Notes on the Non-linear Regression

The model

Non-linear regression models, like ordinary linear models, assume that the response, Y, has a normal distribution with constant variance. Unlike linear models, however, the mean function may depend non-linearly on the unknown parameters. The model my therefore be written generically as

$$Y = g(\boldsymbol{\gamma}; \mathbf{x}) + \varepsilon, \qquad \text{where} \qquad \varepsilon \sim \mathrm{N}\left(0, \sigma^2\right) \tag{1}$$

The function $g(,;)$ is known and usually comes from some well-established theory. It is often the case that the fixed part of the model comes from a deterministic process and the random part from homogeneous measurement errors.

Linear and non-linear parameters

It is sometimes useful to identify any *linear parameters* in the model and to distinguish them from the *non-linear parameters*. To do this, suppose the model can be written in the form

$$Y = \beta_1 g_1(\boldsymbol{\tau}; \mathbf{x}) + \beta_2 g_2(\boldsymbol{\tau}; \mathbf{x}) + \cdots + \beta_p g_p(\boldsymbol{\tau}; \mathbf{x}) + \varepsilon \tag{2}$$

In this case the parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ are called *linear parameters*, and the components of $\boldsymbol{\tau}$ are called the *non-linear parameters*. More formally, a parameter, β_j, is called a *linear parameter* if the partial derivative of the model function with respect to β_j does not depend on β_j. In other words, if and only if

$$\frac{\partial^2 g(\boldsymbol{\beta}; \mathbf{x})}{\partial \beta_j^2} \equiv 0$$

The complete parameter vector is $\boldsymbol{\gamma} = (\boldsymbol{\beta}, \boldsymbol{\tau})$.

In matrix terms the model may be written, isolating linear from non-linear parameters, as

$$\mathbf{Y} = \mathbf{G}(\boldsymbol{\tau})\boldsymbol{\beta} + \boldsymbol{\varepsilon} \quad \text{where the matrix} \quad \mathbf{G}(\boldsymbol{\tau})^{n \times p} = \left[g_j(\boldsymbol{\tau}; \mathbf{x}_i)\right]$$

Estimation of parameters

Since the model has normal independent responses with equal variance, maximum likelihood estimation is equivalent to least squares. Hence we estimate $\boldsymbol{\gamma}$ by minimising

$$\mathrm{RSS}(\boldsymbol{\gamma}) = \sum_{i=1}^{n} \left(y_i - g(\boldsymbol{\gamma}; \mathbf{x}_i)\right)^2 = \left\| \mathbf{y} - \mathbf{g}(\boldsymbol{\gamma}) \right\|^2$$

Solution locus and tangent space

It is useful to have a geometric understanding of this problem. Let Γ be the set of all possible values of the unknown parameter vector, $\boldsymbol{\gamma}^{p\times 1}$. The $p-$dimensional surface $\boldsymbol{S}$ in $n-$dimensional sample space defined by

$$\boldsymbol{S} = \{\mathbf{g}(\boldsymbol{\gamma}) \mid \boldsymbol{\gamma} \in \Gamma\} \subset \mathbb{R}^n$$

is sometimes called the "solution locus". It is the known set in which the true mean vector lies. The points within $\boldsymbol{S}$ are uniquely indexed by the parameter vector $\boldsymbol{\gamma}$, which therefore defines a coordinate system within $\boldsymbol{S}$. The response vector, $\mathbf{y}^{n\times 1}$, is also in $\mathbb{R}^n$, and the estimation problem amounts to finding the vector in $\boldsymbol{S}$ closest to the observation vector, $\mathbf{y}$ in the sense of Euclidean distance. The vector itself is the estimated mean and the coordinates of the vector are the least squares estimates of the parameters.

Let $\boldsymbol{\gamma}^{(0)}$ be an initial approximate to the least squares estimate vector, $\hat{\boldsymbol{\gamma}}$, and expand the model function in a first order Taylor series about $\boldsymbol{\gamma}^{(0)}$:

$$g(\boldsymbol{\gamma};\mathbf{x}_i) = g(\boldsymbol{\gamma}^{(0)};\mathbf{x}_i) + \left.\frac{\partial g}{\partial \gamma_1}\right|_{\boldsymbol{\gamma}=\boldsymbol{\gamma}^{(0)}} (\gamma_1 - \gamma_1^{(0)}) + \cdots + \left.\frac{\partial g}{\partial \gamma_p}\right|_{\boldsymbol{\gamma}=\boldsymbol{\gamma}^{(0)}} (\gamma_p - \gamma_p^{(0)}) + \text{"smaller order terms"}$$

The vectors

$$\mathbf{g}_1^{(0)} = \left.\frac{\partial \mathbf{g}}{\partial \gamma_1}\right|_{\boldsymbol{\gamma}=\boldsymbol{\gamma}^{(0)}}, \ldots, \mathbf{g}_p^{(0)} = \left.\frac{\partial \mathbf{g}}{\partial \gamma_p}\right|_{\boldsymbol{\gamma}=\boldsymbol{\gamma}^{(0)}}$$

define the tangent space to the solution locus, $\boldsymbol{S}$, at the point defined by $\boldsymbol{\gamma}^{(0)}$.

