Nonlinear Regression and Nonlinear Least Squares

Appendix to *An R and S-PLUS Companion to Applied Regression*

John Fox

January 2002

1 Nonlinear Regression

The normal *linear* regression model may be written

$$y_i = \mathbf{x}_i'\boldsymbol{\beta} + \varepsilon_i$$

where $\mathbf{x}_i'$ is a (row) vector of predictors for the ith of n observations, usually with a 1 in the first position representing the regression constant; $\boldsymbol{\beta}$ is the vector of regression parameters to be estimated; and ε_i is a random error, assumed to be normally distributed, independently of the errors for other observations, with expectation 0 and constant variance: $\varepsilon_i \sim \text{NID}(0, \sigma^2)$.

In the more general normal *nonlinear regression model*, the function $f(\cdot)$ relating the response to the predictors is not necessarily linear:

$$y_i = f(\boldsymbol{\beta}, \mathbf{x}_i') + \varepsilon_i$$

As in the linear model, $\boldsymbol{\beta}$ is a vector of parameters and $\mathbf{x}_i'$ is a vector of predictors (but in the nonlinear regression model, these vectors are not generally of the same dimension), and $\varepsilon_i \sim \text{NID}(0, \sigma^2)$.

The likelihood for the nonlinear regression model is

$$L(\beta, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{n/2}} \exp\left\{ -\frac{\sum_{i=1}^n \left[y_i - f\left(\boldsymbol{\beta}, \mathbf{x}_i'\right)\right]^2}{2\sigma^2} \right\}$$

This likelihood is maximized when the sum of squared residuals

$$S(\boldsymbol{\beta}) = \sum_{i=1}^n \left[y_i - f\left(\boldsymbol{\beta}, \mathbf{x}_i'\right)\right]^2$$

is minimized. Differentiating $S(\boldsymbol{\beta})$,

$$\frac{\partial S(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -2 \sum \left[y_i - f\left(\boldsymbol{\beta}, \mathbf{x}_i'\right)\right] \frac{\partial f\left(\boldsymbol{\beta}, \mathbf{x}_i'\right)}{\partial \boldsymbol{\beta}}$$

Setting the partial derivatives to 0 produces estimating equations for the regression coefficients. Because these equations are in general nonlinear, they require solution by numerical optimization. As in a linear model, it is usual to estimate the error variance by dividing the residual sum of squares for the model by the number of observations less the number of parameters (in preference to the ML estimator, which divides by n).

Coefficient variances may be estimated from a linearized version of the model. Let

$$F_{ij} = \frac{\partial f\left(\widehat{\boldsymbol{\beta}}, \mathbf{x}_i'\right)}{\partial \widehat{\beta}_j}$$

and $\mathbf{F} = \{F_{ij}\}$. Then the estimated asymptotic covariance matrix of the regression coefficients is

$$\widehat{\mathcal{V}}(\widehat{\boldsymbol{\beta}}) = s^2(\mathbf{F}'\mathbf{F})^{-1}$$

where s^2 is the estimated error variance.

Bates and Watts (1988) provide a comprehensive reference on nonlinear regression and nonlinear least squares estimation; an accessible, brief treatment is Gallant (1975).

1.1 An Illustration

A simple model for population growth towards an asymptote is the logistic model

$$y_i = \frac{\beta_1}{1 + e^{\beta_2 + \beta_3 x_i}} + \varepsilon_i$$

where y_i is the population size at time x_i; β_1 is the asymptote towards which the population grows; β_2 reflects the size of the population at time $x = 0$ (relative to its asymptotic size); and β_3 controls the growth rate of the population.

2 The `nls` Function in S

The S function `nls` (located in the standard `nls` library in R) performs nonlinear least-squares estimation. Like the `lm` function, `nls` takes `formula`, `data`, `subset`, `weights`, and `na.action` arguments, but the right-hand side of the `formula` argument is treated as a standard algebraic expression rather than as a linear-model formula. There are additional, technical, arguments to `nls`: I discuss the arguments `start` and `trace` below; for further information, see `help(nls)`.

The data frame `US.pop` in the `car` library has decennial Census population data for the United States (in millions), from 1790 through 1990. The data are graphed in Figure 1 (a):

```
> library(car)
. . .
> data(US.pop)
> attach(US.pop)
> plot(year, population)
>
```

[The line in Figure 1 (a), which I shall add to the plot presently, represents the fit of the logistic population-growth model.]

To fit the logistic model to the U. S. Census data, we need start values for the parameters. It is often important in nonlinear least-squares estimation to choose reasonable start values, and this generally requires some insight into the structure of the model. We know that β_1 represents asymptotic population. The data in Figure 1 (a) show that in 1990 the U. S. population stood at about 250 million and did not appear to be close to an asymptote; so as not to extrapolate too far beyond the data, let us set the start value of β_1 to 350.

