

Stat 511 HW#6 Spring 2004 (Corrected)

1. The book *Nonlinear Regression Analysis and its Applications* by Bates and Watts contains a small data set taken from an MS thesis of M.A. Treloar “Effects of Puromycin on Galactosyltransferase of Golgi Membranes.” It is reproduced below. y is reaction velocity (in counts/min²) for an enzymatic reaction and x is substrate concentration (in ppm) for untreated enzyme and enzyme treated with Puromycin.

x		0.02	0.06	0.11	0.22	0.56	1.10
y	Untreated	67, 51	84, 86	98, 115	131, 124	144, 158	160
	Treated	76, 47	97, 107	123, 139	159, 152	191, 201	207, 200

Apparently a standard model here (for either the untreated enzyme or for the treated enzyme) is the “Michaelis-Menten model”

$$y_i = \frac{\theta_1 x_i}{\theta_2 + x_i} + \varepsilon_i \quad (*)$$

Note that in this model, 1) the mean of y is 0 when $x = 0$, 2) the limiting (large x) mean of y is θ_1 , and 3) the mean of y reaches half of its limiting value when $x = \theta_2$.

Begin by considering only the “Treated” part of the data set (and an iid $N(0, \sigma^2)$ ε_i 's version of the model). Of course, use R to help you do all that follows. Begin by reading in 12×1 vectors y and x .

a) Plot y vs x and make “eye-estimates” of the parameters based on your plot and the interpretations of the parameters offered above. (Your eye-estimate of θ_1 is what looks like a plausible limiting value for y , and your eye-estimate of θ_2 is a value of x at which y has achieved half its maximum value.)

b) Add the `nls` package to your R environment. Then issue the command

```
> REACT.fm<-
nls(formula=y~theta1*x/(theta2+x), start=c(theta1=#, theta2=##), trace=T)
```

where in place of # and ## you enter your eye-estimates from a). This will fit the nonlinear model (*) via least squares. What are the least squares estimate of the parameter vector and the “deviance” (error sum of squares)

$$\hat{\theta}_{OLS} = \begin{pmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \end{pmatrix} \text{ and } SSE = \sum_{i=1}^{12} \left(y_i - \left(\hat{\theta}_1 x_i / (\hat{\theta}_2 + x_i) \right) \right)^2$$

c) Re-plot the original data with a superimposed plot of the fitted equation. To do this, you may type

```
> conc<-seq(0,1.5,.05)
> velocity<-coef(REACT.fm)[1]*conc/(coef(REACT.fm)[2]+conc)
> plot(c(0,1.5),c(0,250),type="n",xlab="Conc (ppm)",ylab="Vel
(counts/sqmin)")
> points(x,y)
```

```
> lines(conc,velocity)
```

(The first two commands set up vectors of points on the fitted curve. The third creates an empty plot with axes appropriately labeled. The fourth plots the original data and the fifth plots line segments between points on the fitted curve.)

d) Get more complete information on the fit by typing

```
> summary(REACT.fm)
> vcov(REACT.fm)
```

Verify that the output of this last call is $MSE(\hat{\mathbf{D}}'\hat{\mathbf{D}})^{-1}$ and that the standard errors produced by the first are square roots of diagonal elements of this matrix. Then use the information produced here and make an approximate 95% prediction interval for one additional reaction velocity, for substrate concentration .50 ppm.

e) The concentration, say x_{100} , at which mean reaction velocity is 100 counts/min² is a function of θ_1 and θ_2 . Find a sensible point estimate of x_{100} and a standard error (estimated standard deviation) for your estimate.

f) As a means of visualizing what function the R routine `nls` minimized in order to find the least squares coefficients, do the following. First set up a grid of (θ_1, θ_2) pairs as follows. Type

```
> theta<-coef(REACT.fm)
> se<-sqrt(diag(vcov(REACT.fm)))
> dv<-deviance(REACT.fm)
> gsize<-101
> th1<-theta[1]+seq(-4*se[1],4*se[1],length=gsiz)
> th2<-theta[2]+seq(-4*se[2],4*se[2],length=gsiz)
> th<-expand.grid(th1,th2)
```

Then create a function to evaluate the sums of squares

```
> ss<-function(t)
+ {
+ sum((y-t[1]*x/(t[2]+x))^2)
+ }
```

