

## Linear least squares

A system of linear equations is considered *overdetermined* if there are more equations than unknown variables. If all equations of an overdetermined system are linearly independent, the system has no exact solution.

A *linear least-squares problem* is the problem of finding an approximate solution to an overdetermined system. It often arises in applications where a theoretical model is fitted to experimental data.

### Linear least-squares problem

Consider a linear system

$$A\mathbf{c} = \mathbf{b}, \quad (1)$$

where  $A$  is an  $n \times m$  matrix,  $\mathbf{c}$  is an  $m$ -component vector of unknown variables and  $\mathbf{b}$  is an  $n$ -component vector of the right-hand side terms. If the number of equations  $n$  is larger than the number of unknowns  $m$ , the system is overdetermined and generally has no solution.

However, it is still possible to find an approximate solution – the one where  $A\mathbf{c}$  is only approximately equal  $\mathbf{b}$ , in the sense that the Euclidean norm of the difference between  $A\mathbf{c}$  and  $\mathbf{b}$  is minimized,

$$\min_{\mathbf{c}} \|A\mathbf{c} - \mathbf{b}\|^2. \quad (2)$$

The problem (2) is called the linear least-squares problem and the vector  $\mathbf{c}$  that minimizes  $\|A\mathbf{c} - \mathbf{b}\|^2$  is called the *least-squares solution*.

### Solution via QR-decomposition

The linear least-squares problem can be solved by QR-decomposition. The matrix  $A$  is factorized as  $A = QR$ , where  $Q$  is  $n \times m$  matrix with orthogonal columns,  $Q^T Q = 1$ , and  $R$  is an  $m \times m$  upper triangular matrix. The Euclidean norm  $\|A\mathbf{c} - \mathbf{b}\|^2$  can then be rewritten as

$$\|A\mathbf{c} - \mathbf{b}\|^2 = \|QR\mathbf{c} - \mathbf{b}\|^2 = \|R\mathbf{c} - Q^T\mathbf{b}\|^2 + \|(1 - QQ^T)\mathbf{b}\|^2 \geq \|(1 - QQ^T)\mathbf{b}\|^2. \quad (3)$$

The term  $\|(1 - QQ^T)\mathbf{b}\|^2$  is independent of the variables  $\mathbf{c}$  and can not be reduced by their variations. However, the term  $\|R\mathbf{c} - Q^T\mathbf{b}\|^2$  can be reduced down to zero by solving the  $m \times m$  system of linear equations

$$R\mathbf{c} - Q^T\mathbf{b} = 0. \quad (4)$$

The system is right-triangular and can be readily solved by back-substitution.

Thus the solution to the linear least-squares problem (2) is given by the solution of the triangular system (4).

### Ordinary least-squares curve fitting

Ordinary (or linear) least-squares curve fitting is a problem of fitting  $n$  (experimental) data points  $\{x_i, y_i \pm \Delta y_i\}$ , where  $\Delta y_i$  are experimental errors, by a linear combination of  $m$  functions

$$F(x) = \sum_{k=1}^m c_k f_k(x). \quad (5)$$

The objective of the least-squares fit is to minimize the square deviation, called  $\chi^2$ , between the fitting function and the experimental data,

$$\chi^2 = \sum_{i=1}^n \left( \frac{F(x_i) - y_i}{\Delta y_i} \right)^2. \quad (6)$$

Individual deviations from experimental points are weighted with their inverse errors in order to promote contributions from the more precise measurements.

Minimization of  $\chi^2$  with respect to the coefficient  $c_k$  in (5) is apparently equivalent to the least-squares problem (2) where

$$A_{ik} = \frac{f_k(x_i)}{\Delta y_i}, \quad b_i = \frac{y_i}{\Delta y_i}. \quad (7)$$

If  $QR = A$  is the QR-decomposition of the matrix  $A$ , the formal least-squares solution is

$$\mathbf{c} = R^{-1}Q^T\mathbf{b}. \quad (8)$$

However in practice it is better to back-substitute the system  $R\mathbf{c} = Q^T\mathbf{b}$ .

### Variations and correlations of fitting parameters

Suppose  $\delta y_i$  is a (small) deviation of the measured value of the physical observable from its exact value. The corresponding deviation  $\delta c_k$  of the fitting coefficient is then given as

$$\delta c_k = \sum_i \frac{\partial c_k}{\partial y_i} \delta y_i. \quad (9)$$

In a good experiment the deviations  $\delta y_i$  are statistically independent and distributed normally with the standard deviations  $\Delta y_i$ . The deviations (9) are then also distributed normally with *variances*,

$$\langle \delta c_k \delta c_k \rangle = \sum_i \left( \frac{\partial c_k}{\partial y_i} \Delta y_i \right)^2 = \sum_i \left( \frac{\partial c_k}{\partial b_i} \right)^2. \quad (10)$$

The standard errors in the fitting coefficients are then given as the square roots of variances,

$$\Delta c_k = \sqrt{\langle \delta c_k \delta c_k \rangle} = \sqrt{\sum_i \left( \frac{\partial c_k}{\partial b_i} \right)^2}. \quad (11)$$

The variances are diagonal elements of the *covariance matrix*,  $\Sigma$ , made of *covariances*,

$$\Sigma_{kq} \equiv \langle \delta c_k \delta c_q \rangle = \sum_i \frac{\partial c_k}{\partial b_i} \frac{\partial c_q}{\partial b_i}. \quad (12)$$

Covariances  $\langle \delta c_k \delta c_q \rangle$  are measures of to what extent the coefficients  $c_k$  and  $c_q$  change together if the measured values  $y_i$  are varied. The normalized covariances,

$$\frac{\langle \delta c_k \delta c_q \rangle}{\sqrt{\langle \delta c_k \delta c_k \rangle \langle \delta c_q \delta c_q \rangle}} \quad (13)$$

are called *correlations*.

Using (12) and (8) the covariance matrix can be calculated as

$$\Sigma = \left( \frac{\partial \mathbf{c}}{\partial \mathbf{b}} \right) \left( \frac{\partial \mathbf{c}}{\partial \mathbf{b}} \right)^T = R^{-1} (R^{-1})^T = (R^T R)^{-1} = (A^T A)^{-1}. \quad (14)$$

The square roots of the diagonal elements of this matrix provide the estimates of the errors of the fitting coefficients and the (normalized) off-diagonal elements are the estimates of their correlations.

### JavaScript implementation

```
function lsfit(xs,ys,dys,funs){ // Linear least squares fit
// uses: qrdec, qrback, inverse
// input: data points {x,y,dy}; functions {funs}
// output: fitting coefficients c and covariance matrix S
var dot = function(a,b) // a.b
  {let s=0;for(let i in a)s+=a[i]*b[i];return s}
var ttimes = function(A,B) // A^T*B
```

```
    [[dot(A[r],B[c]) for(r in A)] for(c in B)];  
var A=[[funs[k](xs[i])/dys[i] for(i in xs)] for(k in funs)];  
var b=[ys[i]/dys[i] for(i in ys)];  
var [Q,R]=qrdec(A);  
var c=qrback(Q,R,b);  
var S=inverse(ttimes(R,R));  
return [c,S];  
}
```