Quantitative Understanding in Biology Module II: Model Parameter Estimation Lecture III: Quantitative Comparison of Models

A classic mathematical model for enzyme kinetics is the Michaelis-Menten equation:

$$V = \frac{V_{max}\,[S]}{K_m + [S]}$$

Given data of V versus [S] for a particular enzyme and substrate, we can determine the Michaelis-Menten parameters V_{max} and K_m using a regression procedure. Consider, for example, the following data for the association of myoglobin and oxygen.

P_{O2} (torr)	1.1	1.5	1.6	2.3	3.4	5.3	7.5	8.4	14.1
$[O_2]$ (mL/dL)	1.49	1.79	1.79	2.11	2.83	3.42	3.79	3.97	4.08

We begin by entering the data into R and inspecting a basic plot.

```
> myoglobin <- data.frame(s=c(1.1,1.5,1.6,2.3,3.4,5.3,7.5,8.4,14.1),
v=c(1.49,1.79,1.79,2.11,2.83,3.42,3.79,3.97,4.08))
> plot(v~s, data=myoglobin, xlim=c(0,15), ylim=c(0,5))
```

Note that here we explicitly give limits for the ordinate and abscissa. In this case, allowing R to choose axes limits results in a deceiving plot.

Next, we proceed with a non-linear regression in the normal fashion. Initial guesses for V_{max} and K_m are based on a quick glance at the plot. Recall that V_{max} is the maximal reaction rate, and K_m is the value of [S] at which half of V_{max} is realized. A plot of the curve predicted by the model is consistent with the observed data.

```
> m0.myoglobin <- nls(v ~ Vmax * s / (Km + s),
start=list(Vmax=4,Km=2), data=myoglobin)
> summary(m0.myoglobin)

Formula: v ~ Vmax * s/(Km + s)

Parameters:
     Estimate Std. Error t value Pr(>|t|)
Vmax   5.1171     0.1678   30.50 1.05e-08 ***
Km     2.8282     0.2511   11.26 9.72e-06 ***
```

```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1244 on 7 degrees of freedom

Number of iterations to convergence: 3
Achieved convergence tolerance: 6.637e-06

> x <- 0:15; lines(x, predict(m0.myoglobin, newdata=data.frame(s=x)),
col="blue", lwd=2)
```

Furthermore, we proceed with quality control plots of residuals vs. fitted values, and a QQ-plot of the residuals. It is difficult to evaluate these plots with so few data points, and the best that we can hope for is that we see nothing overtly wrong. For good measure, we also perform a Shapiro test for normality of the residuals.

```
> plot(residuals(m0.myoglobin) ~ predict(m0.myoglobin))
> qqnorm(residuals(m0.myoglobin))
> qqline(residuals(m0.myoglobin))
> shapiro.test(residuals(m0.myoglobin))

        Shapiro-Wilk normality test

data:  residuals(m0.myoglobin)
W = 0.8802, p-value = 0.1578
```

Finally, we check the confidence intervals of the model parameters to ensure that they include only physiologically feasible values, and that they are not too wide considering the small amount of data we have to work with.

```
> confint(m0.myoglobin)
Waiting for profiling to be done...
         2.5%    97.5%
Vmax 4.747424 5.535149
Km   2.296327 3.476164
```

In this case, we are reasonably happy with our results. Ideally, we’d like a bit more data to rule out any systematic variation in the residuals and heteroschadisticity, but otherwise we are satisfied with our fit.

We will now repeat this exercise for similar data taken for hemoglobin. The experimental observations are:

P_{O2} (torr)	2	10	18	20	31	42	50	60	80	98
$[O_2]$ (mL/dL)	0.4	2.0	5.6	6.2	11.0	15.0	16.8	18.2	19.0	18.8

```
> hemoglobin=data.frame(s=c(2,10,18,20,31,42,50,60,80,98),
 v=c(0.4,2.0,5.6,6.2,11.0,15.0,16.8,18.2,19.0,18.8))
> plot(v ~ s, data=hemoglobin)
```

The plot of the raw data already suggests a sigmoidal shape that may not be consistent with our model. However, this could be just noise in the model, so we proceed objectively with a similar fit as before.

```
> m0.hemoglobin <- nls(v ~ Vmax * s / (Km + s),
start=list(Vmax=19,Km=40), data=hemoglobin)
> x <- 0:100
> lines(x, predict(m0.hemoglobin, newdata=data.frame(s=x)),
col="blue", lwd=2)
```