It is convenient to scale time so that $x_1 = 0$ in 1790, and so that the unit of time is 10 years. Then substituting $\beta_1 = 350$ and $x = 0$ into the model, using the value $y_1 = 3.929$ from the data, and assuming that the error is 0, we have

$$3.929 = \frac{350}{1 + e^{\beta_2 + \beta_3 0}}$$

Solving for β_2 gives us a plausible start value for this parameter:

$$\begin{aligned} e^{\beta_2} &= \frac{350}{3.929} - 1 \\ \beta_2 &= \log_e\left(\frac{350}{3.929} - 1\right) \simeq 4.5 \end{aligned}$$

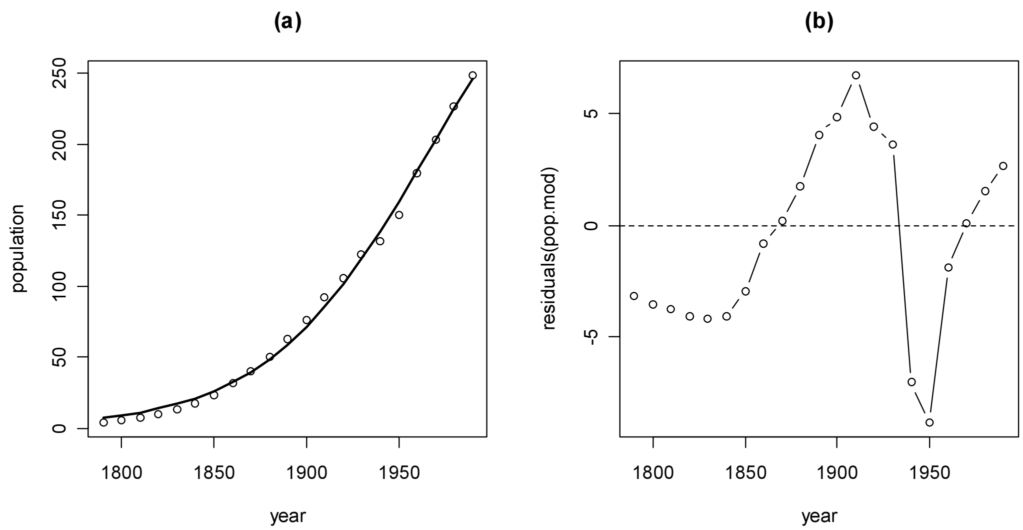

Figure 1: (a) U. S. population, showing the fit of the logistic growth model. (b) Residuals from the model.

Finally, returning to the data, at time $x = 1$ (i.e., at the second Census, in 1800), population was $y_2 = 5.308$. Using this value, along with the previously determined start values for β_1 and β_2, and again setting the error to 0, we have

$$5.308 = \frac{350}{1 + e^{4.5+\beta_3 1}}$$

Solving for β_3,

$$\begin{aligned} e^{4.5+\beta_3} &= \frac{350}{5.308} - 1 \\ \beta_3 &= \log_e \left(\frac{350}{5.308} - 1 \right) - 4.5 \simeq -0.3 \end{aligned}$$

Start values for `nls` are given via the `start` argument, which takes a named list of parameters. To fit the logistic growth model to the U. S. population data:

```
> library(nls)
> time <- 0:20
> pop.mod <- nls(population ~ beta1/(1 + exp(beta2 + beta3*time)),
+     start=list(beta1 = 350, beta2 = 4.5, beta3 = -0.3),
+     trace=T)
13007 :  350.0   4.5  -0.3
609.57 :  351.80749   3.84050  -0.22706
365.44 :  383.70453   3.99111  -0.22767
356.41 :  389.13503   3.98972  -0.22658
356.4 :  389.14629   3.99038  -0.22663
356.4 :  389.16653   3.99034  -0.22662
356.4 :  389.16551   3.99035  -0.22662

> summary(pop.mod)

Formula: population ~ beta1/(1 + exp(beta2 + beta3 * time))
```

```
Parameters:
      Estimate Std. Error t value Pr(>|t|)
beta1 389.1655    30.8120    12.6  2.2e-10
beta2   3.9903     0.0703    56.7  < 2e-16
beta3  -0.2266     0.0109   -20.9  4.6e-14

Residual standard error: 4.45 on 18 degrees of freedom

Correlation of Parameter Estimates:
       beta1  beta2
beta2 -0.166
beta3  0.915 -0.541
```

Setting `trace=T` produces a record of the iterative minimization of the residual sum of squares, along with the parameter estimates at each iteration. In this case, six iterations were required. The summary of the model gives coefficients, standard errors, the estimated error variance, and the correlations among the coefficients. The latter can be useful when `nls` has difficulty producing a solution: Very high correlations between coefficients are indicative of ill-conditioning.

