Exercise sheet: Day 1

Vangelis Theodorakis, Fatemeh Behjati, Julien Gagneur

11 October, 2020

Contents

1 Vectors

First, create three named numeric vectors of size 10, 11 and 12 respectively in the following manner:

- One vector with the "colon" approach: *from:to*
- One vector with the `seq()` function: *seq(from, to)*
- And one vector with the `seq()` function and the `by` argument: *seq(from, to, by)*

For easier naming you can use the vector `letters` or `LETTERS` which contain the latin alphabet in small and capital, respectively. In order to select specific letters just use e.g. `letters[1:4]` to get the first four letters. Check their types. What is the outcome? Where do you think the difference comes from?

Then combine all three vectors in a list. Check the attributes of the vectors and the list. What is the difference and why?

Hint: If list elements have no names, we can access them with the double brackets and an index, e.g. `my_list[[1]]`

```
# Answer :

# A. Create vectors
vector.1 <- 1:10
names(vector.1) <- letters[vector.1]

vector.2 <- seq(1, 11)
names(vector.2) <- letters[vector.2]

vector.3 <- seq(1, 12, by = 1)
names(vector.3) <- letters[vector.3]

typeof(vector.1)
## [1] "integer"
typeof(vector.2)
## [1] "integer"
typeof(vector.3)
## [1] "double"

# B. Combine in a list
awesome.list <- list(vector.1, vector.2, vector.3)
attributes(vector.1)
## $names
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"
attributes(vector.2)
## $names
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k"
attributes(vector.3)
## $names
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j" "k" "l"
attributes(awesome.list)
## NULL
attributes(awesome.list[[1]])
```

```
## $names
##  [1] "a" "b" "c" "d" "e" "f" "g" "h" "i" "j"

## Why is the last vector of type double and not integer?
## By default seq returns integers from:to. But the ‘by‘
## parameter returns always doubles

## myList got no names since we did not assign any compared to our vectors
```

2 Factors

```
f1 <- factor(letters)
levels(f1) <- rev(levels(f1))
f2 <- rev(factor(letters))
f3 <- factor(letters, levels = rev(letters))
```

The function `rev` reverses the order of an order-able object. What is the difference between f1, f2 and f3? Why?

```
# Answer :

f1 <- factor(letters)
levels(f1) <- rev(levels(f1))
# f1 goes from z - a, but the underlying encoding goes from z = 1 to a = 26
# We create the vector with the letters a to z and the mapped integer
# structure 1 to 26. THEN we reverse the levels = the mapping. As 1 becomes z
# and a becomes 26 the letters are mapped back to the unchanged integer
# structure and hence reversed.
f1
##  [1] z y x w v u t s r q p o n m l k j i h g f e d c b a
## Levels: z y x w v u t s r q p o n m l k j i h g f e d c b a

f2 <- rev(factor(letters))
# f2 goes from z - a, but the underlying encoding goes from a = 1 to z = 26
# We create the vector with the letters a to z and the mapped integer
# structure 1 to 26. Then we reverse the vector, i.e. the underlying integers,
# hence the vector gets reversed, but not the levels.
f2
##  [1] z y x w v u t s r q p o n m l k j i h g f e d c b a
## Levels: a b c d e f g h i j k l m n o p q r s t u v w x y z

f3 <- factor(letters, levels = rev(letters))
# f3 goes from a - z, but the underlying encoding goes from z = 1 to a = 26.
# We create the vector with the letters a to z BUT the mapped integer
# structure 26 to 1. Hence the levels but not the vector are reversed.
f3
##  [1] a b c d e f g h i j k l m n o p q r s t u v w x y z
## Levels: z y x w v u t s r q p o n m l k j i h g f e d c b a
```

```
# Reversing f3 will give f1
rev(f3)
##  [1] z y x w v u t s r q p o n m l k j i h g f e d c b a
## Levels: z y x w v u t s r q p o n m l k j i h g f e d c b a
```

3 Data tables

The purpose of this exercise is to get familiarize with data.table and try out some of its useful features.

