

Introduction to R



Part II

Part II

Please interrupt with questions

Review

Working with Scripts and Writing
Functions

Libraries and Packages

More Graphics

Working with Scripts and Writing Functions

R scripts

Scripts are a little different from libraries

They are simple text files and don't follow any special requirements

Write scripts for often used procedures

Scripts are saved as .r or .R files

To load/run a script

```
source("myScript.R")
```

All lines of code within myScript will be executed as they were typed into the command line within your R console

Let's write our first script

Coding a generic hist() function

Remember, the hist function's default for binning is not ideal and can skew the shape of the histogram

To avoid this problem write a generic hist function that will work for any numeric vector

Here is what we did:

```
png(filename = "myhist.png" , width=7, height=8.5, units="in", pointsize=12, bg="white", res=600)
h = hist(datVector, breaks=seq(START,END, by=binWidth), col=COLOR, xlab=xAXIS, main=TITLE)
dev.off()
```

What are the varying input variables here?

Writing functions

Right now you have to go into the script and assign the input

If you rerun the script it will overwrite the variables - not ideal

Let's make the process more efficient by writing a function

What should the input variable be?

What should the output be?

```
FuncName <-  
function(input1,input2,...) {  
  
    DO STUFF  
  
    return(output)  
  
}
```

Variable names within a function are local! In other words outside of the function these variables have no meaning.

```

4 #####
5 # reading a numeric tab-separated data file
6 #####
7 # Function assumes that file is in table format
8 # line 1 should have the same length as line 2
9
10 readinTabNumFile <- function(fileName){
11     l1 = scan(fileName, sep = "\t", what = "", nlines = 1)
12
13     tmpM = matrix(scan(fileName, what = "", sep = "\t", skip = 1), ncol = length(l1), byrow = TRUE)
14     myData = matrix(as.numeric(tmpM[,-1]), ncol = length(l1)-1, byrow = FALSE)
15     colnames(myData) = l1[-1]
16     rownames(myData) = tmpM[,1]
17
18     # check if the file was read correctly
19     print(dim(myData))
20     print(head(myData))
21
22     return(myData)
23     # myData is a numeric matrix object
24
25 }
26

```

Libraries and Packages

Libraries

The basic R functions can be extended by libraries and packages

Packages on CRAN and Bioconductor have to follow specific regulations

Packages outside of these archives can still be valuable, but possibly do not follow CRAN's standard

To install and use a package

```
> install.packages("<PackageName>")
```

```
> library(<PackageName>)
```

Let us install ggplot2 - a powerful graphics library

```
> install.packages("ggplot2")
```

R will prompt you to select a mirror, I often chose CA1

After installing the library, it has to be loaded to enable its functions

```
> library(ggplot2)
```

Now all of ggplot2's functions are available

Updating R on Windows

```
> install.packages("installr")
```

```
> library(installr)
```

