meat0.r: Explanation of code

Goals of code:

- Fitting a regression line
- Predicting mean Y at new values of X

This week's material on linear regression is an introduction. Next week we'll see a lot more.

The code file starts by reading the meat.txt file and creating a new variable called logtime.

Fitting a regression line: `lm()`
The lm() function is the workhorse R model fitting function. When the X variable (right hand side of the ~ **is not** a factor, lm() fits a regression line.

The command `meat.lm <- lm(ph~logtime, data=meat)` will fit a regression model to predict y = ph from x = logtime, where both variables are in the data frame called meat. The result is stored in the meat.lm object.

If you print the object (meat.lm), you see what was done to create meat.lm and the estimated coefficients. Intercept is the intercept (β_0) and logtime is the regression slope (β_1). Slope coefficients are always labeled by the name of the associated X variable. That will be very useful when we have more than one X variable in a model.

If you use summary() on the output object, you get a lot more information. I don't find the first block of results (a five number summary of the residuals) very useful. The second block of results gives you the estimated coefficients, their se's, T values for the test of parameter = 0 and the associate p-value for each parameter. Again, the slope is labeled by the name of the associated X variable.

The residual se is the pooled sd ($\sqrt{\text{MSE}}$). The rest of the numbers can be ignored or will be discussed later.

The predict() function extracts predicted values for X values in the data set used to fit the regression. It also can provide predicted values for new values of X.

Estimating mean Y for a specified X: `predict() with a newdata= argument`
The predict() function can provide predicted values and associated information for X values not in the data set. This is done by providing a new data frame with the desired X values. That new data frame is specified by the `newdata=` argument. This must be a data frame (you can't just say: newdata=log(5). Sorry. And, the name in the data frame must match the variable name in the model.

The data.frame() function creates a data frame from a set of values. In this case, we want to predict for time = 5 hours, so logtime is log(5).

This data frame **must** include a column with the same variable name as that in the data frame used to fit the model. The `logtime=` piece specifies the variable name. The `log(5)` specifies the value. That data frame is printed, then used as the value of the newdata= argument. The result of predict() is the predicted value. As always, this can be stored in a variable or in a data frame if you want to save the result.

Specifying many X values to predict at: data.frame()
You can extend what you just did to many X values. In all cases, you need a data frame with a named vector specifying the desired values. The code in meat0.r illustrates one way to produce the desired data frame. We'll see two other ways next week.

`c(1,1.5,2,2.5,3,3.5,4,4.5,5)`:
The c() function concatenates (hence the c) values to make a vector. c() can be used anywhere you need to create or specify a vector of values. The values are separated by commas.