WIKIPEDIA

Non-linear least squares

Non-linear least squares is the form of least squares analysis used to fit a set of m observations with a model that is non-linear in n unknown parameters ($m \geq n$). It is used in some forms of nonlinear regression. The basis of the method is to approximate the model by a linear one and to refine the parameters by successive iterations. There are many similarities to linear least squares, but also some significant differences. In economic theory, the non-linear least squares method is applied in (i) the probit regression, (ii) threshold regression, (iii) smooth regression, (iv) logistic link regression, (v) Box-Cox transformed regressors ($m(x,\theta_i) = \theta_1 + \theta_2 x^{(\theta_3)}$).

Contents

Theory

Geometrical interpretation

Computation

- Initial parameter estimates
- Solution
- Convergence criteria
- Calculation of the Jacobian by numerical approximation
- Parameter errors, confidence limits, residuals etc.
- Multiple minima
- Transformation to a linear model

Algorithms

- Gauss–Newton method
  - Shift-cutting
  - Marquardt parameter
- QR decomposition
- Singular value decomposition
- Gradient methods
- Direct search methods

See also

References

Further reading

Theory

Consider a set of m data points, $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, and a curve (model function) $y = f(x, \boldsymbol{\beta})$, that in addition to the variable x also depends on n parameters, $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_n)$, with $m \geq n$. It is desired to find the vector $\boldsymbol{\beta}$ of parameters such that the curve fits best the given data in the least squares sense, that is, the sum of squares

$$S = \sum_{i=1}^{m} r_i^2$$

is minimized, where the residuals (in-sample prediction errors) r_i are given by

$$r_i = y_i - f(x_i, \boldsymbol{\beta})$$

for $i = 1, 2, \ldots, m$.

The minimum value of S occurs when the gradient is zero. Since the model contains n parameters there are n gradient equations:

$$\frac{\partial S}{\partial \beta_j} = 2\sum_i r_i \frac{\partial r_i}{\partial \beta_j} = 0 \quad (j = 1, \ldots, n).$$

In a nonlinear system, the derivatives $\frac{\partial r_i}{\partial \beta_j}$ are functions of both the independent variable and the parameters, so in general these gradient equations do not have a closed solution. Instead, initial values must be chosen for the parameters. Then, the parameters are refined iteratively, that is, the values are obtained by successive approximation,

$$\beta_j \approx \beta_j^{k+1} = \beta_j^k + \Delta\beta_j.$$

Here, *k* is an iteration number and the vector of increments, $\Delta\boldsymbol{\beta}$ is known as the shift vector. At each iteration the model is linearized by approximation to a first-order Taylor polynomial expansion about $\boldsymbol{\beta}^k$

$$f(x_i, \boldsymbol{\beta}) \approx f(x_i, \boldsymbol{\beta}^k) + \sum_j \frac{\partial f(x_i, \boldsymbol{\beta}^k)}{\partial \beta_j} \left(\beta_j - \beta_j^k\right) = f(x_i, \boldsymbol{\beta}^k) + \sum_j J_{ij}\, \Delta\beta_j.$$

The Jacobian, **J**, is a function of constants, the independent variable *and* the parameters, so it changes from one iteration to the next. Thus, in terms of the linearized model, $\frac{\partial r_i}{\partial \beta_j} = -J_{ij}$ and the residuals are given by

$$\Delta y_i = y_i - f(x_i, \boldsymbol{\beta}^k).$$
$$r_i = y_i - f(x_i, \boldsymbol{\beta}) = \left(y_i - f(x_i, \boldsymbol{\beta}^k)\right) + \left(f(x_i, \boldsymbol{\beta}^k) - f(x_i, \boldsymbol{\beta})\right) \approx \Delta y_i - \sum_{s=1}^{n} J_{is} \Delta\beta_s.$$

Substituting these expressions into the gradient equations, they become

$$-2\sum_{i=1}^{m} J_{ij} \left(\Delta y_i - \sum_{s=1}^{n} J_{is}\, \Delta\beta_s\right) = 0,$$

which, on rearrangement, become *n* simultaneous linear equations, the **normal equations**

$$\sum_{i=1}^{m}\sum_{s=1}^{n} J_{ij} J_{is}\, \Delta\beta_s = \sum_{i=1}^{m} J_{ij}\, \Delta y_i \qquad (j = 1, \ldots, n).$$

