Fitting non-linear functions

There are two ways to fit a non-linear function. We can transform the variables in such a way that the function we want to fit becomes linear or we can fit the non-linear function directly. The former method is the BC (before computers) method but can still be appropriate. We look at this method first.

Consider the following example. The built-in dataset cars has data on the speed (speed) and stopping distance (dist) of 50 cars (in the 1920’s). Obviously these two variables are not linearly related. Try the following plots:

```r
> xyplot(dist~speed,data=cars)
> xyplot(log(dist)~log(speed),data=cars)
```

The second relationship looks more linear. In other words, we might want to fit this model:

\[
\log(\text{dist}) = \beta_0 + \beta_1 \log(\text{speed}) + \epsilon
\]

Notice that this model is linear in the variables \(\log(\text{speed})\) and \(\log(\text{dist})\) so \texttt{lm} can be used to find \(\beta_0\) and \(\beta_1\).

```r
> llog=lm(log(dist)~log(speed), data=cars)
> llog
Call:
  lm(formula = log(dist) ~ log(speed), data = cars)
Coefficients:
(Intercept) log(speed)
-0.7297 1.6024
```

On the other hand, you might think that physics would suggest the following model:

\[
\text{dist} = \beta_0 + \beta_1 \text{speed} + \beta_2 \text{speed}^2 + \epsilon
\]

Again \texttt{lm} can be used to find the coefficients.

```r
> lone=lm(dist~speed,data=cars)
> ltwo=lm(dist~speed+I(speed^2),data=cars)
> ltwo
Call:
  lm(formula = dist ~ speed + I(speed^2), data = cars)
Coefficients:
(Intercept) speed I(speed^2)
 2.47014 0.91329 0.09996
```

Again \texttt{lm} can be used to find the coefficients.

```r
> lthree=lm(dist~speed+I(speed^2),data=cars)
> lthree
Call:
  lm(formula = dist ~ speed + I(speed^2), data = cars)
Coefficients:
(Intercept) speed I(speed^2)
 2.47014 0.91329 0.09996
```

On the other hand, you might think that physics would suggest the following model:

\[
\text{dist} = \beta_0 + \beta_1 \text{speed} + \beta_2 \text{speed}^2 + \epsilon
\]

Again \texttt{lm} can be used to find the coefficients.

```r
> lthree=lm(dist~speed+I(speed^2),data=cars)
> lthree
Call:
  lm(formula = dist ~ speed + I(speed^2), data = cars)
Coefficients:
(Intercept) speed I(speed^2)
 2.47014 0.91329 0.09996
```

Notice that in this case, the \(F\)-test tells us that the model with the quadratic term is probably not warranted – that is we do not have statistical evidence to suggest that \(\beta_2 \neq 0\).
Michaelis-Menten Equation

The Michaelis-Menten equation relates the rate $v$ of an enzymatic reaction to the concentration $S$ of the substrate in which the reaction occurs. The equation is

$$v = \frac{V_{\text{max}} S}{K_m + S}$$

In this equation, $V_{\text{max}}$ and $K_m$ are constants that depend on the particular reaction being described. Writing this equation in more familiar notation (with $y$ for the velocity, $x$ for the concentration), and $\beta_0, \beta_1$ for the parameters, we have:

$$y = \frac{\beta_0 x}{\beta_1 + x} \quad (1)$$

We are going to solve this in two different ways. We will use the dataframe `reaction.csv` which is in Stob’s data files.

Transforming the equation to a linear one

1. Plot Velocity versus Conc and notice the non-linearity of the relationship. Use the fact that

$$\lim_{x \to \infty} \frac{\beta_0 x}{\beta_1 + x} = \beta_0$$

   to estimate $\beta_0$.

2. A classical solution to this problem (the Lineweaver-Burk method) starts by noticing that a plot of $1/y$ versus $1/x$ is approximately linear. Do that plot for these data.

3. Rewrite equation 1 as an equation for $1/y$ in terms of $1/x$. Notice that the resulting equation is a linear equation. What are the slope and intercept in terms of $\beta_0$ and $\beta_1$?

4. We can use R to estimate the slope and intercept of this line.

   ```r
   > lm(1/y ~ I(1/x), data=reaction)
   ```

5. Use the result of the linear model to give estimates for $\beta_0$ and $\beta_1$. 
Non-linear function fitting

Instead of linearizing, we might want to find \( \beta_0 \) and \( \beta_1 \) directly to minimize the sums of the squares of the residuals. R has a function to do that - `nls`. The difficulty with non-linear function fitting is that there usually is not closed-form solution to the minimization problem. Instead what is usually done is to find the best fit using an iterative process (think Newton’s method). Iterative processes usually need a starting guess for the unknowns that is sufficiently close to the minimum so that it converges.

6. Try the following and compare the solution that you got here with that using the Lineweaver-Burk method.

```r
> l=nls(Velocity~ Gamma0*Conc/(Gamma1+Conc), start=list(Gamma0=10,Gamma1=10),data=reaction)
> l
```

7. Both solutions (this one and the Lineweaver-Burk) method find the solution by minimizing the sums of the squares of the residuals. Why aren’t the solutions the same?

Men’s Track Records

The dataframe `mentrack` gives the world record time in Seconds for various length track events in Meters. Some people have proposed the following relationship between the two variables:

\[
\text{Meters} = \beta_0 \text{Seconds}^{\beta_1}
\]

Use each of the two methods developed above to find approximate values for \( \beta_0 \) and \( \beta_1 \).