STAT 705 Nonlinear regression

Timothy Hanson

Department of Statistics, University of South Carolina

Stat 705: Data Analysis II

Chapter 13 Parametric nonlinear regression

Throughout most of STAT 704 and 705, we concentrated on linear models where $E(Y_i) = \mathbf{x}_i'\boldsymbol{\beta}$. Notable exceptions arose when we considered non-normal data. For logistic regression we had $E(Y_i) = e^{\mathbf{x}_i'\boldsymbol{\beta}}/[1 + e^{\mathbf{x}_i'\boldsymbol{\beta}}]$; Poisson regression gave us $E(Y_i) = t_i e^{\mathbf{x}_i'\boldsymbol{\beta}}$.

Sometimes scientists have a parametric non-linear mean function in mind for normal data. Theoretical considerations may lead to such a model, or else empirical evidence collected over time. Examples: dose-response models, growth curves, heating in swine due to MRI.

Parametric nonlinear regression

A parametric nonlinear model (13.1–13.5) has a prespecified parametric form indexed by parameters γ

$$Y_i = f(\mathbf{x}_i, \gamma) + \epsilon_i.$$

For example the exponential growth/decay model is $Y_i = \gamma_0 e^{\gamma_1 x_i} + \epsilon_i$. Data reduction takes place through the estimation of $\boldsymbol{\gamma} = (\gamma_0, \gamma_1)$ and σ.

Another example is the logistic growth curve $Y_i = \gamma_0[1 + \gamma_1 \exp(\gamma_2 x_i)]^{-1} + \epsilon_i$.

Note that model diagnostics are similar to the linear case, for example $r_i = Y_i - f(\mathbf{x}_i, \hat{\gamma})$ can be used to assess model adequacy.

Fitting parametric nonlinear models

Fitting of such models is carried out via maximum likelihood using Newton-Raphson. Several functions in SAS can carry this out, but PROC NLMIXED is the most versatile. Good starting values can make or break the program; need to think about what the parameters mean in the model.

There is a bit on fitting at the end of the logistic regression notes. In your book see pp. 517–521. This theory is covered in more detail in STAT 823 (large sample theory) and STAT 740 (advanced statistical computing).

PROC NLMIXED provides the MLE's as well as standard errors. Also, functions of parameters can be estimated as well.

Example (pp. 514–515) with SAS code

A hospital administrator wants to predict the degree of long-term recovery after discharge for severely injured patients; x_i is number of days hospitalized and Y_i is a prognostic index for long-term recovery (larger is better prognosis). Earlier studies suggest an exponential relationship $Y_i = \gamma_0 e^{\gamma_1 x_i} + \epsilon_i$.

```
data hosp;
input index days @@;
datalines;
  54   2  50   5  45   7  37  10  35  14  25  19  20  26  16  31
  18  34  13  38   8  45  11  52   8  53   4  60   6  65
;

proc sgscatter; plot index*days;

* starting values picked by looking at plot;
proc nlmixed data=hosp;
 parms g0=60 g1=-0.1 sigma=2;
 mu=g0*exp(g1*days);
 model index ~ normal(mu,sigma*sigma);
 predict g0*exp(g1*days) out=fit; * try predict (index-mu)/sigma out=res;

proc sgplot data=fit;
 scatter x=days y=index;
 series x=days y=pred;
```

Mixed effects nonlinear models

Note that *hierarchical* versions of these models can also be fit to repeated measures data. For example, if we have m hospitals instead of just one, there can be a separate curve for each hospital. Say $i = 1, \ldots, m$ denotes the hospital and $j = 1, \ldots, n_i$ denotes the patient within hospital i. A hierarchical model is

$$Y_{ij} = \gamma_{i0} \exp(\gamma_{i1} x_{ij}) + \epsilon_{ij},$$

where

$$\gamma_i = \left[\begin{array}{c} \gamma_{i0} \\ \gamma_{i1} \end{array} \right] \overset{iid}{\sim} N_2 \left(\left[\begin{array}{c} \gamma_0 \\ \gamma_1 \end{array} \right], \left[\begin{array}{cc} \sigma_{11} & \sigma_{12} \\ \sigma_{12} & \sigma_{22} \end{array} \right] \right).$$

This model can also be fit in PROC NLMIXED.

