Quantitative Understanding in Biology Module II: Model Parameter Estimation Lecture I: Linear Correlation and Regression

Correlation

Linear correlation and linear regression are often confused, mostly because some bits of the math are similar. However, they are fundamentally different techniques. We'll begin this section of the course with a brief look at assessment of linear correlation, and then spend a good deal of time on linear and non-linear regression.

If you have a set of pairs of values (call them x and y for the purposes of this discussion), you may ask if they are correlated. Let's spend a moment clarifying what this actually means. First, the values must come in pairs (e.g., from a paired study). It makes no sense to ask about correlation between two univariate distributions.

Also, the two variables must both be observations or outcomes for the correlation question to make sense. The underlying statistical model for correlation assumes that both x and y are normally distributed; if you have systematically varied x and have corresponding values for y, you cannot ask the correlation question (you can, however, perform a regression analysis). Another way of thinking about this is that in a correlation model, there isn't an independent and a dependent variable; both are equal and treated symmetrically. If you don't feel comfortable swapping x and y, you probably shouldn't be doing a correlation analysis.

The standard method for ascertaining correlation is to compute the so-called Pearson correlation coefficient. This method assumes a linear correlation between x and y. You could have very well correlated data, but if the relationship is not linear the Pearson method will underestimate the degree of correlation, often significantly. Therefore, it is always a good idea to plot your data first. If you see a non-linear but monotonic relationship between x and y you may want to use the Spearman correlation; this is a non-parametric method. Another option would be to transform your data so that the relationship becomes linear.

In the Pearson method, the key quantity that is computed is the correlation coefficient, usually written as r. The formula for r is:

$$r = \frac{1}{N}\sum\left[\frac{(x_i - \bar{x})}{SD_x} \cdot \frac{(y_i - \bar{y})}{SD_y}\right]$$

The correlation coefficient ranges from -1 to 1. A value of zero means that there is no correlation between x and y. A value of 1 means there is perfect correlation between them: when x goes up, y goes up in a perfectly linear fashion. A value of -1 is a perfect anti-correlation: when x goes up, y goes down in an exactly linear manner.

Note that x and y can be of different units of measure. In the formula, each value is standardized by subtracting the average and dividing by the SD. This means that we are looking at how far each value is from the mean in units of SDs. You can get a rough feeling for why this equation works. Whenever both x and y are above or below their means, you get a positive contribution to r; when one is above and one is below you get a negative contribution. If the data are uncorrelated, these effects will tend to cancel each other out and the overall r will tend toward zero.

A frequently reported quantity is r^2. For a linear correlation, this quantity can be shown to be the fraction of the variance of one variable that is due to the other variable (the relationship is symmetric). If you compute a Spearman correlation (which is based on ranks), r^2 does not have this interpretation. Note that for correlation, we do not compute or plot a 'best fit line'; that is regression!

Many people take their data, compute r^2, and, if it is far from zero, report that a correlation is found, and are happy. This is a somewhat naïve approach. Now that we have a framework for statistical thinking, we should be asking ourselves if there is a way to ascertain the statistical significance of our computed r or r^2. In fact there is; we can formulate a null hypothesis that there is no correlation in the underlying distributions (they are completely independent), and then compute the probability of observing an r value as large or larger in magnitude as the one we actually observed just be chance. This p-value will be a function of the number of pairs of observations we have as well as of the values themselves. Similarly, we can compute a CI for r. If the p-value is less than your pre-established cutoff (or equivalently, if your CI does not include zero), then you may conclude that there is a statistically significant correlation between your two sets of observations.

The relevant function in R to test correlation is `cor.test`. You can use the `method` argument to specify that you want a Pearson or a Spearman correlation.

You can also compute a CI for r^2. Just square the lower and upper limits of the CI for r (but take due account of intervals that include zero); note that the CI for r^2 will generally be non-symmetric. In many cases, you may see weak r^2s reported in the literature, but no p-value or CI. If you wish, you can compute a p-value yourself just by knowing N (the number of pairs) and r; see a text if you need to do this.

Introduction to Modeling

Regression, or curve fitting, is a much richer framework than correlation. There are several reasons why we may want to perform a regression analysis:

1) Artistic: we want to present our data with a smooth curve passing near the points. We don't have any quantitative concerns; we just want the figure to "look good".

2) Predictive: We want to use the data we've collected to predict new values for an outcome given measured values for an explanatory variable. This may be, for example, a standardization curve for an instrument or assay. We'd like our predictions to be as consistent with our data as possible, and we don't care too much about the math that generates our predicted values (although simpler equations would be preferred for practical purposes).
3) Modeling: We want to test a hypothesized mechanistic model of a system against data, or we wish to use a mechanistic model we believe in to predict new data. In this case, we do care about the mathematical structure of our model, as it is derived from (or informs) our mechanistic model.

As basic scientists trying to figure out how the world works, we will focus on the third technique throughout most of this course.