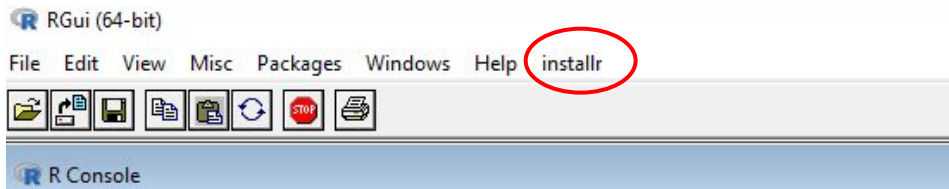
```
> updateR()
```

Click on `installr`

Select Update R

Go through Windows' Wizard Setup

After updating R, you might have to reinstall
some of your package



Updating Libraries

Just like R, libraries consistently upgrade as well

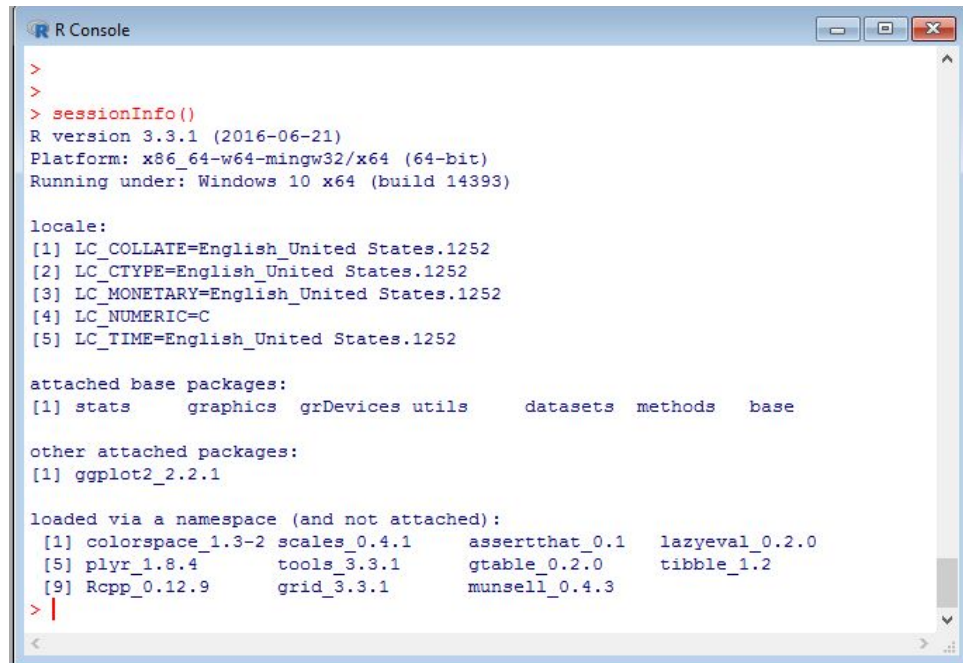
To check which R and library versions you are currently using

```
> sessionInfo()
```

I recommend to record this information with your results as it might be required for publication.

To update R libraries

```
> update.packages()
```



```
R Console
>
>
> sessionInfo()
R version 3.3.1 (2016-06-21)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 14393)

locale:
 [1] LC_COLLATE=English_United States.1252
 [2] LC_CTYPE=English_United States.1252
 [3] LC_MONETARY=English_United States.1252
 [4] LC_NUMERIC=C
 [5] LC_TIME=English_United States.1252

attached base packages:
[1] stats      graphics  grDevices  utils      datasets  methods   base

other attached packages:
[1] ggplot2_2.2.1

loaded via a namespace (and not attached):
 [1] colorspace_1.3-2 scales_0.4.1   assertthat_0.1  lazyeval_0.2.0
 [5] plyr_1.8.4      tools_3.3.1    gtable_0.2.0    tibble_1.2
 [9] Rcpp_0.12.9     grid_3.3.1     munsell_0.4.3
```

More Graphics

Lines, Venn Diagrams

Library: ggplot2

```
> library(ggplot2)

> plot(myData[1,], myData[2,])
> colnames(myData)
> group = c(rep("Old",18), rep("Young",19))
```

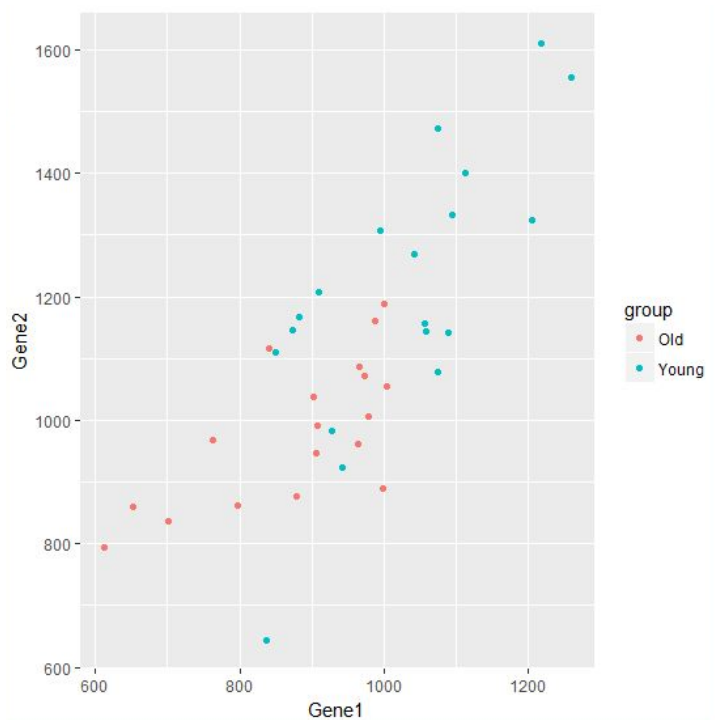
Resources:

<http://ggplot2.org/book/>

<http://tutorials.iq.harvard.edu/R/Rgraphics/Rgraphics.html>

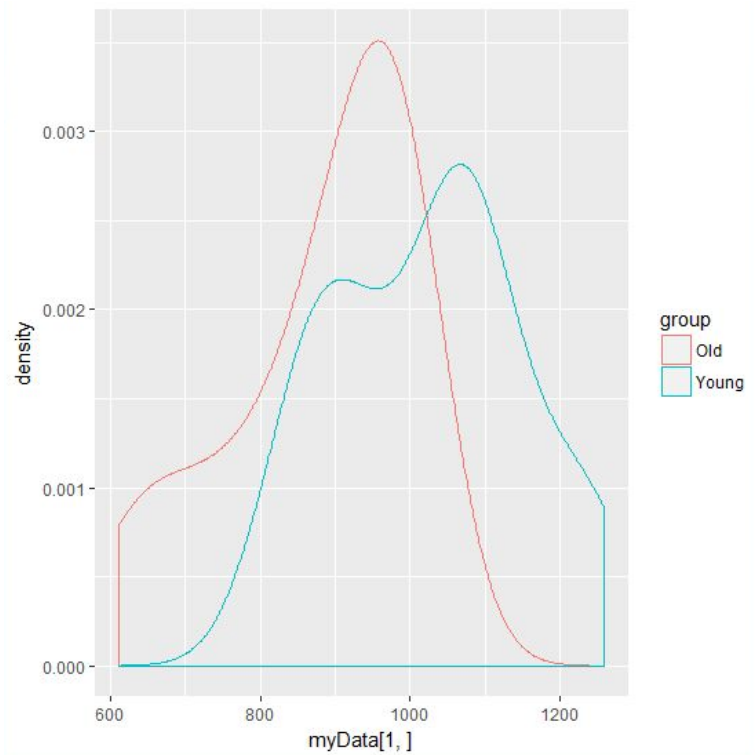
Library: ggplot2

```
> ggplot(myData[1,], myData[2,], colour=group, xlab="Gene1", ylab="Gene2")
```