Learning example pp. 533–537

Electronics company is making something new at two locations, A is coded 1 and B is coded 0. Location B has more modern facilities and is expected to be more efficient than A after an initial learning period. The efficiency is measured by Y_i; x_{i1} is location and x_{i2} is the number of weeks that have gone by manufacturing this part. A model that allows for initial learning followed by a horizontal asymptote is

$$Y_i = \gamma_0 + \gamma_1 x_{i1} + \gamma_3 \exp(\gamma_2 x_{i2}) + \epsilon_i.$$

This is two exponential regressions with an intercept γ_0, the same efficiency growth rate γ_2, and a shifted vertical difference due to locations γ_1.

Crude starting values for Newton-Raphson maximization

To come up with crude starting values note that when γ_2 and γ_3 are less than zero, γ_0 is upper asymptote for location B and $\gamma_0 + \gamma_1$ is upper asymptote for location A.

Based on an initial scatterplot set $\gamma_0 = 1$, $\gamma_1 = -0.05$, and $\sigma = 0.02$. The exponential is "used up" by 30 weeks, maybe try $\gamma_2 = -0.1$.

γ_3 is hard to think about, maybe just try $\gamma_3 = -1$?

Learning example in SAS

```
data learn;
input location week efficiency @@;
datalines;
  1   1   .483  1   2   .539  1   3   .618  1   5   .707  1   7   .762  1  10   .815
  1  15   .881  1  20   .919  1  30   .964  1  40   .959  1  50   .968  1  60   .971
  1  70   .960  1  80   .967  1  90   .975  0   1   .517  0   2   .598  0   3   .635
  0   5   .750  0   7   .811  0  10   .848  0  15   .943  0  20   .971  0  30  1.012
  0  40  1.015  0  50  1.007  0  60  1.022  0  70  1.028  0  80  1.017  0  90  1.023
;

proc sgscatter data=learn; plot efficiency*week / group=location;

proc nlmixed data=learn;
 parms g0=1 g1=-0.05 sigma=0.02 g2=-0.1 g3=-1;
 mu=g0+g1*location+g3*exp(g2*week);
 model efficiency ~ normal(mu,sigma*sigma);
 predict g0+g1*location+g3*exp(g2*week) out=fit;

proc sgplot data=fit;
 scatter x=week y=efficiency / group=location;
 series x=week y=pred / group=location;
```

Problem from 2010 MS/PhD qualifying exam

The May 2010 qualifying exam (part II) has a nice problem.

```
data snake;
input conc rate @@;
datalines;
 31.25 53.01 62.5 81.42 125 122.11 250 304.57 500 376.87
 1000 414.13 2000 553.46
;

proc sgscatter; plot rate*conc;

proc nlmixed data=snake;
 parms b1= b2= b3= sigma=; * let’s find these in class;
 mu=b1/(1+(b2/conc)**b3);
 model rate ~ normal(mu,sigma*sigma);
 predict b1/(1+(b2/conc)**b3) out=fit;
 estimate "mean rate at conc=750" b1/(1+(b2/750)**b3);

proc sgplot data=fit;
 scatter x=conc y=rate;
 series x=conc y=pred;
```

Nonparametric regression

Consider a continuous response with three predictors (although these methods can be extended to other types of response).

An additive model stipulates

$$Y_i = \mu + f_1(x_{i1}) + f_2(x_{i2}) + f_3(x_{i3}) + \epsilon_i,$$

and seeks to estimate the functions $f_1(x)$, $f_2(x)$, and $f_3(x)$ (typically via splines). These are fit in `proc gam` and `proc transreg`. Can also consider transformation of Y_i as well as pairwise interaction surfaces.

