Workshop 4: Linear models

In this workshop we will use R to fit linear models to data. It would be useful to read through the lecture notes ahead of time to review basic principles of linear model fitting and implementation in R.
Linear models for fixed effects are implemented in the R command `lm`. This method is not suitable for models that contain random effects.

Prediction using simple linear regression

We'll start with linear regression because you are probably most familiar with this type of linear model. The data are from Whitman et al (2004 *Nature* 428: 175-178), who noticed that the amount of black pigmentation on the noses of male lions increases as they get older. They used data on the proportion of black on the noses of 32 male lions of known age in Tanzania. We will fit a linear model to these data to predict a lion's age from the proportion of black in his nose. The data can be downloaded here.

Read and examine the data

1. Read the data into R.
2. View the first few lines of data to make sure it was read correctly.
3. Use the `plot` (or `geom_point()`) command to create a scatter plot of the data. Choose the response and explanatory variables with care: we want to predict age from the proportion black in the nose.

Fit a linear model

1. Fit a linear model to the lion data. Store the output in an `lm` object. Choose the response and explanatory variables with care: we want to predict age from the proportion black in the nose. Fill in the missing bits below

```
out <- lm(--- ~ ---, data =  lion)
```

2. Add the best-fit line to the scatter plot. If your model output is stored in an object `out`, you can access the output from the model, in matrix form, using `coef(summary(out))`. You can then extract the intercept and slope with

```
int <- coef(summary(out))['(Intercept)','Estimate']
bla <- coef(summary(out))['black',      'Estimate']
```

Now you should be able to use the `abline` (or `geom_abline()`) command to add a line with this intercept and slope to the plot. Does the relationship appear linear? Check for any serious problems such as outliers or changes in the variance of residuals.

3. Using the `summary` command, obtain the estimate for R^2 (a measure of model fit).

4. Calculate 95% confidence intervals for the slope and intercept using the the `confint` function.

5. Summarize the fit of the model to the data via an F-test, using the `drop1` or `anova` commands.

6. Apply the `plot` command to the `lm` object created in (1) to diagnose violations of assumptions (keep hitting `<return>` in the command window to see all the plots). Recall the assumptions of linear models. Do the data conform to the assumptions? Are there any potential concerns? Most of the plots will be self-explanatory, except perhaps the last one. "Leverage" calculates the influence that each data point has on the estimated parameters. For example if the slope changes a great deal when a point is removed, that point is said to have high leverage. "Cook's distance" measures the effect of each data point on the predicted values for all the other data points. A value greater than 1 is said to be worrisome. Points with high leverage don't necessarily have high Cook's distance, and vice versa.

7. Optional: One of the data points (the oldest lion) has rather high leverage. To see the effect this has on the results, refit the data leaving this point out. Did this change the regression line substantially?

Prediction

1. Display the data once again in a scatter plot with the regression line.

2. Use the `predict` function to add confidence bands to the scatter plot. These are confidence limits for the prediction of mean of lion age at each value of the explanatory variable. You can think of these as putting bounds on the most plausible values for the "true" or population regression line. Note the spread between the upper and lower limits and how this changes across the range of values for age. To do this, you will need to first create a data-frame containing the set of values for which you'd like to generate confidence intervals. E.g., the following:

```
new.vals <- data.frame(black=seq(min(ll$black),max(ll$black),
                       length=20))
```

will create a data-frame with 20 values that span the range of values present in the data. Then you can use the `predict` function to add confidence (or prediction) intervals for these new values. You'll have to pass in your model output, the new data-frame, and also specify whether you want confidence intervals or prediction intervals. See the help for `predict` for more details.

3. Use the `predict` command to add prediction intervals to the scatter plot. These are confidence limits for the prediction of new individual lion ages at each value of the explanatory variable. Whereas confidence bands address questions like "what is the mean age of lions whose proportion black in the nose is 0.5?", prediction intervals address questions like "what is the age of that new lion over there, which has a proportion black in the nose of 0.5 ?".

4. Examine the confidence bands and prediction intervals. Is the prediction of mean lion age from black in the nose relatively precise? Is prediction of individual lion age relatively precise? Could this relationship be used in the field to age lions?

Effects of light treatment on circadian rhythms

Our second example fits a linear model with a categorical explanatory variable. The data are from an experiment by Wright and Czeisler (2002. *Science* 297: 571) that re-examined a previous claim that light behind the knees could reset the circadian rhythm of an individual the same way as light to the eyes. One of three light treatments was randomly assigned to 22 subjects (a three-hour episode of bright lights to the eyes, to the knees, or to neither). Effects were measured two days later as the magnitude of phase shift in each subject's daily cycle of melatonin production. A negative measurement indicates a delay in melatonin production, which is the predicted effect of light treatment. The data can be downloaded here.

Read and examine the data

1. Read the data from the file.

2. View the first few lines of data to make sure it was read correctly.

3. Determine whether the categorical variable "treatment" is a factor. If not a factor, convert treatment to a factor using the factor command. This will be convenient when we fit the linear model.

4. Use the levels command on the factor variable "treatment" to see how R has ordered the different treatment groups. The order will be alphabetical, by default. Conveniently, you will find that the control group is listed first in the alphabetical sequence. As you are about to analyze these data with a linear model in R, can you think of why having the control group first in the order is convenient?

5. To get practice, change the order of the levels so that the "knee" treatment group is second in the order, after "control", and the "eyes" group is listed third.

6. Plot the phase shift data, showing the individual data points in each treatment group.

Fit a linear model

1. Fit a linear model to the light treatment data. Store the output in an `lm` object.

2. Create a graphic that illustrates the fit of the model to the data. In other words, include the predicted (fitted) values to your plot. Try plotting the values yourself manually. Then try using the `visreg` function from the `visreg` package. Remember to install the package using `install.packages()`.

```
library(visreg)
visreg(out, xvar='treatment')
```

   where `out` is your model output.

