp\left(-\frac{1}{2}\left(\mathbf{y}-\mathbf{Y}\right)^T \mathbf{\Sigma}^{-1} \left(\mathbf{y}-\mathbf{Y}\right)\right). \quad (5)$$

What we need is the probability of the model given the data, $P\left(a_1 \dots a_m|\mathbf{y}\right)$, and then we need to maximize this probability by choosing best parameters.

According to the Bayes' theorem,

$$P\left(a_1 \dots a_m|\mathbf{y}\right) = P\left(\mathbf{y}|a_1 \dots a_m\right) \frac{P\left(a_1 \dots a_m\right)}{P\left(\mathbf{y}\right)}. \quad (6)$$

We can't change $P\left(\mathbf{y}\right)$. As for probability of the model $P\left(a_1 \dots a_m\right)$, the easiest route to take here would be to treat all models equal. Therefore,

$$\frac{P\left(a_1 \dots a_m\right)}{P\left(\mathbf{y}\right)} = const, \quad (7)$$

and so

$$P\left(a_1 \dots a_m|\mathbf{y}\right) \propto P\left(\mathbf{y}|a_1 \dots a_m\right). \quad (8)$$

Taking a natural logarithm yields us

$$\ln P\left(a_1 \dots a_m|\mathbf{y}\right) = -\frac{1}{2}\left(\mathbf{y}-\mathbf{Y}\right)^T \mathbf{\Sigma}^{-1} \left(\mathbf{y}-\mathbf{Y}\right) + K, \quad (9)$$

where K is some constant of no importance to us. Hence, to maximize the probability of the model given the data, we need to minimize

$$S = \left(\mathbf{y}-\mathbf{Y}\right)^T \mathbf{\Sigma}^{-1} \left(\mathbf{y}-\mathbf{Y}\right) \quad (10)$$

$$= \sum_{i,j=1}^{n} \left(y_i - Y\left(x_i|a_1 \dots a_m\right)\right) w_{ij} \left(y_j - Y\left(x_j|a_1 \dots a_m\right)\right), \quad (11)$$

where w_{ij} is an element from the matrix $\mathbf{W} = \mathbf{\Sigma}^{-1}$. In case our measurements are not correlated, the covariance matrix $\mathbf{\Sigma}$ is diagonal (with measurement variance σ_i^2 as diagonal elements, that is $w_{ij} = \delta_{ij}\frac{1}{\sigma_i^2}$). This results in the classical

$$S = \sum_{i=1}^{n} \frac{\left(y_i - Y\left(x_i|a_1 \dots a_m\right)\right)^2}{\sigma_i^2}. \quad (12)$$

3 Statistical testing, chi-square distribution

Equation (12) is as per our assumptions a sum of squared normally distributed ($\sim \mathcal{N}(0,1)$) values and so is of chi-square distribution. As different members are related through parameters, the effective number of degrees of freedom is reduced to dof. $= n - m$. Don't ask me what degrees of freedom really are (rigorously). All I have now is a vague feeling.

We will now describe statistical testing. If the sum (12) lies far on the tail of the chi-square distribution, it is very unlikely. In such a case we can conclude that the model is probably wrong. Hence, to accept the model at probability p, we must have

$$S \leq \chi^2_{n-m}(p), \tag{13}$$

where $\chi^2_{\text{dof.}}(p)$ is the quantile at p of the chi-square distribution with $n - m$ degrees of freedom (dof.).

4 Degrees of freedom, the case of correlated data

In case of correlations, the question arises whether (11) is again of chi-square distribution with $n - m$ degrees of freedom. I will try to answer it here. We do this by diagonalizing the covariance matrix and changing the basis for the model and the results.

We assume that the covariance matrix can be diagonalized, that is we can find matrix $\mathbf{P} = (p_{ij})$, such that

$$\mathbf{\Sigma} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}, \tag{14}$$

$$\mathbf{D} = \begin{pmatrix} \hat{\sigma}_1^2 & 0 & \cdots & 0 \\ 0 & \hat{\sigma}_2^2 & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \hat{\sigma}_n^2 \end{pmatrix}. \tag{15}$$