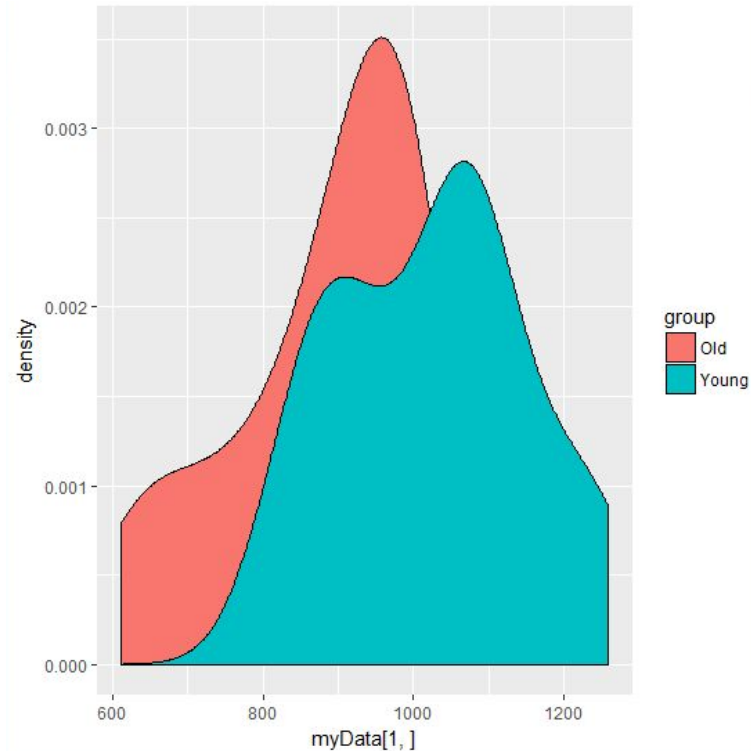


Library: ggplot2

```
qplot(myData[1,], geom="density", colour=group)
```

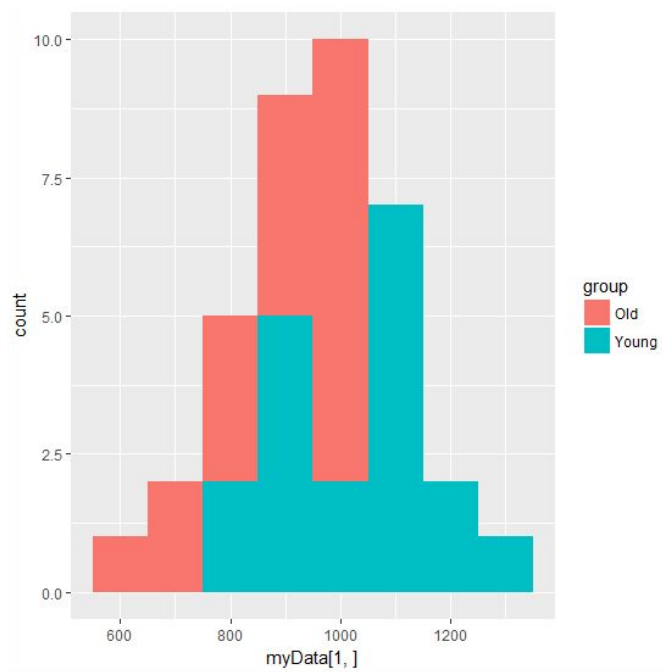
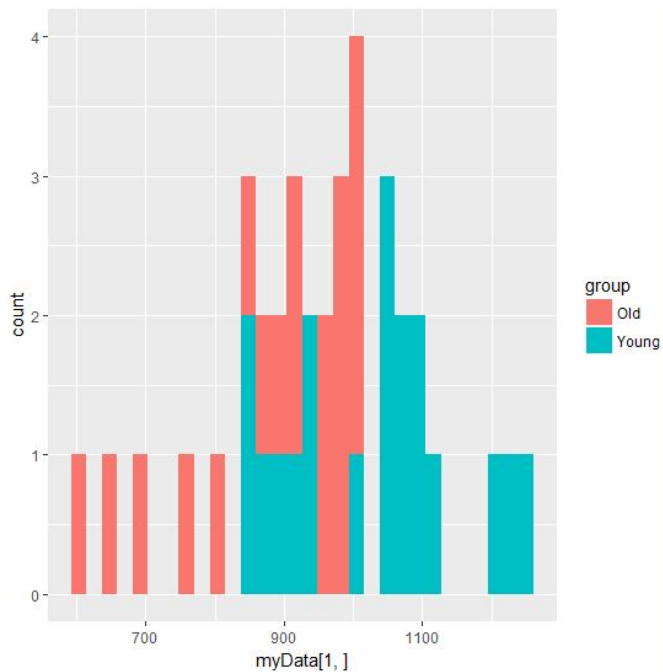


```
qplot(myData[1,], geom="density", fill=group)
```



Library: ggplot2

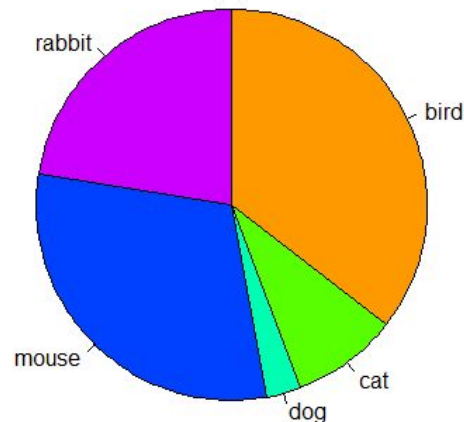
```
qplot(myData[1,], geom="histogram", fill=group)
qplot(myData[1,], geom="histogram", fill=group, binwidth=100)
qplot(myData[1,], geom="histogram", fill=group, binwidth=100, xlab=rownames(myData)[1])
```



