Hypothesis Test Exercises

The data files used in the exercises below can be found on the blackboard site! Download them from the blackboard site and save them in an appropriate folder on your computer or network drive. Point your R session to this location with this command setwd(D:/Your/Folder/Goes\_here).

# Exercise 1

In the first exercise we are using the built-in data set precip on precipitation data from 70 US cities (see also book page 119).

1. Inspect the data set.

* > precip

1. Assume we know the world average precipitation is 38. We expect that the average from the 70 US cities is lower. Using an hypothesis test, can you show this to be true? Although this is not a statistics lecture, try to give an answer as complete as possible. Use the function t.test to calculate the t-test.

* > t.test(precip, mu=38, alternative="less")

# Exercise 2

In this exercise, we practise the two-sample t-test using the built-in dataset 'mtcars' (see page 126). Our goal will be to find out if manual transmission cars have a higher mileage than automatic cars. We will assume equal variances.

1. Inspect the data. Make a boxplot showing the distributions of the variable miles per gallon (mpg) for automatic and manual transmission cars.

* > head(mtcars)
* > manual <- which(mtcars$am==0)  
  > auto <- which(mtcars$am==1)  
  > boxplot(mtcars$mpg[manual], mtcars$mpg[auto], ylab="mpg", names=c("manual", "auto"))

1. Do the test. What are your conclusions?

* > t.test(mtcars$mpg[manual], mtcars$mpg[auto], var.equal = TRUE, alternative="less")

1. To calculate the p-value, We use the fact that our test statistic follows approximately a t-18 distribution if the H0 is true. If we did not know this we could also have used a permutation test. To find the distribution in a permutation test we permute the group labels, ensuring no significant difference exists anymore between the two groups. In this distribution, we find the actual test statistic value and we determine the p-value in the permuted distribution.

* > distr <- sapply(1:10000, function(i) {  
  + index <- sample(mtcars$am)  
  + manual <- which(index==0)  
  + auto <- which(index==1)  
  + t.test(mtcars$mpg[manual], mtcars$mpg[auto], var.equal = TRUE, alternative="less")$statistic  
  + })  
  > hist(distr)  
  > real\_t\_statistic <- t.test(mtcars$mpg[manual], mtcars$mpg[auto], var.equal = TRUE, alternative="less")$statistic  
  > points(real\_t\_statistic,0, col="red", pch=16)  
  > sprintf("%f", sum(distr<real\_t\_statistic)/length(distr))

1. Explain an eventual discrepancy between the p-value from the t-test and the permutation test. How could you make these p-values more similar?

# Exercise 3

High-quality expression profiles were successfully derived from 52 prostate tumours and 50 non-tumour prostate samples from patients undergoing surgery. Oligonucleotide microarrays containing probes for approximately 12600 genes. Since prostate tumours are among the most heterogeneous of cancers, both histologically and clinically, the goal here is to classify tumour and non-tumour samples. The dataset consists of 102 prostate tissues of which 50 are normal and 52 tumour samples. The number of gene expression levels is 12600.

1. Load in the data file Singh.rda.

* > load("Singh.rda")
* The variable Singh contains a list with 4 named elements, X, y, Xt, yt, which are respectively the training design matrix, training classes, test design matrix and test classes. The groups are indicated by a 1 or -1 according as tissue is normal/tumour. There are 12600 gene expressions. We focus in this exercise on the training part of the data (in variable Singh$X).

1. Inspect the data and the elements of the list. Make sure you understand the structure.

* > names(Singh)  
  > dim(Singh$X)  
  > Singh$y

1. Create 2 separate variables containing all gene expressions. One for the healthy individuals and one for the diseased individuals.

* > healthy <- Singh$X[Singh$y==1,] # 1=healthy  
  > diseased <- Singh$X[Singh$y==-1,] #-1=diseased

1. Do a two sample t-test one the first gene. Interpret de outcome. Is there a significant difference in expression level for gene 1 between the 2 groups?

