Usefull Base R

Looking up functions

You can look up how a function works with the ? function. If you do not know the functions used in this task you can easily look them up!

```
?data()
```

Saving and loading

Saving things to read them in R later

You can save a lot of time, when you save your processed R files after certain steps. This way you do not have to rerun all your steps everytime again. There are 2 base R functions to save R objects, in a way that you can reopen them directly as R objects again. - save() and load() can be used to save .RData files - saveRDS(), readRDS() are used for .rds files

Task 1

Use the data(iris) data set or any other data set you like. Save the file once as a RData and as a rds and relaod. What is the difference, between the file types?

```
data(iris)
```

Using paste to get you paths and file names cleaned up

When you save something you have to specify a path. This can get messy if you save 10 files and afterwards decide to move the output folder somewhere else. Paste allows you to add strings together. You can in the beginning of your script define the output path once and then refere to this every time you save something. You can also paste certain variables like a cutoff into the file name, so you recognise it later.

Task 2

- Define a output folder for this coding task. And save the files from task 1 in there using paste(). You can use sep to define how the parts inside paste should be conctet.
- Often it is most usefull to use paste0. What does it do?
- Use Sys.Date() to add the date to your filenames

```
# example
outpath = "my/output/path/"
save(paste(outpath, "file1.RData", sep = "."))
```

Reading from and saving to table documents

During you project you might get data files in table formats that you want to read int R. And maybe you want to save some of your results in a common table format like .txt or .csv.

Task 3

Use the given coding-task.txt and coding.task.csv to familarise yourself with the functions below. What options do the functions have and what do they do?

```
read.table()
read.csv()
read.delim()

write.table()
write.csv()
```

Saving figures

If you make plots with ggplot the easies way to save the plot is with ggsave(). However some packages like clusterProfiler and UpsetR do not output a ggplot format. You can save these (and potentially also all ggplots) by turning on a pdf or png device with pdf() or png(). It is important that you turn of the device afterwards (dev.off()). You can specify the figure height, width, poinsize and fontsize for the machine

```
pdf(paste0(outpath,"plot.pdf"))
hist(iris$Sepal.Width)
dev.off()

png(paste0(outpath,"plot.png"))
hist(iris$Sepal.Width)
dev.off()
```

Task 4

Save the histogram from the example above in a 5 x 5 pdf and a 10 x 5 png. Choose a usefull fontsize.

What is inside an R object?

First look

Usefull functions are

```
head()
tail()
table()
summary()
```

Task 5

Try out the functions from above, that you do not know yet. Are there more functions of this kind that you whould recommend?

Logical commands and some math

There are some usefull logical commands, that allow you to check for certain occurences inside a huge data set, without scrolling through it.

```
any()
all()
which()
```

```
min()
max()
sum()

duplicated()
unique()

is.na()
is.finite()
is.infinite()
```

Task 6

Find out the following for the Iris data set: - Are all Petal Lengths smaller then 7? - Does one of the columns contain a zero? - Are there any rows with a Sepal Length of 5.8.

- Which rows contain the versicolor species?
- Which rows d not belng to the versicolor species?
- Which row contains the maximum/minimum Petal width?
- How many Sepal width are bigger then 4? How many are exactly 3?
- Which petal witdh are duplicated/unique?
- Does the sepal length contain an NA or infitite value?
- Are all petal lengths finite numbers?

Random sampling

You can use the following function to randomly draw from a vector.

```
set.seed()
sample()
runif()
```

Task 7

- Use sample to extract 5 random flowers species from the iris species column.
- Use sample and runif to get 4 random numbers between 1 and 20. What is the difference? What happens if you rerun the command?
- Now rerun the command again two times but use set.seed to get the same result both times

Checking versions

When you write a thesis you should write the versions of R, RStudio and all used packages. This is how you can look them up quickly:

```
R.version()
package_version()
RStudio.Version()
```