We also note here that as $\mathbf{\Sigma}$ is symmetric, $\mathbf{P}^{-1} = \mathbf{P}^T$. Replacing (14) into (10) yields us

$$S = (\mathbf{y} - \mathbf{Y})^T \mathbf{P}\mathbf{D}^{-1}\mathbf{P}^T (\mathbf{y} - \mathbf{Y}) \tag{16}$$

$$= \left(\mathbf{P}^T\mathbf{y} - \mathbf{P}^T\mathbf{Y}\right)^T \mathbf{D}^{-1} \left(\mathbf{P}^T\mathbf{y} - \mathbf{P}^T\mathbf{Y}\right) \tag{17}$$

$$= \sum_{i=1}^{n} \frac{\left(\hat{y}_i - \hat{Y}(x_i|a_1 \ldots a_m)\right)^2}{\hat{\sigma}_i^2}, \tag{18}$$

where

$$\hat{y}_i = \sum_j p_{ji} y_j, \tag{19}$$

$$\hat{Y}(x_i|a_1 \ldots a_m) = \sum_j p_{ji} Y(x_j|a_1 \ldots a_m). \tag{20}$$

Assuming multivariate normal distribution for $\mathbf{y}$ (any linear combination of its members is then of normal distribution), $\hat{y}_i$ is of normal distribution and so sum

(18) is of chi-square distribution with $n-m$ degrees of freedom. But as that sum is the same as (11), we can conclude that the answer to our question is yes, sum (11) is of chi-square distribution with $n-m$ degrees of freedom.

In a sense we have moved from one model to another, and one should be able to show that $\operatorname{cov}(\hat{y}_i, \hat{y}_j) = 0,\ i \neq j$, that is the transformation has purged the correlations from our measurements.

5 Iteratively towards the minimum

Finding the optimal parameters of a non-linear model is a non-trivial task. Suppose we start with a set of "initial" parameters, and try to improve these gradually/iteratively towards a minimum. Note here that we can end up in a local minimum, not the global one; and that there are no universal/consistent algorithms to search for the global minimum.

Let us define the vector of the m parameters as

$$\mathbf{a} = (a_1, \ldots, a_m)^T . \tag{21}$$

The first idea would be to move along the gradient of S, that is

$$\mathbf{a}_{next} = \mathbf{a}_{cur} - \text{constant} \cdot \nabla S(\mathbf{a}_{cur}) . \tag{22}$$

A naive implementation of this can be very slow and inefficient.

The second idea would be to use the Taylor expansion, function S can then be approximated as

$$S(\mathbf{a}_{next}) = S(\mathbf{a}_{cur}) + (\mathbf{a}_{next} - \mathbf{a}_{cur})^T DS(\mathbf{a}_{cur}) + \frac{1}{2}(\mathbf{a}_{next} - \mathbf{a}_{cur})^T \left\{D^2 S(\mathbf{a}_{cur})\right\}(\mathbf{a}_{next} - \mathbf{a}_{cur}) . \tag{23}$$

This is done on the condition that we are near the minimum where the third derivatives are usually very small and we assume here that they are 0 (a bit fudging here). Differentiating this (taking the gradient), yields us

$$DS(\mathbf{a}_{next}) = DS(\mathbf{a}_{cur}) + D^2 S(\mathbf{a}_{cur})(\mathbf{a}_{next} - \mathbf{a}_{cur}) . \tag{24}$$

In case of an extremum (the minimum), this must be $\mathbf{0}$. Hence, we get a "single-step" equation to the minimum as

$$\mathbf{a}_{next} = \mathbf{a}_{cur} - \left\{D^2 S(\mathbf{a}_{cur})\right\}^{-1} DS(\mathbf{a}_{cur}) . \tag{25}$$

We can rewrite this as

$$\mathbf{a}_{next} = \mathbf{a}_{cur} + \mathbf{H}^{-1}(-\nabla S(\mathbf{a}_{cur})) , \tag{26}$$

where $\mathbf{H} = D^2 S(\mathbf{a}_{cur})$ is the Hessian matrix and $\nabla S = DS$ is the gradient of S. Designating

$$\delta\mathbf{a} \equiv \mathbf{a}_{next} - \mathbf{a}_{cur} \tag{27}$$

as the increment towards the next step, we get two equations to use for finding the minimum:

$$\delta\mathbf{a} = -\text{constant} \cdot \nabla S(\mathbf{a}_{cur}) , \tag{28}$$

$$-\nabla S(\mathbf{a}_{cur}) = \mathbf{H}\,\delta\mathbf{a}. \tag{29}$$

The idea (in primitive form) is to pick one of them, solve for $\delta\mathbf{a}$, and update $\mathbf{a}_{next} = \mathbf{a}_{cur} + \delta\mathbf{a}$. Repeat this until either improvement in the value of S gets really small or we run out of patience (in such a case new ideas are needed).