3. Use the `plot` command to check whether the assumptions of linear models are met in this case. Examine the plots. Are there any potential concerns? There are several options available to you if the assumptions are not met (transformations, robust regression methods, etc.) but we don't seem to need them in this case.

4. Remember from lecture that R represents the different levels of the categorical variable using dummy variables. To peek at this behind-the-scenes representation, use the `model.matrix` command on the model object from your linear model fit in step (1). The output should have a column of 1's for the intercept and two additional columns representing two of the three levels of the explanatory variable. Why is one level left out? Which level is the one not represented by a dummy variable?

5. Using the `lm` model object, obtain the parameter estimates (coefficients) along with standard errors. Examine the parameter estimates. If you've done the analysis correctly, you should see the three coefficients. Rounded, they are -0.309, -0.027, and -1.24. What do each of these coefficients represent? Note that the output will also include an R^2 value. This is loosely interpretable as the "percent of the variance in phase shift that is explained by treatment."

6. Obtain 95% confidence intervals for the three parameters using `emmeans`. Why are the SE's and confidence limits in the `emmeans` table not the same as if you'd calculated them by hand separately for each treatment group? To use `emmeans` you will need to install and load the package `emmeans`.

7. Test the effect of light treatment on phase shift using `drop1`.

Fly sex and longevity revisited

We analyzed these data previously in our graphics workshop. Here we will analyze them further by fitting a linear model to the data.
The data are from L. Partridge and M. Farquhar (1981), Sexual activity and the lifespan of male fruit flies, *Nature* 294: 580-581. The experiment placed male fruit flies with varying numbers of previously-mated or virgin females to investigate whether mating activity affects male lifespan. To begin, download the file here.
The linear model will have longevity as the response variable, and two explanatory variables: treatment (categorical) and thorax length (numerical; representing body size). The goal will be to compare differences in fly longevity among treatment groups, correcting for

differences in thorax length. Correcting for thorax length will possibly improve the estimates of treatment effect. The method is also known as analysis of covariance, or ANCOVA.

Read and examine the data

1. Read the data from the file.

2. View the first few lines of data to make sure it was read correctly.

3. Determine whether the categorical variable "treatment" is a factor. If not a factor, convert treatment to a factor. This will be convenient when we fit the linear model.

4. Use the "levels" command on the factor variable "treatment" to see how R has ordered the different treatment groups (should be alphabetically).

5. Change the order of the categories so that a sensible control group is first in the order of categories. Arrange the order of the remaining categories as you see fit.

6. Optional: This repeats an exercise from the graphics workshop. Create a scatter plot, with longevity as the response variable and body size (thorax length) as the explanatory variable. Use a single plot with different symbols (and colors too, if you like) for different treatment groups. Or make a multipanel plot using `layout` or the lattice or ggplot2 packages, if you know how to use those.

Fit a linear model

1. Fit a linear model to the fly data, including both body size (thorax length) and treatment as explanatory variables. Place thorax length before treatment in the model formula. **Leave out the interaction term for now (use + instead of * to specify the model).** We'll start off by assuming that there is no interaction between the explanatory variables thorax and treatment.

2. Use the `plot` command to check whether the assumptions of linear models are met in this case. Are there any potential concerns? If you have done the analysis correctly, you will see that the variance of the residuals is not constant, but increases with increasing fitted values. This violates the linear model assumption of equal variance of residuals.

3. Attempt to fix the problem identified in step (3) using a log-transformation of the response variable. Refit the model and reapply the graphical diagnostic tools to check assumptions. Any improvements? (To my eye the situation is improved but the issue has not gone away entirely.) Let's continue anyway with the log-transformed analysis.

4. Visualize the fit of the model to the data using the visreg package. Try two different possibilities. In the first, plot the fit of the response variable to thorax length separately for each treatment group. In the second, plot the fit of the data to treatment, conditioning on the value of the covariate (thorax length).

5. Obtain the parameter estimates and standard errors for the fitted model. Interpret the parameter estimates. What do they represent? Which treatment group differs most from the control group?

6. Obtain 95% confidence intervals for the thorax and treatment parameters.

7. Test overall treatment effects with `drop1`. Interpret each significance test - what exactly is being tested?

8. Our analysis so far has assumed that the regression slopes for different treatment groups are the same. Is this a valid assumption? We have the opportunity to investigate just how different the estimated slopes really are. To do this, fit a new linear model to the data, but this time include an interaction term between the explanatory variables (use * instead of +).

9. The parameters will be more complicated to interpret in the model including an interaction term, so lets skip this step. Instead, go right to `drop1` to test the interaction term using the new model fit. Interpret the result. Does it mean that the interaction term really is zero?

10. Another way to help assess whether the assumption of no interaction is a sensible one for these data is to determine whether the fit of the model is "better" when an interaction term is present or not, and by how much. We will learn new methods later in the course to determine this, but in the meantime a simple measure of model fit can be obtained using the adjusted R^2 value. The ordinary R^2 measures the fraction of the total variation in the response variable that is "explained" by the explanatory variables. This, however, cannot be compared between models that differ in the number of parameters because fitting more parameters always results in a larger R^2, even if the added variables are just made-up random numbers. To compare the fit of models having different parameters, use the adjusted R^2 value instead, which takes account of the number of parameters being fitted. Use the summary command on each of two fitted models, one with and the other without an interaction term, and compare their adjusted R^2 values. Are they much different? If not, then maybe it is OK to assume that any interaction term is likely small and can be left out.