The normal equations are written in matrix notation as

$$\left(\mathbf{J^T J}\right) \Delta\boldsymbol{\beta} = \mathbf{J^T}\, \Delta\mathbf{y}.$$

When the observations are not equally reliable, a weighted sum of squares may be minimized,

$$S = \sum_{i=1}^{m} W_{ii} r_i^2.$$

Each element of the diagonal weight matrix **W** should, ideally, be equal to the reciprocal of the error variance of the measurement.[1] The normal equations are then

$$\left(\mathbf{J^T W J}\right) \Delta\boldsymbol{\beta} = \mathbf{J^T W}\, \Delta\mathbf{y}.$$

These equations form the basis for the Gauss–Newton algorithm for a non-linear least squares problem.

Geometrical interpretation

In linear least squares the objective function, S, is a quadratic function of the parameters.

$$S = \sum_i W_{ii} \left(y_i - \sum_j X_{ij} \beta_j \right)^2$$

When there is only one parameter the graph of S with respect to that parameter will be a parabola. With two or more parameters the contours of S with respect to any pair of parameters will be concentric ellipses (assuming that the normal equations matrix $\mathbf{X^T W X}$ is positive definite). The minimum parameter values are to be found at the centre of the ellipses. The geometry of the general objective function can be described as paraboloid elliptical. In NLLSQ the objective function is quadratic with respect to the parameters only in a region close to its minimum value, where the truncated Taylor series is a good approximation to the model.

$$S \approx \sum_i W_{ii} \left(y_i - \sum_j J_{ij} \beta_j \right)^2$$

The more the parameter values differ from their optimal values, the more the contours deviate from elliptical shape. A consequence of this is that initial parameter estimates should be as close as practicable to their (unknown!) optimal values. It also explains how divergence can come about as the Gauss–Newton algorithm is convergent only when the objective function is approximately quadratic in the parameters.

Computation

Initial parameter estimates

Some problems of ill-conditioning and divergence can be corrected by finding initial parameter estimates that are near to the optimal values. A good way to do this is by computer simulation. Both the observed and calculated data are displayed on a screen. The parameters of the model are adjusted by hand until the agreement between observed and calculated data is reasonably good. Although this will be a subjective judgment, it is sufficient to find a good starting point for the non-linear refinement. Initial parameter estimates can be created using transformations or linearizations. Better still evolutionary algorithms such as the Stochastic Funnel Algorithm can lead to the convex basin of attraction that surrounds the optimal parameter estimates. Hybrid algorithms that use randomization and elitism, followed by Newton methods have been shown to be useful and computationally efficient.

Solution

Any method among the ones described below can be applied to find a solution.

Convergence criteria

The common sense criterion for convergence is that the sum of squares does not decrease from one iteration to the next. However this criterion is often difficult to implement in practice, for various reasons. A useful convergence criterion is

$$\left| \frac{S^k - S^{k+1}}{S^k} \right| < 0.0001.$$

The value 0.0001 is somewhat arbitrary and may need to be changed. In particular it may need to be increased when experimental errors are large. An alternative criterion is

$$\left| \frac{\Delta \beta_j}{\beta_j} \right| < 0.001, \qquad j = 1, \ldots, n.$$

Again, the numerical value is somewhat arbitrary; 0.001 is equivalent to specifying that each parameter should be refined to 0.1% precision. This is reasonable when it is less than the largest relative standard deviation on the parameters.

Calculation of the Jacobian by numerical approximation

There are models for which it is either very difficult or even impossible to derive analytical expressions for the elements of the Jacobian. Then, the numerical approximation

$$\frac{\partial f(x_i, \boldsymbol{\beta})}{\partial \beta_j} \approx \frac{\delta f(x_i, \boldsymbol{\beta})}{\delta \beta_j}$$

is obtained by calculation of $f(x_i, \boldsymbol{\beta})$ for β_j and $\beta_j + \delta\beta_j$. The increment, $\delta\beta_j$, size should be chosen so the numerical derivative is not subject to approximation error by being too large, or round-off error by being too small.