Iterative approximation

An iterative method for finding the least squares estimates, equivalent to the usual Newton-Raphson scheme, is as follows:

1. Begin with an initial approximation, $\boldsymbol{\gamma}^{(0)}$, obtained by any convenient method.
2. Calculate the tangent vectors to $\boldsymbol{S}$ at the point defined by $\boldsymbol{\gamma}^{(0)}$, that is, $\mathbf{g}_1^{(0)}, \ldots, \mathbf{g}_p^{(0)}$.
3. Regress $\mathbf{y} - \mathbf{g}(\boldsymbol{\gamma}^{(0)})$ on $\mathbf{G} = \left[\mathbf{g}_1^{(0)}, \ldots, \mathbf{g}_p^{(0)}\right]$ using ordinary least squares, and call the coefficient vector $\boldsymbol{\delta}^{(0)}$.
4. The new approximation to $\hat{\boldsymbol{\gamma}}$ is then $\boldsymbol{\gamma}^{(1)} = \boldsymbol{\gamma}^{(0)} + \boldsymbol{\delta}^{(0)}$.
5. Iterate the process until, (hopefully), convergence to the MLE, $\boldsymbol{\gamma}^{(i)} \to \hat{\boldsymbol{\gamma}}$.
6. The usual estimate of σ^2 is then

$$s^2 = \frac{\left\|\mathbf{y} - \mathbf{g}(\hat{\boldsymbol{\gamma}})\right\|^2}{n-p} = \frac{\mathrm{RSS}(\hat{\boldsymbol{\gamma}})}{n-p}$$

   by analogy with the linear regression case, (though it is neither the maximum likelihood estimate, nor a REML estimate).

A geometrical interpretation of the algorithm is as follows:

- Start at a point on the solution locus defined by the initial approximation.
- Calculate a basis for the tangent space at the initial approximation. The tangent space is a linear space which shares the same coordinate system as the solution locus itself.
- Project the response vector orthogonally onto the tangent space.
- The coordinates of the projection in the tangent space define a new approximation.
- Go to the point on the solution locus *corresponding* to the coordinates of the point on the tangent space, and repeat the process.
- If the process converges, the projection onto the tangent space should be at the same point as where he tangent space itself touches the solution locus, which implies that the point on the solution locus closest to the response vector, in the sense of Euclidean distance, has been obtained.

Inference and diagnostic checks

Most of the inferential procedures with non-linear regression are based on the assumption that the approximating tangential regression at the MLE is adequate, and linear regression theory is used. Thus nested models are tested using $\boldsymbol{F}-$tests, as if they were linear regressions. So called "profiling" of the likelihood (or least squares) surface can offer some check on these assumptions, by giving some insight into deviation from the expected quadratic behaviour.

The usual summary output from a non-linear least squares fit shows much the same information as for a linear regression.

The same diagnostic checks apply for non-linear regression as apply for linear. Plots of residuals against fitted values, and normal scores plots are just as important in non-linear regression as in linear. Other measures, such as Cook's distances and leverage, however, can only really be applied to the approximating tangential linear regression, and there are few ready-made tools for doing this. It would be easily possible, however, and this is a gap in the suite of `R` facilities that could usefully be plugged.

Notes on the software

Non-linear regression is a very general suite of models and, unlike generalized linear models, there are no reasonable automatic methods for generating good starting values for the iterative process. The fitting algorithms generally need a lot of input from the user to ensure the processes work.

The `R` fitting function, `nls`, provides facilities to assist the user in this. In particular

1. the `deriv` function allows the user to create a function to specify the form of the non-linear regression, and the partial derivatives of the model function with respect

to the parameters as well. Providing the partial derivatives allows the algorithm to find the tangent space directly. If these are not supplied, the algorithm has to find the tangent space by numerical differences, which can be a less stable process;

2. the `selfStart` function allows the user to include an automatic starting procedure in the model specification function as well as first derivatives. This not only makes the process simpler to use, but if the starting value procedure is good enough, it can make the iterative process safer and quicker.

   Several self-starting model functions have already been written for particular classes of non-linear regression models commonly in use. These are located in the `stats` package, along with `nls` itself.

```
> require(stats)

[1] TRUE

> objects("package:stats", patt = "^SS.*")

 [1] "SSasymp"     "SSasympOff"  "SSasympOrig" "SSbiexp"     "SSD"
 [6] "SSfol"       "SSfpl"       "SSgompertz"  "SSlogis"     "SSmicmen"
[11] "SSweibull"
```

3. Linear parameters have the property that, if the non-linear parameters are known (or fixed), they may then be estimated by linear least squares. If there are many linear parameters, it can be very important to take advantage of this simplification. The `plinear` algorithm in `nls` allows this. Using this algorithm the model funcion is supplied as a matrix, the columns of which the linear parameters multiply. That is, as the matrix $\mathbf{G}(\boldsymbol{\tau})$ defined here in an earlier section.