As a check to see that you have it programmed correctly, evaluate this function at $\hat{\boldsymbol{\theta}}_{OLS}$ for the data in hand, and verify that you get the deviance. Then evaluate the error sum of squares over the grid of parameter vectors $\boldsymbol{\theta}$ set up earlier and produce a contour plot using

```
> SumofSquares<-apply(th,1,ss)
> SumofSquares<-matrix(SumofSquares,gsiz,gsiz)
> plot(th1,th2,type="n",main="Error Sum of Squares Contours")
> contour(th1,th2,SumofSquares,levels=c(seq(1000,4000,200)))
```

What contour on this plot corresponds to an approximately 90% approximate confidence region for the parameter vector $\boldsymbol{\theta}$?

g) Now redo the contour plotting, placing only two contours on the plot using the following code.

```
> plot(th1, th2, type="n", main="Error Sum of Squares Contours")
> contour(th1, th2, SumofSquares, levels=dv*c((1+.1*qf(.95, 1, 10)),
(1+.2*qf(.95, 2, 10))))
```

Identify on this plot an approximately 95% (joint) confidence region for θ and individual 95% confidence regions for θ_1 and θ_2 . (By the way, it would have been possible to simply add these contours to first plot, by making the second call to `contour()` as above, except for setting “add=T” as a parameter of the call.)

h) Use the standard errors for the estimates of the coefficients produced by the routine `nls()` and make 95% t intervals for θ_1 and θ_2 . How much different are these from your intervals in g)?

(Notice that the sample size in this problem is small and reliance on *any version of* large sample theory to support inferences is tenuous. I would take any of these inferences above as *very approximate*. We will later discuss the possibility of using "bootstrap" calculations as an alternative method of inference.)

i) Make two different approximate 95% confidence intervals for σ . Base one on carrying over the linear model result that $SSE / \sigma^2 \sim \chi_{n-k}^2$. Base the other on the “profile likelihood” material.

j) Use the R function `confint()` to get 95% intervals for θ_1 and θ_2 . That is, add the MASS package in order to get access to the function. Then type

```
> confint(REACT.fm, level=.95)
```

How do these intervals compare to the ones you found in part g)?

k) Apparently, scientific theory suggests that treated enzyme will have the same value of θ_2 as does untreated enzyme, but that θ_1 may change with treatment. That is, if

$$z_i = \begin{cases} 0 & \text{if treated (Puromycin is used)} \\ 1 & \text{otherwise} \end{cases}$$

a possible model is

$$y_i = \frac{(\theta_1 + \theta_3 z_i) x_i}{\theta_2 + x_i} + \varepsilon_i$$

and the parameter θ_3 then measures the effect of the treatment. Go back to the data table and now do a fit of the (3 parameter) nonlinear model including a possible Puromycin effect using all 23 data points. Make 2 different approximately 95% confidence intervals for θ_3 . Interpret these. (Do they indicate a statistically detectable effect? If so, what does the sign say about how treatment affects the relationship between x and y ?) Plot on the same set of axes the curves

$$y = \frac{\hat{\theta}_1 x}{\hat{\theta}_2 + x} \quad \text{and} \quad y = \frac{(\hat{\theta}_1 + \hat{\theta}_3) x}{\hat{\theta}_2 + x} \quad \text{for } 0 < x < 2$$

l) NOT REQUIRED ... “for fun.” A confidence region for a parameter vector can be turned into a confidence region for anything that is computable based on the parameter vector. If you are looking for something challenging to do, try this: Use the approximately 95% confidence region for θ created in g) and make a corresponding 95% (simultaneous) confidence band for the function

$$y = \frac{\theta_1 x}{\theta_2 + x}$$

You will have to do the following kind of R programming: For a grid of x values (covering, say, $(0,2)$) one x at a time, compute values of y for every θ on the grid created earlier in this problem *that is inside the confidence set* (checking that a point is inside the region requires you to see how big the corresponding error sum of squares is). Save the smallest such value of y (call it $l(x)$) and the largest such value of y (call it $u(x)$) for each x . Then plot both

$$l(x) \text{ vs } x \text{ and } u(x) \text{ vs } x$$

on a single set of axes. The result is a (simultaneous) 95% confidence band for the mean response function.