3.1 Basic operations

Please follow the steps listed below:

1) Download the GTEx data (annotation v7) from the following link: https://storage.googleapis.com/gtex_analysis_v7/annotations/GTEx_v7_Annotations_SampleAttributesDS.txt
2) Read the file downloaded above and store it in a variable named: *data*.
3) Inspect *data* by checking properties such as: *class(data)*, *dim(data)*, *colnames(data)*, *data[1:3, 1:5]*, *unique(data$SMTS)*.
4) Count how many NA's exist in *data*.

```
# Answer :
library(data.table)
data <- fread("~/GTEx_v7_Annotations_SampleAttributesDS.txt")

print("class of data is")
## [1] "class of data is"
class(data)
## [1] "data.table" "data.frame"

print("dim of data is")
## [1] "dim of data is"
dim(data)
## [1] 15598    63

print("column names of data are")
## [1] "column names of data are"
colnames(data)
##  [1] "SAMPID"     "SMATSSCR"   "SMCENTER"   "SMPTHNTS"   "SMRIN"      "SMTS"
##  [7] "SMTSD"      "SMUBRID"    "SMTSISCH"   "SMTSPAX"    "SMNABTCH"   "SMNABTCHT"
## [13] "SMNABTCHD" "SMGEBTCH"   "SMGEBTCHD" "SMGEBTCHT" "SMAFRZE"    "SMGTC"
## [19] "SME2MPRT"   "SMCHMPRS"   "SMNTRART"   "SMNUMGPS"   "SMMAPRT"    "SMEXNCRT"
## [25] "SM550NRM"   "SMGNSDTC"   "SMUNMPRT"   "SM350NRM"   "SMRDLGTH"   "SMMNCPB"
## [31] "SME1MMRT"   "SMSFLGTH"   "SMESTLBS"   "SMMPPD"     "SMNTERRT"   "SMRRNANM"
## [37] "SMRDTTL"    "SMVQCFL"    "SMMNCV"     "SMTRSCPT"   "SMMPPDPR"   "SMCGLGTH"
## [43] "SMGAPPCT"   "SMUNPDRD"   "SMNTRNRT"   "SMMPUNRT"   "SMEXPEFF"   "SMMPPDUN"
## [49] "SME2MMRT"   "SME2ANTI"   "SMALTALG"   "SME2SNSE"   "SMMFLGTH"   "SME1ANTI"
```

```
## [55] "SMSPLTRD"  "SMBSMMRT"  "SME1SNSE"  "SME1PCTS"  "SMRRNART"  "SME1MPRT"
## [61] "SMNUM5CD"  "SMDPMPRT"  "SME2PCTS"

print("a small subset of data looks like")
## [1] "a small subset of data looks like"
data[1:3, 1:5]
##                      SAMPID SMATSSCR SMCENTER SMPTHNTS SMRIN
## 1: GTEX-1117F-0003-SM-58Q7G       NA       B1             NA
## 2: GTEX-1117F-0003-SM-5DWSB       NA       B1             NA
## 3: GTEX-1117F-0003-SM-6WBT7       NA       B1             NA

print("tissue types in data:")
## [1] "tissue types in data:"
unique(data$SMTS)
##  [1] "Blood"           "Adipose Tissue"  "Muscle"          "Blood Vessel"
##  [5] "Heart"           "Ovary"           "Uterus"          "Vagina"
##  [9] "Breast"          "Skin"            "Salivary Gland"  "Brain"
## [13] "Adrenal Gland"   "Thyroid"         "Lung"            "Spleen"
## [17] "Pancreas"        "Esophagus"       "Stomach"         "Colon"
## [21] "Small Intestine" "Prostate"        "Testis"          "Nerve"
## [25] "Pituitary"       "Liver"           "Kidney"          "Fallopian Tube"
## [29] "Bladder"         "Cervix Uteri"    "Bone Marrow"
```

3.2 More exciting operations

Continue from the previous part and perform the following actions:

3) Subset the data based on the *Brain* cell type sample and store the result in a variable called: *data_Brain*.

4) Inspect the *data_Brain* similar to the point 3 above.

5) Examine the range of values in *SMEXPEFF* field of *data_Brain*. How can you make it more meaningful?

