**BIOL 850 Workshop: Basic Statistical Analysis in R**

Goals for today:

* Conduct some basic statistical analyses with base R
* Get a taste of a wide array of analyses you can do in base R

Before doing actual statistical modelling and analysis, it is often useful to make some simple characterizations of your data in terms of summary statistics and graphics. We discussed making histograms, boxplots and other graphs last week. Visualization of your data is often an important first step. Summarizing your data and understanding it is also important.

Summary statistics are easy to calculate with R. For example, to calculate the mean, standard deviation, variance and median:

|  |
| --- |
| x <- rnorm(50)  mean(x)  # [1] -0.1092592  sd(x)  # [1] 0.9693084  var(x)  # [1] 0.9395587  median(x)  # [1] -0.112664 |

In this example, an artificial data vector x of 50 normally distributed observations was created. Will use this data for some of our examples. Note that when reproducing the examples, your results will vary, as your random numbers will differ.

Empirical quantiles can be obtained by the function *quantile*()

|  |
| --- |
| quantile (x)  ## 0% 25% 50% 75% 100%  ## -2.0510027 -0.7054569 -0.1126640 0.2817860 1.9637988 |

By default, you get the minimum, maximum and the three quartiles (0.25, 0.50 and 0.75)

What happens if there are missing values in our data?

Let’s see:

|  |
| --- |
| y <- c(11,-3,NA,56,5,NA,12) #here we create a vector with NA values  mean(y)  # [1] NA |

R will not skip missing values unless explicitly told to do so. The mean of a vector with an unknown value is unknown. However, you can give the na.rm argument (*not available, remove)* to request that missing values be removed

|  |
| --- |
| mean(y,na.rm=T)  # [1] 16.2 |

You can obtain a nice summary display of a numeric variable from the *summary* function:

|  |
| --- |
| summary(y)  ## Min. 1st Qu. Median Mean 3rd Qu. Max. NA's  ## -3.0 5.0 11.0 16.2 12.0 56.0 2 |

You can also summarize an entire data frame with summary

For example, lets read in this small dataset: sitedata.csv

|  |
| --- |
| sitedata <-read.csv(“sitedata.csv”)  summary(s(sitedata)  ## Site Temp pH rain\_mm  ## A : 1 Min. :12.20 Min. :6.400 Min. : 2.90  ## B : 1 1st Qu.:17.60 1st Qu.:6.725 1st Qu.:12.45  ## C : 1 Median :19.15 Median :7.300 Median :16.10  ## D : 1 Mean :18.71 Mean :7.225 Mean :15.76  ## E : 1 3rd Qu.:20.18 3rd Qu.:7.500 3rd Qu.:19.55  ## F : 1 Max. :24.10 Max. :8.400 Max. :26.40  ## (Other):16 NA's :2 NA's :3 |

### **t-tests**

A t-test can be done simply with the function **t.test()** and you control the specifics (paired, two-sided, etc.) with options. For the simple case of comparing the difference of two means we can use all of the defaults:

The two-sampled t-test is used to test the hypothesis that two samples may be assumed to come from distributions with the same mean

|  |
| --- |
| pop1 <- rnorm(30, mean=3, sd=2)  pop2 <- rnorm(30, mean=10, sd=5)  pop\_ttest <- t.test(pop1, pop2)  pop\_ttest  ## Welch Two Sample t-test  ##  ## data: pop1 and pop2  ## t = -9.6747, df = 43.062, p-value = 2.298e-12  ## alternative hypothesis: true difference in means is not equal to 0  ## 95 percent confidence interval:  ## -10.661301 -6.983417  ## sample estimates:  ## mean of x mean of y  ## 2.986902 11.809261 |

By default, t.test does **not** assume equal variances; instead of Student’s t-test, it uses the Welch t-test by default. To use Student’s t-test, set var.equal=TRUE.