In accordance with Occam's Razor, all else being equal, we prefer simpler models to more complex ones. In mathematical terms, we prefer:

- Fewer explanatory variables
- Fewer parameters (model coefficients, etc.)
- Linear over non-linear relationships
- Monotonic vs. non-monotonic relationships
- Fewer interactions among explanatory variables

Of course, there is always a natural tension, because more complex models tend to fit our data better. There is always a tradeoff between model complexity and accuracy, and scientific judgment is often necessary to resolve this inherent conflict. As we shall see, there are some objective techniques that can help.

Simple Linear Regression

We'll start with the mechanics of a simple linear regression; you have probably done this before. Say we have our pairs of values, and we wish to fit a line to them. The mechanics of this process in R are as follows:

```
> x <- seq(0,10,0.1)
> y <- 2 * x + 3 + rnorm(x)
> plot(y ~ x)
> mymodel <- lm(y ~ x)
> abline(mymodel, lwd = 2)
> summary(mymodel)

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-2.4042 -0.6696  0.1068  0.5935  2.6076
```



```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.80888    0.19743   14.23   <2e-16 ***
x            2.02950    0.03411   59.50   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.9995 on 99 degrees of freedom
Multiple R-Squared: 0.9728,     Adjusted R-squared: 0.9725
F-statistic:  3540 on 1 and 99 DF,  p-value: < 2.2e-16
```

We begin by generating a sequence of x values, and then corresponding y values. The underlying relationship that we have simulated is simple: y = 2x + 3; but there is also a random component. Once we generate this data, we pretend we don't know where it came from.

We begin our modeling by plotting the data. In this case, simple inspection makes it clear that a linear model would be appropriate. Note the weird syntax for the plot command. This is an example of a "model formula" in R; you should read it as "y as a function of x".

We then ask R to derive a linear model for the relationship between y and x. Again, we use a model formula. The object that is returned is quite complex, and there are many things that we can do with it. The important thing here is to squirrel the returned object away in a variable that we can refer to later.

One of the best ways to assess if our model describes our data well is to plot the regression relationship. The `abline` function does this, taking the model object as its only argument. We can see from the updated plot that the fit is pretty good.

Our example concludes by generating a summary of the linear model. We see that the best-fit line has a slope of 2.03 and an intercept of 2.81. Furthermore, we get an r^2 of 0.9728, and a very low p-value.

It turns out that if we asked for a correlation between x and y, we'd get the same r^2 and p-value (`cor.test` reports r, so you'd need to square it yourself).

```
> cor.test(x,y)$estimate^2
     cor
0.972794
```

This is one of the reasons why correlation and regression are often confused. Since you get the same result for r^2, people often confuse them. For regression, life is not as simple as just looking at r^2.

As you can see, there is a great deal of additional information in the linear model (and this is just a summary). We'll spend a fair amount of time going through some of these results and how to use them.

When you perform a basic regression (linear or otherwise), the model parameters are chosen to minimize the sum of the squares of the residuals. A residual is the difference between a predicted and

an observed value. This breaks the symmetry in the mathematics between x and y. You will get different lines if you regress (using R's notation) y ~ x and x ~ y.

We have used the term linear a few times without formally defining what that means. We've taken for granted here that a linear regression fits data to a model of the form y = mx + b, and for present purposes, this is what we'll take to mean "linear". However, you should be aware that in other contexts, the word "linear" implies something slightly different; for instance, linear algebra is the study of systems that follow the form **y** = **A**·**x**; this is a matrix equation whose one-dimensional analog is y = ax; there is no intercept term. A transformation that includes such an intercept is called an "affine" transformation. Converting inches to centimeters is a linear transformation, whereas converting temperatures from Fahrenheit to Kelvin is an affine transformation. We'll have a bit more to say about this when we cover dynamical systems.

Model Formulas and More Complex Linear Regression

There is more (or sometimes less) to linear models that just y=mx+b. For now, we'll go through the mechanics of fitting models with R. We'll leave the very important topic of interpretation of the results for another time.

Here is another dataset that we can fit some sample models to:

```
> x <- 0:100
> y <- x + (1/400)*x^2 + rnorm(x, sd=3)
> plot(y ~ x)
> model1 <- lm(y ~ x)
> abline(model1, lwd = 2)
> summary(model1)

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-9.1970 -2.4664  0.4569  2.4463  7.9105

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -3.89586    0.65919   -5.91 4.86e-08 ***
x            1.24341    0.01139  109.18  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.337 on 99 degrees of freedom
Multiple R-Squared: 0.9918,     Adjusted R-squared: 0.9917
F-statistic: 1.192e+04 on 1 and 99 DF,  p-value: < 2.2e-16
```


The model is "mostly linear" in the region we are inspecting, but it has a slight upward curve due to the quadratic term. This is well masked by a fair amount of noise.