Nonparametric regression

A parametric nonlinear model (Chapter 13) has a prespecified parametric form indexed by parameters $\boldsymbol{\gamma}$

$$Y_i = f(\mathbf{x}_i, \boldsymbol{\gamma}) + \epsilon_i.$$

For example the exponential growth/decay model is $Y_i = \gamma_0 e^{\gamma_1 x_i} + \epsilon_i$. Data reduction takes place through the estimation of $\boldsymbol{\gamma}$ and σ.

Nonparametric regression is essentially unspecified

$$Y_i = f(\mathbf{x}_i) + \epsilon_i,$$

and seeks to estimate $f(\mathbf{x}) : \mathbb{R}^k \to \mathbb{R}$ directly. Two useful and popular methods are *lowess* and *kernel smoothing*.

Kernel smoothing

Let's start with a univariate predictor yielding data $\{(x_i, Y_i)\}_{i=1}^n$. At each $x \in \mathbb{R}$, the kernel-smoothed estimate of $f(\cdot)$ is a weighted average of the Y_i's:

$$\hat{f}_h(x) = \sum_{i=1}^{n} \left[\frac{k\{(x_i - x)/h\}/h}{\sum_{j=1}^n k\{(x_j - x)/h\}/h} \right] Y_i.$$

Here, $k(d)$ is the kernel. Common choices are Gaussian $k(d) = e^{-0.5d^2}$ (most common), uniform $k(d) = I\{|d| < 1\}$, and Epanechnikov $k(d) = 0.75(1 - d^2)I\{|d| < 1\}$ (there are many more). Different kernel functions simply weight neighboring points differently.

Bandwidth

The parameter h is called the bandwidth. The larger the bandwidth, the smoother the estimate $\hat{f}_h$. What happens to $\hat{f}_h$ as $h \to \infty$? Is it possible to have $\hat{f}_h(x)$ outside the range of Y_i-values?

A common way to choose the bandwidth is through cross-validation, $\hat{h} = \text{argmin}_{h>0} \sum_{i=1}^{n} (Y_i - \hat{f}_{h,i}(x_i))^2$ where $\hat{f}_{h,i}$ is the kernel-smoothed estimate based on the $(n-1)$ pairs $\{(x_j, Y_j)\}_{j \neq i}$.

`ksmooth` in R gives kernel-smoothed regression estimates without standard errors. A great package that does a lot more (including handling categorical predictors) is `np`. You need to install it from CRAN.

Long-term recovery example in R with Gaussian kernel-smoothing

Recall that Y_i is prognostic index and x_i is days hospitalized. The default bandwidth h selection is cross-validation. The default kernel is Gaussian.

```
library(np)
index=c(54,50,45,37,35,25,20,16,18,13,8,11,8,4,6)
days=c(2,5,7,10,14,19,26,31,34,38,45,52,53,60,65)
fit1=npreg(index~days)
plot(fit1,plot.errors.method="asymptotic",plot.errors.style="band",main="Kernel-smoothed")
points(days,index)
```

11.4 **LO**cally **WE**ighted **S**catterplot **S**moothing (lowess)

Kernel-smoothing is biased at the boundaries $\min\{x_i\}$ and $\max\{x_i\}$, and at the extrema of $f(\cdot)$. A method that solves some of these issues uses locally fitted polynomials to estimate $f(x)$ at each x via weighted least squares (WLS). Lowess was introduced by Cleveland (1979).

Recall that weighted least squares weights some pairs (x_i, Y_i) more heavily when "more information" is known about Y_i, e.g. $var(Y_i)$ is smaller than for other values. The weight w_i attached to (x_i, Y_i) is the *i*th diagonal of the matrix $\mathbf{W}$; the remaining elements are zero. The weighted least squares estimate of $\boldsymbol{\beta}$ is given by $\hat{\boldsymbol{\beta}} = (\mathbf{XWX}')^{-1}\mathbf{X}'\mathbf{WY}$.