6 Calculating the gradient and the Hessian

We designate

$$\beta_k \equiv -\frac{1}{2}\frac{\partial S}{\partial a_k}, \qquad \alpha_{kl} \equiv \frac{1}{2}\frac{\partial^2 S}{\partial\alpha_k \partial\alpha_l}. \tag{30}$$

Then we can rewrite (28) and (29) using matrix elements as

$$\sum_{l=1}^{m} \alpha_{kl}\delta a_l = \beta_k, \tag{31}$$

$$\delta a_l = \text{constant} \cdot \beta_l. \tag{32}$$

Let us now calculate the partial derivatives of S, using equation (11):

$$\frac{\partial S}{\partial a_k} = -\sum_{i,j=1}^{n} w_{ij}\left[\frac{\partial Y\left(x_i|\mathbf{a}\right)}{\partial a_k}\left(y_j - Y\left(x_j|\mathbf{a}\right)\right) + \frac{\partial Y\left(x_j|\mathbf{a}\right)}{\partial a_k}\left(y_i - Y\left(x_i|\mathbf{a}\right)\right)\right]. \tag{33}$$

The second partial derivatives are

$$\frac{\partial^2 S}{\partial a_k \partial a_l} = -\sum_{i,j=1}^{n} w_{ij}\left[\frac{\partial^2 Y\left(x_i|\mathbf{a}\right)}{\partial a_k \partial a_l}\left(y_j - Y\left(x_j|\mathbf{a}\right)\right) - \frac{\partial Y\left(x_i|\mathbf{a}\right)}{\partial a_k}\frac{\partial Y\left(x_j|\mathbf{a}\right)}{\partial a_l} + \frac{\partial^2 Y\left(x_j|\mathbf{a}\right)}{\partial a_k \partial a_l}\left(y_i - Y\left(x_i|\mathbf{a}\right)\right) - \frac{\partial Y\left(x_j|\mathbf{a}\right)}{\partial a_k}\frac{\partial Y\left(x_i|\mathbf{a}\right)}{\partial a_l}\right]. \tag{34}$$

We are going to fudge again here by claiming that near the minimum the elements containing second derivatives of Y are very small and their statistical mean is 0 (they cancel out). Hence we simplify the second partial derivative of S to

$$\frac{\partial^2 S}{\partial a_k \partial a_l} = \sum_{i,j=1}^{n} w_{ij}\left[\frac{\partial Y\left(x_i|\mathbf{a}\right)}{\partial a_k}\frac{\partial Y\left(x_j|\mathbf{a}\right)}{\partial a_l} + \frac{\partial Y\left(x_j|\mathbf{a}\right)}{\partial a_k}\frac{\partial Y\left(x_i|\mathbf{a}\right)}{\partial a_l}\right]. \tag{35}$$

But don't worry too much. This fiddling will only affect the path we take to the minimum (maybe making it slightly longer), but will not affect the end result.

7 Levenberg–Marquardt algorithm

Using the equations given for the derivatives of S, one can now apply the Levenberg-Marquardt algorithm, as described in "Numerical Recipes: The Art of Scientific Computing". I've been using the third edition. Wonderful book.

The outline is following: We define a matrix $\alpha' = (\alpha'_{kl})$ by transforming $\alpha_{kl} \longrightarrow \alpha'_{kl}$, so that

$$\alpha'_{jj} \equiv \alpha_{jj}\left(1+\lambda\right), \tag{36}$$

$$\alpha'_{jk} \equiv \alpha_{jk}, \quad (j \neq k), \tag{37}$$

where λ is the "weight parameter", and start solving the system of m equations

$$\sum_{l=1}^{m} \alpha'_{kl} \delta a_l = \beta_k, \quad (k = 1 \dots m) . \tag{38}$$

The algorithm is as following:

- Make a guess of initial parameters and place them into $\mathbf{a}$;
- Compute $S(\mathbf{a})$;
- Pick a modest value for λ, such as $\lambda = 0.001$;
- (*) Solve the system of linear equations (38) for $\delta\mathbf{a}$ and evaluate $S(\mathbf{a} + \delta\mathbf{a})$;
- If $S(\mathbf{a} + \delta\mathbf{a}) \geq S(\mathbf{a})$, increase λ by a factor of 10, and go back to (*);
- If $S(\mathbf{a} + \delta\mathbf{a}) < S(\mathbf{a})$, decrease λ by a factor of 10, update $\mathbf{a} \leftarrow \mathbf{a} + \delta\mathbf{a}$ and go back to (*).

Somewhere along the way you have to stop of course, say when the change in S has been marginal for several consequent iterations.

8 Uncertainty of the parameters

While we have found the minimum of S, that is the best parameters, and say they have passed the chi-square test, there is still something unclear. How accurate are the parameters we have found? Quantitatively, what are their uncertainties?

It can be shown (with some fudging for the non-linear models I think), that the parameter covariance matrix $\mathbf{C}$ can be found from

$$\mathbf{C} = \alpha^{-1},$$

where $\alpha = (\alpha_{kl})$ is the matrix formed from elements described in (30). Parameter standard deviation is then given by

$$\sigma_{a_k} = \sqrt{c_{kk}}.$$

All of this is of course on the condition of multivariate normal distribution for the measurements vector $\mathbf{y}$.