Here we see that the model curve does not fit the data too well. While we could proceed with our quality control plots, for current purposes we'll stop here and reconsider the model. It turns out that if you assume that hemoglobin is a multimeric protein, and that affinity for binding at different sites is not independent, you get a more elaborate form of the Michaelis-Menten relationship, called the Hill model:

$$V = \frac{V_{max}\,[S]^n}{{K_m}^n + [S]^n}$$

The exponent, n, is called the Hill exponent, and is an indication of the degree of cooperativity the system exhibits. If n>1, the system is said to exhibit positive cooperativity; if n<1, the system exhibits negative cooperativity.

We also see that when n=1, the model reduces to the Michaelis-Menten model. In other words, the Michaelis-Menten model is a special case of the Hill model. This relationship between the models is important, and has a specific term: the models are said to be 'nested'.

We proceed to fit the Hill model to our data:

```
> m1.hemoglobin <- nls(v ~ Vmax * s^n / (Km^n + s^n),
start=list(Vmax=19, Km=40, n=1), data=hemoglobin)
Error in numericDeriv(form[[3]], names(ind), env) :
  Missing value or an infinity produced when evaluating the model
> m1.hemoglobin <- nls(v ~ Vmax * s^n / (Km^n + s^n),
start=list(Vmax=19, Km=25, n=1), data=hemoglobin)
> summary(m1.hemoglobin)

Formula: v ~ Vmax * s^n/(Km^n + s^n)

Parameters:
     Estimate Std. Error t value Pr(>|t|)
Vmax  20.3000     0.5683   35.72 3.50e-09 ***
Km    27.5289     1.0494   26.23 2.99e-08 ***
n      2.4347     0.1789   13.61 2.72e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4781 on 7 degrees of freedom
```

```
Number of iterations to convergence: 8
Achieved convergence tolerance: 1.159e-06
```

Note that on our first attempt, the iterative numerical procedure failed to converge. A more informed initial guess met with success. Getting such optimization to converge is sometimes more art than science, and occasionally it is not possible. Some tips to aid in convergence are:

1) Plot the curve predicted by the model at the initial guess, and adjust the parameters “by hand” to get a decent starting guess.
2) Try fitting the model with one or more of the parameters fixed. Then use the optimized values for the remaining parameters as starting points for a full optimization.
3) Make use of the `trace=TRUE` option in the `nls` function.
4) Try different algorithms; the `nls` function supports `algorithm=”plinear”` and `algorithm=”port”.`

Also note that `nls` is designed to work with real data that contain some noise. If your data were generated from a function and all of the residuals were zero, `nls` would probably fail. This is counter-intuitive, as you would expect most optimizations to perform well when the error is zero. The reason for this is that R not only finds the best values for the parameters, but also makes estimates of their uncertainty; some non-zero residuals are necessary to avoid mathematical singularities.

We now plot the model-predicted curve, and inspect the confidence intervals of the parameters.

```
> lines(x, predict(m1.hemoglobin, newdata=data.frame(s=x)), col="red",
lwd=2)
> confint(m1.hemoglobin)
Waiting for profiling to be done...
          2.5%     97.5%
Vmax 19.145516 21.788665
Km   25.383562 30.266468
n     2.046156  2.879301
```

Everything seems reasonable so far. Additional checks on the model are left as an exercise.

We should note here that the physiologically accepted value of the Hill constant for hemoglobin is between 2.5 and 3.0.

Given the success of our Hill model, and given that regular Michaelis-Menten kinetics are a special case of the Hill model, one might wonder why we don’t just always use the Hill model. After all, if the system does not demonstrate cooperativity, the regression will tell us by reporting a Hill exponent of unity.