6) For *data_Brain*, compute the average of the values stored in the "SMEXPEFF" column. Also, compute the min of values stored in "SME1MPRT".

7) Compute the correlation between the two columns mentioned above.

8) Remove the rows that are NA from the *data_Brain$SMEXPEFF*. Retry the correlation on the NA-removed *data_Brain_noNA*.

Hint: Use the is.na() function to find the rows that are NA.

```
# Answer :
#3
data_Brain <- data[data$SMTS == "Brain", ]

#4
print("class of data_Brain is")
## [1] "class of data_Brain is"
class(data_Brain)
## [1] "data.table" "data.frame"
```

```r
print("dim of data_Brain is")
## [1] "dim of data_Brain is"
dim(data_Brain)
## [1] 2076   63

print("column names of data_Brain are")
## [1] "column names of data_Brain are"
colnames(data_Brain)
##  [1] "SAMPID"    "SMATSSCR"  "SMCENTER"  "SMPTHNTS"  "SMRIN"     "SMTS"
##  [7] "SMTSD"     "SMUBRID"   "SMTSISCH"  "SMTSPAX"   "SMNABTCH"  "SMNABTCHT"
## [13] "SMNABTCHD" "SMGEBTCH"  "SMGEBTCHD" "SMGEBTCHT" "SMAFRZE"   "SMGTC"
## [19] "SME2MPRT"  "SMCHMPRS"  "SMNTRART"  "SMNUMGPS"  "SMMAPRT"   "SMEXNCRT"
## [25] "SM550NRM"  "SMGNSDTC"  "SMUNMPRT"  "SM350NRM"  "SMRDLGTH"  "SMMNCPB"
## [31] "SME1MMRT"  "SMSFLGTH"  "SMESTLBS"  "SMMPPD"    "SMNTERRT"  "SMRRNANM"
## [37] "SMRDTTL"   "SMVQCFL"   "SMMNCV"    "SMTRSCPT"  "SMMPPDPR"  "SMCGLGTH"
## [43] "SMGAPPCT"  "SMUNPDRD"  "SMNTRNRT"  "SMMPUNRT"  "SMEXPEFF"  "SMMPPDUN"
## [49] "SME2MMRT"  "SME2ANTI"  "SMALTALG"  "SME2SNSE"  "SMMFLGTH"  "SME1ANTI"
## [55] "SMSPLTRD"  "SMBSMMRT"  "SME1SNSE"  "SME1PCTS"  "SMRRNART"  "SME1MPRT"
## [61] "SMNUM5CD"  "SMDPMPRT"  "SME2PCTS"

print("a small subset of data_Brain looks like")
## [1] "a small subset of data_Brain looks like"
data_Brain[1:3, 1:5]
##                      SAMPID SMATSSCR SMCENTER SMPTHNTS SMRIN
## 1: GTEX-1117F-3226-SM-5N9CT        1       B1 2 pieces   6.2
## 2: GTEX-111FC-3126-SM-5GZZ2        1       B1 2 pieces   6.1
## 3: GTEX-111FC-3326-SM-5GZYV        2       B1 2 pieces   7.1

print("tissue types in data_Brain:e")
## [1] "tissue types in data_Brain:e"
unique(data_Brain$SMTS)
## [1] "Brain"

#5
print("range of values in data_Brain$SMEXPEFF (Expression Profiling Efficiency):")
## [1] "range of values in data_Brain$SMEXPEFF (Expression Profiling Efficiency):"
range(data_Brain$SMEXPEFF)
## [1] NA NA

print("range of values in data_Brain$SMEXPEFF when NA's are removed:")
## [1] "range of values in data_Brain$SMEXPEFF when NA's are removed:"
range(data_Brain$SMEXPEFF, na.rm= T)
## [1] 0.07202903 0.92567736
#6
mean(data_Brain$SMEXPEFF)
## [1] NA
mean(data_Brain$SMEXPEFF, na.rm= T)
## [1] 0.7674499

min(data_Brain$SME1MPRT)
```

```
## [1] NA
min(data_Brain$SME1MPRT, na.rm= T)
## [1] 0.08879356
#7
cor(data_Brain$SME1MPRT, data_Brain$SMEXPEFF)
## [1] NA
#8
data_Brain <- data_Brain[!is.na(data_Brain$SMEXPEFF), ]
cor(data_Brain$SME1MPRT, data_Brain$SMEXPEFF)
## [1] 0.8865701
```