Parameter errors, confidence limits, residuals etc.

Some information is given in the corresponding section on the linear least squares page.

Multiple minima

Multiple minima can occur in a variety of circumstances some of which are:

- A parameter is raised to a power of two or more. For example, when fitting data to a Lorentzian curve

$$f(x_i, \boldsymbol{\beta}) = \frac{\alpha}{1 + \left(\frac{\gamma - x_i}{\beta}\right)^2}$$

where α is the height, γ is the position and β is the half-width at half height, there are two solutions for the half-width, $\hat{\beta}$ and $-\hat{\beta}$ which give the same optimal value for the objective function.

- Two parameters can be interchanged without changing the value of the model. A simple example is when the model contains the product of two parameters, since $\alpha\beta$ will give the same value as $\beta\alpha$.
- A parameter is in a trigonometric function, such as $\sin\beta$, which has identical values at $\hat{\beta} + 2n\pi$. See Levenberg–Marquardt algorithm for an example.

Not all multiple minima have equal values of the objective function. False minima, also known as local minima, occur when the objective function value is greater than its value at the so-called global minimum. To be certain that the minimum found is the global minimum, the refinement should be started with widely differing initial values of the parameters. When the same minimum is found regardless of starting point, it is likely to be the global minimum.

When multiple minima exist there is an important consequence: the objective function will have a maximum value somewhere between two minima. The normal equations matrix is not positive definite at a maximum in the objective function, as the gradient is zero and no unique direction of descent exists. Refinement from a point (a set of parameter values) close to a maximum will be ill-conditioned and should be avoided as a starting point. For example, when fitting a Lorentzian the normal equations matrix is not positive definite when the half-width of the band is zero.[2]

Transformation to a linear model

A non-linear model can sometimes be transformed into a linear one. For example, when the model is a simple exponential function,

$$f(x_i, \boldsymbol{\beta}) = \alpha e^{\beta x_i}$$

it can be transformed into a linear model by taking logarithms.

$$\log f(x_i, \boldsymbol{\beta}) = \log \alpha + \beta x_i$$

Graphically this corresponds to working on a semi-log plot. The sum of squares becomes

$$S = \sum_i (\log y_i - \log \alpha - \beta x_i)^2.$$

This procedure should be avoided unless the errors are multiplicative and log-normally distributed because it can give misleading results. This comes from the fact that whatever the experimental errors on **y** might be, the errors on **log y** are different. Therefore, when the transformed sum of squares is minimized different results will be obtained both for the parameter values and their calculated standard deviations. However, with multiplicative errors that are log-normally distributed, this procedure gives unbiased and consistent parameter estimates.

Another example is furnished by Michaelis–Menten kinetics, used to determine two parameters V_{max} and K_m:

$$v = \frac{V_{\mathrm{max}}[S]}{K_m + [S]}.$$

The Lineweaver–Burk plot

$$\frac{1}{v} = \frac{1}{V_{\mathrm{max}}} + \frac{K_m}{V_{\mathrm{max}}[S]}$$

of $\frac{1}{v}$ against $\frac{1}{[S]}$ is linear in the parameters $\frac{1}{V_{\mathrm{max}}}$ and $\frac{K_m}{V_{\mathrm{max}}}$, but very sensitive to data error and strongly biased toward fitting the data in a particular range of the independent variable $[S]$.

Algorithms

Gauss–Newton method

The normal equations

$$\left(\mathbf{J^T W J}\right) \boldsymbol{\Delta\beta} = \left(\mathbf{J^T W}\right) \boldsymbol{\Delta y}$$

may be solved for $\boldsymbol{\Delta\beta}$ by Cholesky decomposition, as described in linear least squares. The parameters are updated iteratively

$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k + \boldsymbol{\Delta\beta}$$

where k is an iteration number. While this method may be adequate for simple models, it will fail if divergence occurs. Therefore, protection against divergence is essential.

Shift-cutting

If divergence occurs, a simple expedient is to reduce the length of the shift vector, $\boldsymbol{\Delta\beta}$, by a fraction, f

$$\boldsymbol{\beta}^{k+1} = \boldsymbol{\beta}^k + f\,\boldsymbol{\Delta\beta}.$$

For example, the length of the shift vector may be successively halved until the new value of the objective function is less than its value at the last iteration. The fraction, f could be optimized by a line search.[3] As each trial value of f requires the objective function to be re-calculated it is not worth optimizing its value too stringently.