Let’s try this approach with our myoglobin data. Our initial guess is informed by our previous run:

```
> m1.myoglobin <- nls(v ~ Vmax * s^n / (Km^n + s^n),
start=list(Vmax=5.1, Km=2.8, n=1), data=myoglobin)
> summary(m1.myoglobin)

Formula: v ~ Vmax * s^n/(Km^n + s^n)
```

```
Parameters:
     Estimate Std. Error t value Pr(>|t|)
Vmax   4.7768     0.3532  13.523 1.01e-05 ***
Km     2.4393     0.3860   6.319 0.000734 ***
n      1.1398     0.1606   7.099 0.000392 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1252 on 6 degrees of freedom

Number of iterations to convergence: 7
Achieved convergence tolerance: 4.982e-06
> plot(v~s, data=myoglobin, xlim=c(0,15), ylim=c(0,5))
> x <- (0:30)/2; lines(x, predict(m0.myoglobin,
newdata=data.frame(s=x)), col="red")
> x <- (0:30)/2; lines(x, predict(m1.myoglobin,
newdata=data.frame(s=x)), col="blue")
```

Here we see that the three parameter model fits the data almost as well as the two-parameter model. This can be quantified by looking the sum of the squares of the residuals (the objective function of the implicit optimizations we are performing whenever we run `nls`).

```
> sum(residuals(m0.myoglobin)^2)
[1] 0.1083247
> sum(residuals(m1.myoglobin)^2)
[1] 0.09411742
```

Inspecting the CIs for the parameters is informative as well:

```
> confint(m1.myoglobin)
Waiting for profiling to be done...
          2.5%    97.5%
Vmax 4.2146648 6.173016
Km   1.9106186 4.555201
n    0.7903136 1.532182
```

The first thing that you should notice is that the CI for the Hill exponent is quite wide; we could have reasonably significant positive or negative cooperativity. Comparing the CIs for the other parameters with those from the two-parameter model shows that the uncertainty in the three-parameter model is significantly larger. This alone is a reason for rejecting the three parameter model if we can; it will reduce the uncertainty in the two parameter model. However, a more compelling argument is that of maximum parsimony, or Occam’s razor. Given a choice between two models, if we don’t have good evidence to support the more complex model (such as the cooperative Hill model), we should prefer the simpler one.

Another way of looking at the problem is to keep in mind here that we only have nine data points for myoglobin. A two parameter model therefore has seven degrees of freedom, while a three parameter model has six. This is not an insignificant change, and there is a real possibility that the Hill model represents an over-fit of the limited available data.

While choice between models if often a judgment call that should integrate all available scientific information, there are tools that help us in the decision. We will consider two, the F-test and an interesting, non-statistical approach called AIC.

The F-test for Model Comparison

Using the F-test to compare two models follows the classical framework for statistical testing. You state a null hypothesis, assume it is true, and then compute a p-value that gives the probability of observing your data (or something more extreme) under that assumption. If the probability is low enough, you reject the null hypothesis.

We know that the more complex model will always fit better because we have more parameters. If we start with the more complex model and remove a parameter, we expect that the SSQ will go up. In fact, we can quantify this expected change in SSQ: if the simpler model is the correct one, then we expect that the relative change in the SSQ should be about equal to the relative change in the degrees of freedom. If the complex model is correct, we expect the SSQ to change more than this amount.

The p-value computed by the F-test answers the question: assuming that the simpler model is the correct one, what is the probability that we see a change in SSQ at least large as the one we observed when we simplify the complex model. If this p-value is low, then we reject the null hypothesis and accept the more complex model.

From a purely statistical point of view, if the p-value is below our pre-determined cutoff, we could not make any conclusion. However, since either model is considered a viable candidate for explaining our data, we apply the principle of maximum parsimony, and accept the less complex model.

The F-test is intimately related with concepts from ANOVA. In fact, to perform an F-test for model comparison in R, simple use the `anova` function, passing it two models as parameters. We begin by comparing the classic Michaelis-Menten model with the Hill model for our myoglobin data.

```
> anova(m0.myoglobin, m1.myoglobin)
Analysis of Variance Table

Model 1: v ~ Vmax * s/(Km + s)
Model 2: v ~ Vmax * s^n/(Km^n + s^n)
  Res.Df Res.Sum Sq Df   Sum Sq F value Pr(>F)
1      7   0.108325
2      6   0.094117  1 0.014207  0.9057  0.378
```

Note that in this case, the SSQ changed to 0.108 from 0.094, an increase of about 15%. The degrees of freedom changed to 7 from 6, an increase of about 17%. The F-value is the ratio of these changes; since it is close to one, we don't expect these changes to be inconsistent with the null hypothesis.

