**Lab 8: Randomization and Resampling Statistics**

**8.1 Randomization to test for differences between two treatments**

Below is the code you saw in the lecture, but modified. The aim is to perform a robust statistical test to see if there is a difference in mean fecundity between two groups of snakes, hypothetically reared at high and low temperature. We define fitness as the number of eggs laid during the breeding season. In the lecture, we sampled 10 individuals for each group from two normal distributions with different means, and then performed resampling with replacement from these groups pooled together (see lecture slides). In this modified example, we are again sampling 10 individuals for each group, however, the distribution of the data (fecundity) is very different. The number of eggs laid follows a poisson processes (discrete counts simulated with the ***rpois()*** command) and increases at High temperature (because snakes are ectotherms and their physiology depends on temperature). However, the fecundity is also dependent on whether the snake is fertile or not, and fertility decreases when temperatures are too high (probably the main reason why human testicles are carried outside of the body!). Fertility follows a binomial process (0 or 1 outcomes simulated with the ***rbinom()*** command). Thus, fecundity is the product of these two processes, and this is quite different from a normal distribution. How will this affect our test of significance - can we use randomization to produce robust p-values?

#First, let’s create your “real” data from the experiment:

High <- rpois(10, lambda = 6) \* rbinom(10, 1, 0.7) #you can change values

Low <- rpois(10, lambda = 3) \* rbinom(10, 1, 0.9) #if you like.

par(mfrow=c(1,2))

hist(High) #Look at the distributions

hist(Low) #Data do not look normal, so assumptions of parametric tests are not fulfilled.

#Let’s calculate a test statistic. I choose the difference in means of the two groups:

**obs.diff** <- mean(High)-mean(Low)

#Let’s make a dataset of all samples from which to sample randomly with replacement:

total <- c(High, Low)

#Let’s do the randomization 10,000 times. We can use a *for* loop for that:

**random.diff** <- array(0,10000) #create an array in which to store output

for(i in 1:10000) { #define number of simulations

**A** <- sample(total,10, replace=T) #create a random sample with same size as experiment

**B** <- sample(total,10, replace=T) #create a random sample with same size as experiment

**random.diff**[i] <- mean(**A**)-mean(**B**) #calculate test statistic for each simulation and store it

}

hist(**random.diff**, xlim=c(-5,5)) #histogram of statistic from randomizations

abline(v = **obs.diff**, col = "red", lwd=3) #What was the statistic in the real sample?

2\*sum(ifelse(**random.diff** > **obs.diff**, 1, 0)) / 10000 #Get the two-sided p-value

t.test(High, Low, var.equal=T) #Compare to what you would get from a t-test

*# Note that the t-test assumes a normal distribution – which is not correct!*

***8.1a. How do the two tests compare? Do you get the same qualitative result as your bench-mates (you can also run the simulation many times)? Can you explain why the two tests are different? Which one do you consider to be most accurate?***

**8.2 the Mantel test for association**

In the lecture you compared genetic and geographical distances for 19 village tribes. This generated 171 data-points, but these are made up of the 19 villages, so the 171 pair-wise (genetic and geographical) distances that were correlated do not represent independent observations. A classic parametric test would struggle since the degrees of freedom for this test is a bit unclear.

A simpler way to calculate a p-value for the correlation between genetic and geographic distance is to apply the mantel test. The mantel test performs the calculation of the dependence between two matrices by randomly reshuffling the data in one of the matrices before the calculation. This can be done many times to generate a null distribution of the test statistic – usually a correlation (like the example in the lecture), or some other measure - like the z-statistic. The Z-statistic reported is NOT the same as the z-score that you learnt about before (from normally distributed data standardized by dividing by the standard deviation). The Z-statistic in this case refers to the sum of the product of all cross-matrix pairwise elements. This is calculated from your two matrices, and then compared to the corresponding values for all the rearranged (permuted) matrices. If the z-statistic is larger for our observed matrices than for the random matrices, the correlation will also be greater than for the random matrices. If greater or smaller than 97.5% of all random matrix comparisons, it is significant at the alpha level of 0.05 with a two-sided hypothesis test.

We can then test if our observed test statistic falls outside 95% of the randomly generated values, which would indicate that the two matrices are related/dependent.