**Two-sample Wilcoxon test**

For a t-test, the assumption is that the data are normally distributed. If you doubt this assumption, you may prefer a nonparametric test. The two-sampled Wilcoxon test is used by using the function  **wilcox.test()** which behaves similarly to  **t.test()**

|  |
| --- |
| pop\_wilcoxtest <- wilcox.test(pop1, pop2)  pop\_wilcoxtest  ## Wilcoxon rank sum test  ##  ## data: pop1 and pop2  ## W = 43, p-value = 6.334e-12  ## alternative hypothesis: true location shift is not equal to 0 |

### **Correlation**

Next let’s take a look at correlations, which uses the **cor()** function. Unlike mean, var, sd and similar one-vector functions, you can not use na.rm =T to handle missing variables. This is because there are more possibilities than just removing the incomplete cases or failing. If more than two variables are in play, it is also possible to use information from all nonmissing pairs of measurements.

Instead, you can use:

> cor(sitedata$Temp,sitedata$pH, use=”complete.obs”)

First is the use argument which allows you to use the entire dataset or select complete cases which is useful when you have NA values. There are several options. Also, the default correlation method is for Pearson’s. If you would like to use non-parametric correlations (e.g. rank), you specify that here. See ?cor for more information. cor.test already has a default way of omitting NAs using the na.action argument. See ?cor.test.

|  |
| --- |
| # a simple correlation  cor(sitedata$Temp,sitedata$pH, use="complete.obs") |

A data frame as input to cor returns a correlation matrix. However, we can’t just do cor(sitedata) because sitedata has non-numeric columns. we need to use dplyr first to select only the numeric columns otherwise you will have an error.

|  |
| --- |
| sitedata\_numeric <-select(sitedata, -Site)  cor(sitedata\_numeric, use = "complete.obs")  ## Temp pH rain\_mm  ## Temp 1.0000000 0.45639149 0.51455752  ## pH 0.4563915 1.00000000 -0.05735697  ## rain\_mm 0.5145575 -0.05735697 1.00000000 |

### **Linear Regression**

Next let’s take a look at linear regression. One of the common ways of fitting linear regressions is with lm().

|  |
| --- |
| lm(Temp ~ pH, data=sitedata)  ##  ## Call:  ## lm(formula = Temp ~ pH, data = sitedata)  ##  ## Coefficients:  ## (Intercept) pH  ## 1.612 2.374 |

This doesn’t provide much information, but if we save this as an object and then use summary:

|  |
| --- |
| lm\_sites <- lm(Temp ~ pH, data=sitedata)  summary(lm\_sites)  ## Call:  ## lm(formula = Temp ~ pH, data = sitedata)  ##  ## Residuals:  ## Min 1Q Median 3Q Max  ## -4.8438 -1.0612 -0.5682 1.5720 5.2936  ##  ## Coefficients:  ## Estimate Std. Error t value Pr(>|t|)  ## (Intercept) 1.6121 6.8463 0.235 0.8165  ## pH 2.3741 0.9443 2.514 0.0217 \*  ## ---  ## Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  ##  ## Residual standard error: 2.547 on 18 degrees of freedom  ## (2 observations deleted due to missingness)  ## Multiple R-squared: 0.2599, Adjusted R-squared: 0.2188  ## F-statistic: 6.321 on 1 and 18 DF, p-value: 0.02166 |

**You also get a Multiple R-squared value and an Adjusted R-squared value. These values refer to how much of the variation in temperature is explained by our predictor pH. R-squared values tend to increase as you add more terms to your model (see below), but you also need to account for overfitting. The Adjusted R-squared value takes into account how many terms your model has and how many data points are available in the response variable. Looking at our R-squared values, this is not a great model, which makes sense, as pH might not be expected to influence temperature values.**