The magic (or, if you prefer, the mathematics) of the F-test is that it can convert this F-value into a p-value which tells us how surprised we are to see such an F-value assuming that the simpler model is correct. The reported p-value, 0.38, is not less than our standard cutoff of 0.05; we are not surprised. Therefore, we have no reason to reject the simpler Michaelis-Menten model. Invoking the principle of maximum parsimony, we therefore accept this simpler model as the better explanation for this data.

We now apply this test to the hemoglobin models:

```
> anova(m0.hemoglobin, m1.hemoglobin)
Analysis of Variance Table

Model 1: v ~ Vmax * s/(Km + s)
Model 2: v ~ Vmax * s^n/(Km^n + s^n)
  Res.Df Res.Sum Sq Df  Sum Sq F value    Pr(>F)
1      8    25.5516
2      7     1.6002  1 23.9515  104.78 1.834e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Here, we see that the p-value is well below our pre-established cutoff. If the simpler model were correct, we’d be quite surprised to see as large a change in SSQ as we did. Note that SSQ went from 1.6 to 25.5, a nearly 1,500% increase, while the degrees of freedom changed from 7 to 8, a roughly 14% increase.

It is very, very important to observe that the F-Test is only applicable for nested models, and only when you are fitting them to the exact same data. You can’t compare unrelated models with it (e.g., the power law and the asymptotic exponential models we investigated in the previous session). And you can’t compare a transformed and non-transformed model with it (the data are not the same).

Using AIC to Compare Models

The derivation for Akaike’s Information Criteria (AIC) is well beyond the scope of this course. It involves information theory, maximum likelihood theory, and entropy. We can get a rough feel for what the method is doing by looking at the resultant formula.

$$AIC = N \cdot \ln\left(\frac{\mathrm{SSQ}}{\mathrm{N}}\right) + 2(\mathrm{P} + 1)$$

Here, N is the number of observations, and P is the number of model parameters in the regression. We observe that the larger the SSQ, the higher the AIC will be. Also, the AIC increases as we add parameters to the model. Therefore, we can conclude that lower AICs are better. We can imagine that if we add a model parameter (increment P), the SSQ will go down. If the parameter was worth adding, the increase in the second term will be more than offset by a decrease in the first term.

By itself, the AIC is meaningless. The astute observer will realize that the SSQ has units of measure, and therefore there is an implicit standardization. We can therefore make the numerical value of the AIC whatever we like by altering the units of SSQ (or the standard value).

This ostensible shortcoming is overcome, however, when we look at the difference between the AICs of two models:

$$\Delta AIC = N \cdot ln\left(\frac{SSQ_B}{SSQ_A}\right) + 2(P_B - P_A)$$

The problem of the units of SSQ goes away. In practice, however, we can compute our AICs using consistent units, and select the model with the lower value.

A correction to AIC is necessary when N is not much greater than P. The corrected AIC equation is:

$$AIC_C = AIC + 2\frac{(P+1)(P+2)}{N-P}$$

We can use the AIC to compare any two models fitted to the same dataset. However, the models do not need to be nested; this makes use of AICs a very powerful technique for comparing unrelated models.

R can compute AICs for us; unfortunately, it does not apply the above correction, so we need to do that ourselves. Applying this methodology to our myoglobin models again confirms that Michaelis-Menten kinetics is the preferred description.

```
> n<-length(myoglobin$s)
> p<-length(coefficients(m0.myoglobin)); AICC.m0.myoglobin <- AIC(m0.myoglobin) + 2*(p+1)*(p+2)/(n-p)
> p<-length(coefficients(m1.myoglobin)); AICC.m1.myoglobin <- AIC(m1.myoglobin) + 2*(p+1)*(p+2)/(n-p)
> c(michaelis.menten=AICC.m0.myoglobin, hill=AICC.m1.myoglobin)
michaelis.menten             hill 
      -4.8091556       -0.8363698
```

Similarly, we see that the Hill model is preferred for the hemoglobin data.

