Some suggested solutions to computer lab 2  
Course: Introduction to R in biostatistics

February 28, 2017

1 Scripting

1.

2.

3.

4.

5. #
   # Answers to exercises
   # by Petter Mostad
   # and Jonatan Kallus (kallus@chalmers.se)
   #

   # Part 1, exercise 5:

   # Reading in the data:
   myData <- c(34, 54, 25, 53, 24, 41, 49, 32, 26, 51)

   # Some summary statistics
   print(summary(myData))

   # Printing out the standard deviation
   print(sd(myData))

   # Plotting the data values against their indices:
   plot(myData)

   # Making a boxplot:
   boxplot(myData)
2  Data structures in R

1.  

2.  

3.  > myFrame <- data.frame(attribute1, attribute2, attribute3)
   > names(myFrame) <- c("Age", "Number", "Sex")

4.  > myFrame$Age[2] <- 49
   > myFrame[1,1] <- 32

5.  For example

   > myFrame <- data.frame(attribute1, attribute2, attribute3, stringsAsFactors=F)

   or

   > myFrame <- data.frame(attribute1, attribute2, I(attribute3))

6.  > myList <- list(attribute1, attribute2, attribute3)

3  Input and output of data

We are finally getting to a very important point: Input and output of data. Real
data sets will most often be in the form of an output from some other program. A
general way of inputing such data to R, is to make sure it is in some kind of
text format.

1.  

2.  > myData <- read.delim("Example.txt")

3.  > myData <- read.delim("Example.txt", stringsAsFactors = F)

4.  

5.  > newNewData <- newData[substr(newData$Genename, 2, 2)="-",]

6.  

7.  

4 R programming

1. `> myReplace(myData$Genename)`
   To actually make the replacements in myData, you have to write
   
   `> myData$Genename <- myReplace(myData$Genename)`

2. `function(v, oldstring="No", newstring="Yes") {
    v[v==oldstring] <- newstring
    v
}

3. `for (i in 1:10) {
    print(myData$Genename[i])
}

4. `for (i in 1:20) {
    if (abs(myData[i, 2] - myData[i, 3]) > 5000) {
        print(myData$Genename[i])
    }
}

}