Many familiar generic functions, such as `summary`, have methods for the nonlinear-model objects produced by `nls`. For example, the `fitted.values` function makes it simple to plot the fit of the model, adding a line to the graph in Figure 1 (a):

```
> lines(year, fitted.values(pop.mod), lwd=2)
>
```

Likewise, plotting residuals against time [Figure 1 (b)] suggests that while the logistic model reproduces the gross characteristics of the growth of the U. S. population to 1990, it misses the nuances:

```
> plot(year, residuals(pop.mod), type='b')
> abline(h=0, lty=2)
>
```

2.1 Specifying the Gradient

The process of maximizing the likelihood involves calculating the $n \times p$ *gradient* matrix $\mathbf{F} = \{\partial f\left(\widehat{\boldsymbol{\beta}}, \mathbf{x}_i'\right)/\partial \widehat{\beta}_j\}$. By default, `nls` computes the gradient numerically using a finite-difference approximation, but it is also possible to provide a formula for the gradient directly to `nls`. This is done by writing a function of the parameters and predictors that returns the fitted values of y, with the gradient as an attribute.

For the logistic-growth model, the partial derivatives of $f\left(\widehat{\boldsymbol{\beta}}, \mathbf{x}_i'\right)$ with respect to $\widehat{\boldsymbol{\beta}}$ are

$$\begin{aligned}
\frac{\partial f\left(\widehat{\boldsymbol{\beta}}, \mathbf{x}_i'\right)}{\partial \widehat{\beta}_1} &= [1+\exp(\widehat{\beta}_2+\widehat{\beta}_3 x_i)]^{-1} \\
\frac{\partial f\left(\widehat{\boldsymbol{\beta}}, \mathbf{x}_i'\right)}{\partial \widehat{\beta}_2} &= -\widehat{\beta}_1[1+\exp(\widehat{\beta}_2+\widehat{\beta}_3 x_i)]^{-2}\exp(\widehat{\beta}_2+\widehat{\beta}_3 x_i) \\
\frac{\partial f\left(\widehat{\boldsymbol{\beta}}, \mathbf{x}_i'\right)}{\partial \widehat{\beta}_3} &= -\widehat{\beta}_1[1+\exp(\widehat{\beta}_2+\widehat{\beta}_3 x_i)]^{-2}\exp(\widehat{\beta}_2+\widehat{\beta}_3 x_i)x_i
\end{aligned}$$

We may therefore proceed as follows, defining a function `model` for the right-hand side of the model (less the error), and producing the same results as before:

```
> model <- function(beta1, beta2, beta3, time){
+     model <- beta1/(1 + exp(beta2 + beta3*time))
```

```
+     term <- exp(beta2 + beta3*time)
+     gradient <- cbind((1 + term)^-1, # in proper order
+         -beta1*(1 + term)^-2 * term,
+         -beta1*(1 + term)^-2 * term * time)
+     attr(model, 'gradient') <- gradient
+     model
+     }

> summary(nls(population ~ model(beta1, beta2, beta3, time),
+     start=list(beta1=350, beta2=4.5, beta3=-0.3)))

Formula: population ~ model(beta1, beta2, beta3, time)

Parameters:
       Estimate Std. Error t value Pr(>|t|)
beta1 389.16551   30.81196   12.63 2.20e-10
beta2   3.99035    0.07032   56.74  < 2e-16
beta3  -0.22662    0.01086  -20.87 4.60e-14

Residual standard error: 4.45 on 18 degrees of freedom

Correlation of Parameter Estimates:
        beta1   beta2
beta2 -0.1662
beta3  0.9145 -0.5407
```

In many — perhaps most — cases, little is gained by this procedure, because the increase in computational efficiency is more than offset by the additional mathematical and programming effort required. It might be possible, however, to have one's cake and eat it too, by using the `deriv` function in S to compute a formula for the gradient and to build the requisite function for the right side of the model. For the example:

```
> model <- deriv(~ beta1/(1 + exp(beta2 + beta3*time)), # rhs of model
+     c('beta1', 'beta2', 'beta3'), # parameter names
+     function(beta1, beta2, beta3, time){} # arguments for result
+     )

> summary(nls(population ~ model(beta1, beta2, beta3, time),
+     start=list(beta1=350, beta2=4.5, beta3=-0.3)))

Formula: population ~ model(beta1, beta2, beta3, time)

Parameters:
       Estimate Std. Error t value Pr(>|t|)
beta1 389.16551   30.81196   12.63 2.20e-10
beta2   3.99035    0.07032   56.74  < 2e-16
beta3  -0.22662    0.01086  -20.87 4.60e-14

. . .
```

References

Bates, D. M. & D. G. Watts. 1988. *Nonlinear Regression Analysis and Its Applications*. New York: Wiley.

Gallant, A. R. 1975. "Nonlinear Regression." *The American Statistician* 29:73–81.