**You will want to assess your model, and the assumptions. You can use some simple model-checking plots:**

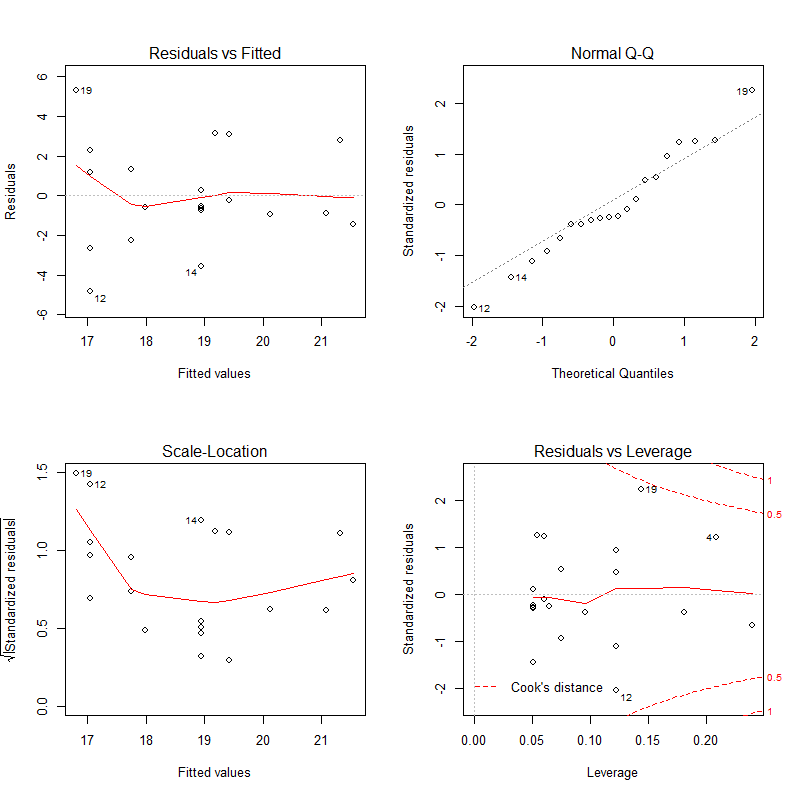
|  |
| --- |
| # plots for evaluating your regression  plot(lm\_sites) |

This will produce a set of four plots: residuals versus fitted values, a Q-Q plot of standardized residuals, a scale-location plot (square roots of standardized residuals versus fitted values, and a plot of residuals versus leverage that adds bands corresponding to Cook's distances of 0.5 and 1.

R will prompt you to click on the graph window or press Enter before showing each plot, but we can do better.

|  |
| --- |
| par(mfrow=c(2,2)) |

Type  to set your graphics window to show four plots at once, in a layout with 2 rows and 2 columns. Then redo the graph using plot(lmfit). To go back to a single graph per window use par(mfrow=c(1,1)). There are many other ways to customize your graphs by setting high-level parameters, type ?par to learn more.



We can also do multiple linear regressions:

|  |
| --- |
| # multiple linear regression  lm\_sites2 <- lm(Temp ~ pH + rain\_mm, data=sitedata)  summary(lm\_sites2)  ## Call:  ## lm(formula = Temp ~ pH + rain\_mm, data = sitedata)  ## Residuals:  ## Min 1Q Median 3Q Max  ## -2.1466 -1.5023 -0.4328 1.0518 3.6008  ##  ## Coefficients:  ## Estimate Std. Error t value Pr(>|t|)  ## (Intercept) 0.61430 5.97382 0.103 0.9196  ## pH 2.05228 0.79556 2.580 0.0218 \*  ## rain\_mm 0.22948 0.07994 2.871 0.0123 \*  ## ---  ## Signif. codes:  ## 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1  ##  ## Residual standard error: 2.001 on 14 degrees of freedom  ## (5 observations deleted due to missingness)  ## Multiple R-squared: 0.5017, Adjusted R-squared: 0.4305  ## F-statistic: 7.046 on 2 and 14 DF, p-value: 0.007634 |

Some other helpful functions when using linear models are coefficients(), residuals(), and fitted.values(). See ?lm for more information.

All of your standard modeling approaches (and then some) are available in R, including typical variable selection techniques (e.g. stepwise with AIC) and logistic regression, which is implemented with the rest of the generalized linear models in glm(). Interaction terms can be specified directly in the model.

**Recommended sources for learning further:**

The Analysis of Biological Data by Whitlock and Schluter - <http://whitlockschluter.zoology.ubc.ca/> This book provides an introduction to modern statistical techniques for analyzing biological data and includes R code!

The R Book – Michael Crawley – all the details on statistical analyses in R

Also see the Cookbook for R. The goal of the cookbook is to provide solutions to common tasks and problems in analyzing data. <http://www.cookbook-r.com/> It includes code you can copy and paste into R.