* > t.test(healthy[,1], diseased[,1], "two.sided")

1. Do a two sample t-test on all 12600 genes in the training set. How many significant differences do you find (use )? Create a histogram of all p-values

* > pvals <- sapply(1:ncol(healthy), function(i) {  
  + t.test(healthy[,i], diseased[,i], "two.sided")$p.value  
  + })  
  > hist(pvals)

1. Do pairwise comparisons according to Benjamini-Hochberg (see slides for details). How many significant differences are you left with?

* > j=1:length(pvals)  
  > m=length(pvals)  
  > delta=0.05  
  > bhfdr <- delta\*(j/m)  
  >   
  > plot(bhfdr, xlab='index', ylab='p-val')  
  > points(sort(pvals), col='red')  
  >   
  > max(which(sort(pvals)<=bhfdr))

# Exercise 4

García-Arenzana et al. (2014) tested associations of 25 dietary variables with mammographic density, an important risk factor for breast cancer, in Spanish women. For these 25 variables the obtained individual p-values. We are going to use this data set to practice multiple testing corrections.

1. The first step will be to load the data into R. The data file is given as an Excel file which means we need an extra package to read the xlsx file. There are several packages that can do this, but we will use the package XLConnect. If not installed before, do this using this command:

* > install.packages("XLConnect")
* Load the package and read in the excel file.
* > require(XLConnect)  
  > theData <- readWorksheet(loadWorkbook("dietaryvariables.xlsx"), sheet=1)

1. R has built in methods to adjust a series of p-values either to control the family-wise error rate or to control the false discovery rate. Not all these methods have been discussed (in depth) in the lecture. However, we can still use them (provided we do it correct). The methods Holm, Hochberg, Hommel, and Bonferroni control the family-wise error rate. These methods attempt to limit the probability of even one false discovery (a type I error, incorrectly rejecting the null hypothesis when there is no real effect), and so are all relatively strong (conservative). The methods BH (Benjamini-Hochberg, which is the same as FDR in R) and BY control the false discovery rate. These methods attempt to control the expected proportion of false discoveries.

* See the help page of the function p.adjust on information how to use these methods.
* > ?p.adjust
* These methods require only the p-values to adjust and the number of p-values that are being compared.

1. First, we order the data by p-value.

* > theData = theData[order(theData$Raw.p),]
* Check if data is ordered the way we intended
* > theData

1. Perform p-value adjustments using the available methods in the p.adjust function. The available methods are "bonferroni", "Benjamini Hochberg", "holm", "hochberg", "hommel", "Benjamini & Yekutieli". Save all adjusted p-values in the theData variable.

* > theData$Bonferroni <- p.adjust(theData$Raw.p, method = "bonferroni")  
  > theData$BH <- p.adjust(theData$Raw.p, method = "BH")  
  > theData$Holm <- p.adjust(theData$Raw.p, method = "holm")  
  > theData$Hochberg <- p.adjust(theData$Raw.p, method = "hochberg")  
  > theData$Hommel <- p.adjust(theData$Raw.p,method = "hommel")  
  > theData$BY <- p.adjust(theData$Raw.p, method = "BY")
* Inspect the resulting theData variable.
* > theData

1. Plot all the different p-values in a single figure.

* > X <- theData$Raw.p  
  > Y <- cbind(theData$Bonferroni, theData$BH, theData$Holm, theData$Hochberg, theData$Hommel, theData$BY)  
  > matplot(X, Y, xlab="Raw p-value", ylab="Adjusted p-value", type="l", asp=1, col=1:6, lty=1, lwd=2)  
  > legend('bottomright',legend = c("Bonferroni", "BH", "Holm", "Hochberg", "Hommel", "BY"), col=1:6, cex=1, pch=16)  
  > abline(0, 1, col=1, lty=2, lwd=1)

1. Interpret the last figure. What do the lines represent? Which method is most strict, which one the most lenient?