It is important to note that some non-linear models are amenable to linear regression. What is required by linear regression is that the residuals are linear functions of the model parameters we are fitting. This is a very different property than the model output being a linear function of the model inputs. For example, if we are perceptive (or lucky), we might try adding a quadratic term to our model:

```
> model2 <- lm(y ~ x + I(x^2))
```

Although y is a non-linear function of x, the residuals are linear functions of the three parameters that are implicit in this model formula: the coefficient of x, the coefficient of x^2, and the implicit constant term.

```
> lines(x, model2$fitted.values, col="red", lwd=2)
> summary(model2)

Call:
lm(formula = y ~ x + I(x^2))

Residuals:
     Min       1Q   Median       3Q      Max
-7.30989 -1.94178  0.08589  1.97688  7.73413

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.2153341  0.8425434  -0.256    0.799
x            1.0203441  0.0389400  26.203  < 2e-16 ***
I(x^2)       0.0022306  0.0003768   5.920 4.76e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.879 on 98 degrees of freedom
Multiple R-Squared: 0.9939,     Adjusted R-squared: 0.9938
F-statistic:  8027 on 2 and 98 DF,  p-value: < 2.2e-16
```

Here we extract the values that the model predicts and use them to plot a curve representing the second model in red.

In our model formula, we used the `I` function. Model formulas do some weird things (as we will see in a moment). The `I` function tells R not do anything special with this term of the model, and just treat it as you would normally expect.

As astute observer might notice that in our revised model we now have an intercept term that is pretty close to zero. In the spirit of Occam's Razor, we might try another model that does not include this term (there are other indications that this is a good idea which we will cover later).

```
> model3 <- lm(y ~ x + I(x^2) - 1)
```



```
> lines(x, model3$fitted.values, col="blue", lty=4)
> summary(model3)

Call:
lm(formula = y ~ x + I(x^2) - 1)

Residuals:
     Min       1Q   Median       3Q      Max
-7.27759 -1.94202 -0.05807  1.94395  7.74185

Coefficients:
        Estimate Std. Error t value Pr(>|t|)
x      1.0117740  0.0197024  51.353  < 2e-16 ***
I(x^2) 0.0023017  0.0002531   9.094 1.03e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.865 on 99 degrees of freedom
Multiple R-Squared: 0.9983,     Adjusted R-squared: 0.9983
F-statistic: 2.9e+04 on 2 and 99 DF,  p-value: < 2.2e-16
```

The model formula here should be interpreted as follows: “y as a function of x, a literal expression computed as x^2, and not including an intercept”.

In this case, the two parameter quadratic model predicts values that are nearly identical to those predicted by the three parameter model. Since this is a simpler model that is just as good as the previous one, we may settle on this one as a good representation of our data.

	Model formula	R-Squared:	SSQ[1]
model1	y ~ x	0.9918	1482
model2	y ~ x + I(x^2)	0.9939	910
model3	y ~ x + I(x^2) - 1	0.9983	917

Looking at the values for R^2, we see that there is something ostensibly amiss. `model3` is a special case of `model2` where the intercept is taken to be zero. Therefore, it is impossible for `model3` to have a better fit to the data than `model2`. It turns out that the definition of R^2 changes when the model does not include an intercept. The moral of the story is that R^2 is not a good judge of how well the model matches the data. While it is used frequently in the case of straight-line (y=mx+b) models, it is not a great indicator of what has been achieved when more complex models are used. You can use R^2 to compare how well different datasets match a given model. But you should not use R^2 to compare how well two different models match a given dataset.

A better criterion for comparing how well different models match the same dataset is to look at the sum of the squares of the residuals. This is, after all, the objective function that is used in the implicit

[1] Use `sum((model1$residuals)^2)` to obtain the SSQ for `model1`.

minimization that linear regression performs. In the above table, we see that both quadratic models perform significantly better than the linear model. Furthermore, we see that the loss in model fit due to the removal of the intercept term is quite small, which may help justify the third model as better even though the fit it not quite as good. We'll have more to say about this in upcoming sessions.

We can also do linear regression against models with multiple explanatory variables. Typically we work with data frames to do this

```
> x <- 0:10
> y <- 0:10
> mydata <- merge(x,y)
> mydata$z <- 2 * mydata$x + 3 * mydata$y + mydata$x * mydata$y +
rnorm(mydata$x)
model1 <- lm(z ~ x + y, data=mydata)
model2 <- lm(z ~ x * y, data=mydata)
model3 <- lm(z ~ x * y - 1, data=mydata)
```

The first model fits z as a function of x and y; it will have three parameters: the coefficient of x, the coefficient of y, and a constant term. The second model will include a fourth additional parameter: the coefficient of the interaction term (the coefficient of a term for x·y). The third model is a three term model; it includes an x term, a y term, and an interaction term, but not a constant. Which model do you expect to best fit the data (i.e., have the lowest SSQ)? Which model do you think will best explain the data (i.e., which model is the best representation of where the data comes from)?

R is capable of dealing with more than two explanatory variables; a model formula such as z ~ w * x * y is acceptable (this will include all binary interaction between the three explanatory variables, but not a ternary product). As you might expect, visualization gets a bit tricky when you have multiple explanatory variables.