4 Looping

- Initialize a variable called *counter* by 0.
- Using a for loop that iterates 10 times, increment *counter* by 1.
- Print the final value in *counter*.

```
#Answer :

# Looping
counter <- 0
for(i in seq(10)){
  counter <- counter + 1
}
print(counter)
## [1] 10
```

Write a function named *get_counts* that takes a GTEx data table as input and outputs the total counts of rows that the sample tissue type (*SMTS*) is *Heart* and the sample analysis freeze (*SMAFRZE*) is *RNASEQ*. How about if you try the same but for *Blood*. If this task was too easy, can you modify your function such that instead of taking only one argument, it takes two additional ones, one for the *SMTS* and another for *SMAFRZE*. Iterate over all possible values of *SMTS* (**Hint:** *unique(data$SMTS)*) and call your function by providing the sample tissue type.

```
#Answer :
get_counts <- function(gtex){
  counter <- 0
  for(i in seq(nrow(gtex)))
    if(gtex$SMTS[i] == "Heart" & gtex$SMAFRZE[i] == "RNASEQ"){
      counter <- counter + 1
    }
  return(counter)
}

fun_res <- get_counts(data)
print(fun_res)
## [1] 600

#modified version
```

```
get_counts2 <- function(gtex, var1, var2){
  counter <- 0
  for(i in seq(nrow(gtex)))
    if(gtex$SMTS[i] == var1 & gtex$SMAFRZE[i] == var2){
      counter <- counter + 1
    }
  return(counter)
}

all_tissues <- unique(data$SMTS)
for(ts in all_tissues){
  fun_res <- get_counts2(data, ts, "RNASEQ")
  print(paste("Number of RNASEQ cases for", ts, ":", fun_res))
}
## [1] "Number of RNASEQ cases for Blood : 537"
## [1] "Number of RNASEQ cases for Adipose Tissue : 797"
## [1] "Number of RNASEQ cases for Muscle : 564"
## [1] "Number of RNASEQ cases for Blood Vessel : 913"
## [1] "Number of RNASEQ cases for Heart : 600"
## [1] "Number of RNASEQ cases for Ovary : 133"
## [1] "Number of RNASEQ cases for Uterus : 111"
## [1] "Number of RNASEQ cases for Vagina : 115"
## [1] "Number of RNASEQ cases for Breast : 290"
## [1] "Number of RNASEQ cases for Skin : 1203"
## [1] "Number of RNASEQ cases for Salivary Gland : 97"
## [1] "Number of RNASEQ cases for Brain : 1671"
## [1] "Number of RNASEQ cases for Adrenal Gland : 190"
## [1] "Number of RNASEQ cases for Thyroid : 446"
## [1] "Number of RNASEQ cases for Lung : 427"
## [1] "Number of RNASEQ cases for Spleen : 162"
## [1] "Number of RNASEQ cases for Pancreas : 248"
## [1] "Number of RNASEQ cases for Esophagus : 1021"
## [1] "Number of RNASEQ cases for Stomach : 262"
## [1] "Number of RNASEQ cases for Colon : 507"
## [1] "Number of RNASEQ cases for Small Intestine : 137"
## [1] "Number of RNASEQ cases for Prostate : 152"
## [1] "Number of RNASEQ cases for Testis : 259"
## [1] "Number of RNASEQ cases for Nerve : 414"
## [1] "Number of RNASEQ cases for Pituitary : 183"
## [1] "Number of RNASEQ cases for Liver : 175"
## [1] "Number of RNASEQ cases for Kidney : 45"
## [1] "Number of RNASEQ cases for Fallopian Tube : 7"
## [1] "Number of RNASEQ cases for Bladder : 11"
## [1] "Number of RNASEQ cases for Cervix Uteri : 11"
## [1] "Number of RNASEQ cases for Bone Marrow : 0"
```