When using shift-cutting, the direction of the shift vector remains unchanged. This limits the applicability of the method to situations where the direction of the shift vector is not very different from what it would be if the objective function were approximately quadratic in the parameters, $\boldsymbol{\beta}^k$.

Marquardt parameter

If divergence occurs and the direction of the shift vector is so far from its "ideal" direction that shift-cutting is not very effective, that is, the fraction, f required to avoid divergence is very small, the direction must be changed. This can be achieved by using the Marquardt parameter.[4] In this method the normal equations are modified

$$\left(\mathbf{J^T W J} + \lambda \mathbf{I}\right) \boldsymbol{\Delta\beta} = \left(\mathbf{J^T W}\right) \boldsymbol{\Delta y}$$

where λ is the Marquardt parameter and $\mathbf{I}$ is an identity matrix. Increasing the value of λ has the effect of changing both the direction and the length of the shift vector. The shift vector is rotated towards the direction of steepest descent

$$\text{when } \lambda\mathbf{I} \gg \mathbf{J^T W J},\ \boldsymbol{\Delta\beta} \approx (1/\lambda)\mathbf{J^T W}\ \boldsymbol{\Delta \mathbf{y}}.$$

$\mathbf{J^T W}\ \boldsymbol{\Delta \mathbf{y}}$ is the steepest descent vector. So, when λ becomes very large, the shift vector becomes a small fraction of the steepest descent vector.

Various strategies have been proposed for the determination of the Marquardt parameter. As with shift-cutting, it is wasteful to optimize this parameter too stringently. Rather, once a value has been found that brings about a reduction in the value of the objective function, that value of the parameter is carried to the next iteration, reduced if possible, or increased if need be. When reducing the value of the Marquardt parameter, there is a cut-off value below which it is safe to set it to zero, that is, to continue with the unmodified Gauss–Newton method. The cut-off value may be set equal to the smallest singular value of the Jacobian.[5] A bound for this value is given by $1/\mathbf{trace}\left(\mathbf{J^T W J}\right)^{-1}$.[6]

QR decomposition

The minimum in the sum of squares can be found by a method that does not involve forming the normal equations. The residuals with the linearized model can be written as

$$\mathbf{r} = \boldsymbol{\Delta \mathbf{y}} - \mathbf{J}\ \boldsymbol{\Delta\beta}.$$

The Jacobian is subjected to an orthogonal decomposition; the QR decomposition will serve to illustrate the process.

$$\mathbf{J} = \mathbf{QR}$$

where $\mathbf{Q}$ is an orthogonal $m \times m$ matrix and $\mathbf{R}$ is an $m \times n$ matrix which is partitioned into an $n \times n$ block, $\mathbf{R}_n$, and a $(m-n) \times n$ zero block. $\mathbf{R}_n$ is upper triangular.

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_n \\ \mathbf{0} \end{bmatrix}$$

The residual vector is left-multiplied by $\mathbf{Q}^T$.

$$\mathbf{Q^T r} = \mathbf{Q^T}\ \boldsymbol{\Delta \mathbf{y}} - \mathbf{R}\ \boldsymbol{\Delta\beta} = \begin{bmatrix} \left(\mathbf{Q^T}\ \boldsymbol{\Delta \mathbf{y}} - \mathbf{R}\ \boldsymbol{\Delta\beta}\right)_n \\ \left(\mathbf{Q^T}\ \boldsymbol{\Delta \mathbf{y}}\right)_{m-n} \end{bmatrix}$$

This has no effect on the sum of squares since $S = \mathbf{r^T Q Q^T r} = \mathbf{r^T r}$ because $\mathbf{Q}$ is orthogonal The minimum value of S is attained when the upper block is zero. Therefore, the shift vector is found by solving

$$\mathbf{R_n}\ \boldsymbol{\Delta\beta} = \left(\mathbf{Q^T}\ \boldsymbol{\Delta \mathbf{y}}\right)_\mathbf{n}.$$

These equations are easily solved as $\mathbf{R}$ is upper triangular.