Let’s load the ***ape*** package to run the mantel test on two random matrices that should be unrelated:

library(ape)

q1 <- matrix(runif(36), nrow = 6) #Create a 6x6 matrix

q2 <- matrix(runif(36), nrow = 6) #One more

mantel.test(q1, q2, graph = TRUE, nperm = 999, alternative = "two.sided",

main = "Mantel test: a random example with 6 X 6 matrices",

xlab = "z-statistic", ylab = "Density",

sub = "The vertical line shows the observed z-statistic")

Now let’s create a third matrix from the first two and run the mantel test on the first and third matrix. These should of course be correlated since we created the third matrix using the first one:

q3 <- q1+q2 #Create a third matrix from the first two.

mantel.test(q1, q3, graph = TRUE, nperm = 999, alternative = "two.sided",

main = "Mantel test: a random example with 6 X 6 matrices",

xlab = "z-statistic", ylab = "Density",

sub = "The vertical line shows the observed z-statistic")

***8.2a Make sure you understand the general approach and output. Note that there are several packages in R that can perform the mantel test.***

**8.3 Bootstrapping a correlation**

Let’s revisit the *American Crime* data and correct two issues from lab 5 (have a look at exercise 5.2: Simple Regression if you don’t remember). First, rather than performing simple regression, we should have probably estimated a correlation (and not a regression slope) between the two variables [violent.crime] and [police.funding], since it was a bit unclear whether there is a unidirectional causality between the variables. Second, there was this nasty outlier, and no matter how we dealt with it (transforming, removing, or trying robust regression) it was not quite satisfactory, and results were heavily dependent on the outlier. Let’s try to estimate a correlation between violent crime rate and police funding using boostrap statistics.

crime <- read.delim("AmericanCrime.txt")

#First, let’s look at what a parametric test looks like (Pearson’s moment correlation):

cor.test(crime$police.funding, crime$violent.crime)

#Now let’s perform non-parametric bootstrapping of the correlation. First, we need to download the #*boot* package

library(boot)

#Here comes the bootstrap. Note that this is a bit complex, and there are several alternatives in how to apply the function. First, we specify a function that should be applied to each #bootstrapped sample. Remember, bootstrap samples from the original data with replacement.

booting <- function(data, i) { #”data” is your dataset, and “i” is in index telling *boot*

resamp <- data[i,] #what to randomly sample (by rows, for all columns)

return(cor(resamp$violent.crime, resamp$police.funding)) }

”return” gives what should be the output in each iteration. Here we choose to store the correlation coefficient as our test statistic.

#Run bootstrap

crimeboot <- boot(crime, booting, R = 1000) #apply our specified function “booting” to

#the *crime* dataset

#Bootstrap estimate

mean(crimeboot$t)

median(crimeboot$t)

”t” is the *boot* functions internal variable where the test statistic is stored. In our case, this is the correlation coefficient, and we now have 1000 bootstrapped estimates of that correlation.

crimeboot

In the brief summary of the results, we get info on the original correlation that we observed (also labelled “t0”, the “bias” which shows how the mean estimate from our bootstrapped samples differ from the observed original correlation, and we get the standard error of the correlation based on our bootstrapped values (which is actually the standard deviation of our bootstrapped values).

#We can use the bootstrap estimates to calculate confidence intervals (and all kinds of things)

boot.ci(crimeboot) #default is 95% CI:s

There are several CIs calculated. Perhaps we can focus on the “Normal”, “Percentile”, and the “BCa”. The normal CI uses an approximation assuming that bootstrap estimates follow an approximately normal distribution, which often may not be true (especially for correlations which are bounded at 1). Ther Percentile method simple takes the values corresponding to the lower and upper 2.5% of the bootstrapped samples. The BCa takes the empirical data distribution into account when calculating the CI from the bootstrap values. It is a more computer intensive calculation (but nothing worth thinking about with modern computers) which often gives more robust estimates.

Which method does the best job? We can plot the data to look at the distribution of bootstrap estimates. If that is skewed, it might imply that the normal estimate will be a bit off.

#Plot the bootstraps:

hist(crimeboot$t, 100)

abline(v = mean(crimeboot$t), col = "blue", lwd = 2) #Plot the mean of bootstrapped estimates

abline(v = median(crimeboot$t), col = "purple", lwd = 2) #Plot the median

abline(v = crimeboot$t0, col = "red", lwd = 2) #Plot the correlation in the raw data

***8.3a Thinking back to that outlier, how do you think random resampling of the data affected the estimate of the correlation; in which bootstrapped estimates of the correlation do you think the outlier got included (perhaps more than once), and in which do you think it was left out?***

***8.3b Perform a parametric correlation test like at the start of the exercise, but do it on logged data, or after removing the outlier. How do the correlations and the 95% CIs from the parametric tests compare to the CIs from the bootstrap method?***