Pie Chart

```
> animal = c("cat", "mouse", "dog", "bird", "rabbit")
> count = c(109,378,36,443,280)
> inputT = table(rep(animal,count))

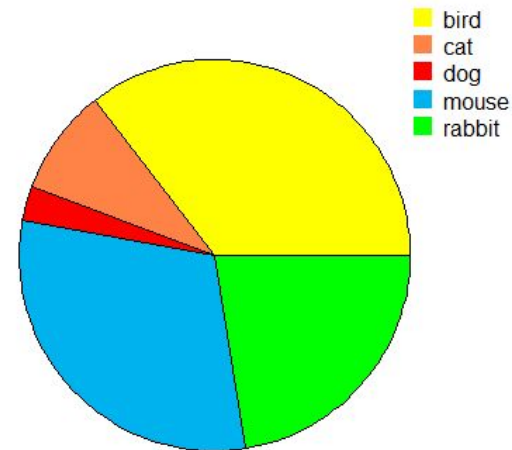
> pie(inputT, col=rainbow(length(animal), start=0.1, end=0.8),
clockwise=T)
```



My animals

```
> myColors = c("yellow1", "sienna1", "red1", "DeepSkyBlue2",
"green")

> pie(inputT, col=myColors, labels=NA, main="My animals")
> legend("topright", legend=row.names(inputT), cex=1, bty="n",
pch=15, pt.cex=1.8, col=myColors, ncol=1)
```



Venn Diagram

```
> install.packages("VennDiagram")
```

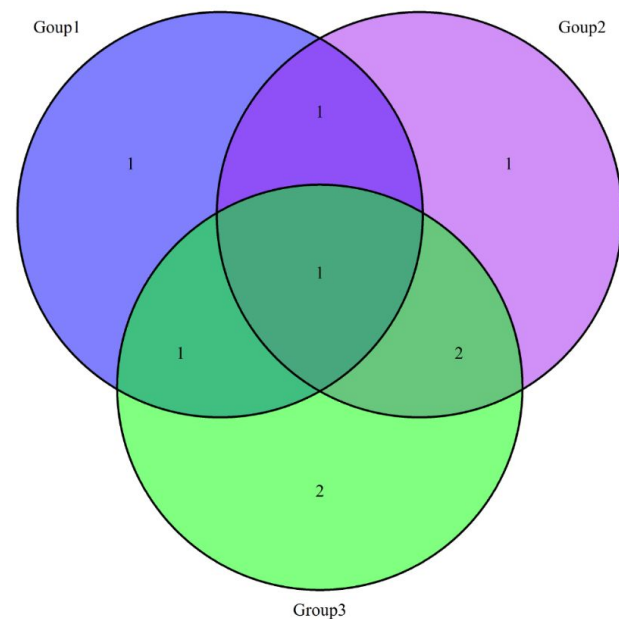
```
> g1 = c("Marie", "Claus", "Stef", "Anna")
```

```
> g2 = c("Tom", "Claus", "Laura", "Stef", "Betty")
```

```
> g3 = c("Marie", "Tom", "Claus", "Clair", "Laura", "Beth")
```

```
> venn.diagram(list(G1 = g1,G2 = g2), fill=c("blue", "purple"), filename="Venn2sets.png")
```

```
> venn.diagram(list(Goup1 = g1,Goup2 = g2, Group3 = g3), fill=c("blue", "purple",  
"green"), filename="Venn3sets.png")
```



Handy functions

> `objects()` does the same as `ls()`

> `rm()`

> `typeof(x)`

> `class()`

> `is.numeric()` > `as.numeric`

> `is.null()`

> `is.matrix()` > `as.matrix`

Review

What did we do again?

How to load, save, and write data

How to investigate data

How to plot a histogram, barplot, scatterplot

Exercise to refresh our brains:

- 1) Read in Data_AA.txt
 - a) Use the script, but you have to make a modification
- 2) What are the dimensions of your dataset?
- 3) What are the column headers?
- 4) What is the average delay arrival time (DAT)?
Minimum DAT? Maximum DAT?
- 5) Plot DEPARTURE_DELAY vs ARRIVAL_DELAY
- 6) Add the identify function ($f(x)=y$) using `abline`

Solution

```
dim(myData)
colnames(myData)
mean(as.numeric(myData[,11]),na.rm=TRUE)
min(as.numeric(myData[,11]),na.rm=TRUE)
max(as.numeric(myData[,11]),na.rm=TRUE)
plot(as.numeric(myData[,10]), as.numeric(myData[,11]), xlab=colnames(myData)[10],
ylab=colnames(myData)[11], pch=16, cex=0.5)
abline(a=1:1400, b=1:1400, col="red", lwd=2)
abline(v=0, col="green", lwd=2)
```

```
# Try, and see what you get, what do you think?
abline(a=0:1400, b=0:1400, col="green", lwd=2)
```