Singular value decomposition

A variant of the method of orthogonal decomposition involves singular value decomposition, in which $\mathbf{R}$ is diagonalized by further orthogonal transformations.

$$\mathbf{J} = \mathbf{U\Sigma V^T}$$

where $\mathbf{U}$ is orthogonal, $\boldsymbol{\Sigma}$ is a diagonal matrix of singular values and $\mathbf{V}$ is the orthogonal matrix of the eigenvectors of $\mathbf{J^T J}$ or equivalently the right singular vectors of $\mathbf{J}$. In this case the shift vector is given by

$$\boldsymbol{\Delta\beta} = \mathbf{V\Sigma^{-1}}\left(\mathbf{U^T}\ \boldsymbol{\Delta \mathbf{y}}\right)_n.$$

The relative simplicity of this expression is very useful in theoretical analysis of non-linear least squares. The application of singular value decomposition is discussed in detail in Lawson and Hanson.[5]

Gradient methods

There are many examples in the scientific literature where different methods have been used for non-linear data-fitting problems.

- Inclusion of second derivatives in The Taylor series expansion of the model function. This is Newton's method in optimization.

$$f(x_i, \boldsymbol{\beta}) = f^k(x_i, \boldsymbol{\beta}) + \sum_j J_{ij}\, \Delta\beta_j + \frac{1}{2} \sum_j \sum_k \Delta\beta_j\, \Delta\beta_k\, H_{jk_{(i)}}\,,\ H_{jk_{(i)}} = \frac{\partial^2 f(x_i, \boldsymbol{\beta})}{\partial\beta_j\, \partial\beta_k}.$$

  The matrix **H** is known as the Hessian matrix. Although this model has better convergence properties near to the minimum, it is much worse when the parameters are far from their optimal values. Calculation of the Hessian adds to the complexity of the algorithm. This method is not in general use.

- Davidon–Fletcher–Powell method. This method, a form of pseudo-Newton method, is similar to the one above but calculates the Hessian by successive approximation, to avoid having to use analytical expressions for the second derivatives.
- Steepest descent. Although a reduction in the sum of squares is guaranteed when the shift vector points in the direction of steepest descent, this method often performs poorly. When the parameter values are far from optimal the direction of the steepest descent vector, which is normal (perpendicular) to the contours of the objective function, is very different from the direction of the Gauss–Newton vector. This makes divergence much more likely, especially as the minimum along the direction of steepest descent may correspond to a small fraction of the length of the steepest descent vector. When the contours of the objective function are very eccentric, due to there being high correlation between parameters, the steepest descent iterations, with shift-cutting, follow a slow, zig-zag trajectory towards the minimum.
- Conjugate gradient search. This is an improved steepest descent based method with good theoretical convergence properties, although it can fail on finite-precision digital computers even when used on quadratic problems.[7]

Direct search methods

Direct search methods depend on evaluations of the objective function at a variety of parameter values and do not use derivatives at all. They offer alternatives to the use of numerical derivatives in the Gauss–Newton method and gradient methods.

- Alternating variable search.[3] Each parameter is varied in turn by adding a fixed or variable increment to it and retaining the value that brings about a reduction in the sum of squares. The method is simple and effective when the parameters are not highly correlated. It has very poor convergence properties, but may be useful for finding initial parameter estimates.
- Nelder–Mead (simplex) search. A simplex in this context is a polytope of $n + 1$ vertices in n dimensions; a triangle on a plane, a tetrahedron in three-dimensional space and so forth. Each vertex corresponds to a value of the objective function for a particular set of parameters. The shape and size of the simplex is adjusted by varying the parameters in such a way that the value of the objective function at the highest vertex always decreases. Although the sum of squares may initially decrease rapidly, it can converge to a nonstationary point on quasiconvex problems, by an example of M. J. D. Powell.

More detailed descriptions of these, and other, methods are available, in *Numerical Recipes*, together with computer code in various languages.

See also

- Least squares support vector machine
- Curve fitting
- Grey box model
- Nonlinear programming
- Nonlinear regression
- Optimization (mathematics)
- Levenberg–Marquardt algorithm