lowess

Consider estimating $f(x)$ at x with a linear or quadratic function. If we assume that pairs (x_i, Y_i) have more information for $f(x)$ at values of x_i near x, we can weight them more using WLS. The most common weight function is tricube

$$w_i(x) = \left\{ \begin{array}{cc} [1 - (|x - x_i|/d_q(x))^3]^3 & |x - x_i| < d_q(x) \\ 0 & |x - x_i| > d_q(x) \end{array} \right\}.$$

$d_q(x)$ is a distance such that the proportion of x_i values within x is q, i.e. $d_q(x) = \min\{d > 0 : \frac{1}{n}\sum_{i=1}^n I\{|x_i - x| < d\} \geq q\}$. A common choice of q is 0.5 (p. 450).

The lowess estimate of $f(x)$, assuming local linear fitting, is then $\hat{f}(x) = [\ 1 \quad x\ ](\mathbf{X}\mathbf{W}(x)\mathbf{X}')^{-1}\mathbf{X}'\mathbf{W}(x)\mathbf{Y}$ where $\mathbf{W}(x) = \text{diag}(w_1(x), \ldots, w_n(x))$ and the ith row of bX is $[\ 1 \quad x_i\ ]$. For *each value* of x, a separate WLS is fitted – lowess requires *a lot* of computation!

Long-term recovery example in R with lowess

Uses defaults. An older function is `lowess`; `loess` has improvements on `lowess` but gives essentially the same answers.

```
index=c(54,50,45,37,35,25,20,16,18,13,8,11,8,4,6)
days=c(2,5,7,10,14,19,26,31,34,38,45,52,53,60,65)
fit2=loess(index~days)
pred.days=seq(2,65,1)
pred2=predict(fit2,pred.days,se=TRUE)
plot(pred.days,pred2$fit,type="l",xlab="Days",ylab="Index",main="Lowess Fit")
lines(pred.days,pred2$fit-1.96*pred2$se.fit,lty=3)
lines(pred.days,pred2$fit+1.96*pred2$se.fit,lty=3)
points(days,index)
```

Similarities between lowess and kernel-smoothing

Both kernel-smoothing and lowess have weight functions and bandwidths that determine how points in a neighborhood of x are weighted.

Both estimates are written as $\hat{f}(x) = \mathbf{c}(x)'\mathbf{Y}$, i.e. are linear combinations of the Y_i's, that depend on x. In STAT 704 regression, $\hat{f}(x) = \mathbf{c}(x)'\mathbf{Y}$ where $\mathbf{c}(x)' = [\ 1 \quad x\ ](\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'$. Note that kernel-smoothing provides a true average of the Y_i's at each point, whereas lowess values of $c_i(x)$ may be negative or greater than one.

Both methods are generalized to more than one predictor similarly. Predictors are standardized to have variance one and Euclidean distance $d = ||\mathbf{x} - \mathbf{x}^*||$ is used in the weight function rather than $|x - x^*|$, or else the Mahalanobis distance used $d = \sqrt{(\mathbf{x} - \mathbf{x}^*)'\mathbf{S}^{-1}(\mathbf{x} - \mathbf{x}^*)}$ (no need to standardized first). Note that categorical predictors need some thought.

Questions and comments

- Is extrapolation a good idea with lowess or kernel-smoothed methods?
- The asymptotics for nonparametric smoothing methods is worth an entire course. A bit is covered it STAT 824 (nonparametrics).
- Which method, lowess or kernel-smoothing, is more appropriate for Bernoulli data? Why?
- There's some nice animation here: http://www.r-bloggers.com/some-heuristics-about-local-regression-and-kernel-smoothing/
- A method worthy of its own lecture is *basis expansions*. Basis expansions write the unknown $f(\cdot)$ as $f(\mathbf{x}) = \sum_{k=1}^{K} \beta_k \phi_k(\mathbf{x})$ for a set of known functions $\phi_k(\cdot)$. The unknown parameters are $\beta_1, \ldots, \beta_K$. *This yields a linear model*.
- Example basis expansions include polynomials, Legendre polynomials, wavelets, sines and cosines, and B-splines.