```
> n<-length(hemoglobin$s)
> p<-length(coefficients(m0.hemoglobin)); AICC.m0.hemoglobin <- AIC(m0.hemoglobin) + 2*(p+1)*(p+2)/(n-p)
> p<-length(coefficients(m1.hemoglobin)); AICC.m1.hemoglobin <- AIC(m1.hemoglobin) + 2*(p+1)*(p+2)/(n-p)
> c(michaelis.menten=AICC.m0.hemoglobin, hill=AICC.m1.hemoglobin)
michaelis.menten             hill 
        46.75993         23.76821
```

You are much more likely to see F-tests in the literature. Because these tests are based on the classical statistical framework, many people feel more comfortable with them. However, comparison of AICs can be more powerful, especially when dealing with non-nested models.

It turns out that the difference in AIC (or AIC_C) is related to the probability that one model is correct relative to another. A difference of 6 corresponds to a 95% chance that the lower scoring model is correct. Therefore, if a more complex model has a lower score than a simpler model, but the difference is less than 6, you may still want to stick with the simpler model, because the evidence in favor of the complex one is not overwhelming. Given two non-nested models (perhaps with the same number of parameters), you might simply choose the one with the lower AIC score, but appreciate that the difference between the models is not ‘significant’. We were warned that some of this is more art than science.

Confidence Intervals with the F-Test

An interesting application of the idea behind the F-test is that it can be used as an alternative means of estimating the uncertainty in model parameters. The basic idea is to use what we learned about F-tests to compare a model with zero parameters to our best fit model.

Consider the Michaelis-Menten myoglobin model. The SSQ for this model is 0.108325. We had N=9 data points and P=2 parameters, so we had DF=7 degrees of freedom. Take this as model A.

Now consider a hypothetical model with no parameters; we would have N=9, P=0, and DF=9. Take this as model B.

When we compare these two models, we compute the F statistic in the usual way.

$$F = \frac{\left(\frac{SSQ_B - SSQ_A}{SSQ_A}\right)}{\left(\frac{DF_B - DF_A}{DF_A}\right)}$$

It turns out that we can compute the critical F-value for a 95% CI using R. This comes from the F-distribution, and is computed in R using the command: `qf(0.95, P, N-P)`. For our case F_{crit}=4.737. This means that if the F-value computed for a comparison between models A and B is less than 4.737, we do not consider them to be significantly different.

Setting F to F_{crit} in the above equation allows us to compute SSQ_{crit}. For the myoglobin data, SSQ_{crit}=0.2549351. This means that any model (with all parameters fixed) where SSQ<0.2549351 is considered not significantly different from the best-fit model we have. Given any choice of parameters, we can compute SSQ and compare to this value.

We can do this systematically:

```
> ssq.crit <- 0.2549351
> ssq.myoglobin <- function(Vmax, Km) {
  sum((Vmax * myoglobin$s / (Km + myoglobin$s) - myoglobin$v)^2)
}
> Vmax <- data.frame(Vmax=seq(2,10,0.02))
> Km <- data.frame(Km=seq(1,5,0.02))
> cases <- merge(Vmax,Km)
> head(cases)         no pun intended here
  Vmax Km
1 2.00  1
2 2.01  1
3 2.02  1
4 2.03  1
5 2.04  1
```

```
6 2.05  1
> cases$ssq <- mapply(ssq.myoglobin, cases$Vmax, cases$Km)
> plot(Vmax~Km, data=subset(cases, ssq<ssq.crit), pch=".")
```

The resultant plot gives an envelope of values that are consistent with the best-fit model. Compare this to the asymptotic CIs reported by `confint`:

```
> coefficients(m0.myoglobin)
    Vmax       Km
5.117063 2.828153
> confint(m0.myoglobin)
Waiting for profiling to be done...
         2.5%    97.5%
Vmax 4.747424 5.535149
Km   2.296327 3.476164
> lines(rep(coefficients(m0.myoglobin)[["Km"]],2),
confint(m0.myoglobin, 'Vmax'), col="blue", lwd=3)
> lines(confint(m0.myoglobin, 'Km'),
rep(coefficients(m0.myoglobin)[["Vmax"]],2), col="blue", lwd=3)
```

If you think about it, this plot makes sense. The combination of parameters V_{max}/K_m is the initial slope of the Michaelis-Menten curve. Based on the data we have (look back at the original plot), we have a pretty good idea of what that should be. However, determining the maximum of the curve is quite difficult from our data (this is notoriously difficult experimentally; you need to go to very high values of [S]). To get K_m, which is the half-maximal concentration, we need a decent idea of V_{max}. So while there is significant uncertainty in both K_m and V_{max}, we do expect that they have a relationship.