



# TEWA 1: Advanced Data Analysis

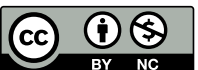
## Lecture 02

Lei Zhang

Social, Cognitive and Affective Neuroscience Unit (SCAN-Unit)  
Department of Cognition, Emotion, and Methods in Psychology

[https://github.com/lei-zhang/tewa1\\_univie](https://github.com/lei-zhang/tewa1_univie)

lei.zhang@univie.ac.at  
lei-zhang.net  
@lei\_zhang\_lz



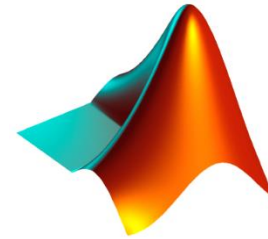
universität  
wien

Fakultät für Psychologie

# INTRO TO COMMON PROGRAMMING LANGUAGES



python<sup>TM</sup>



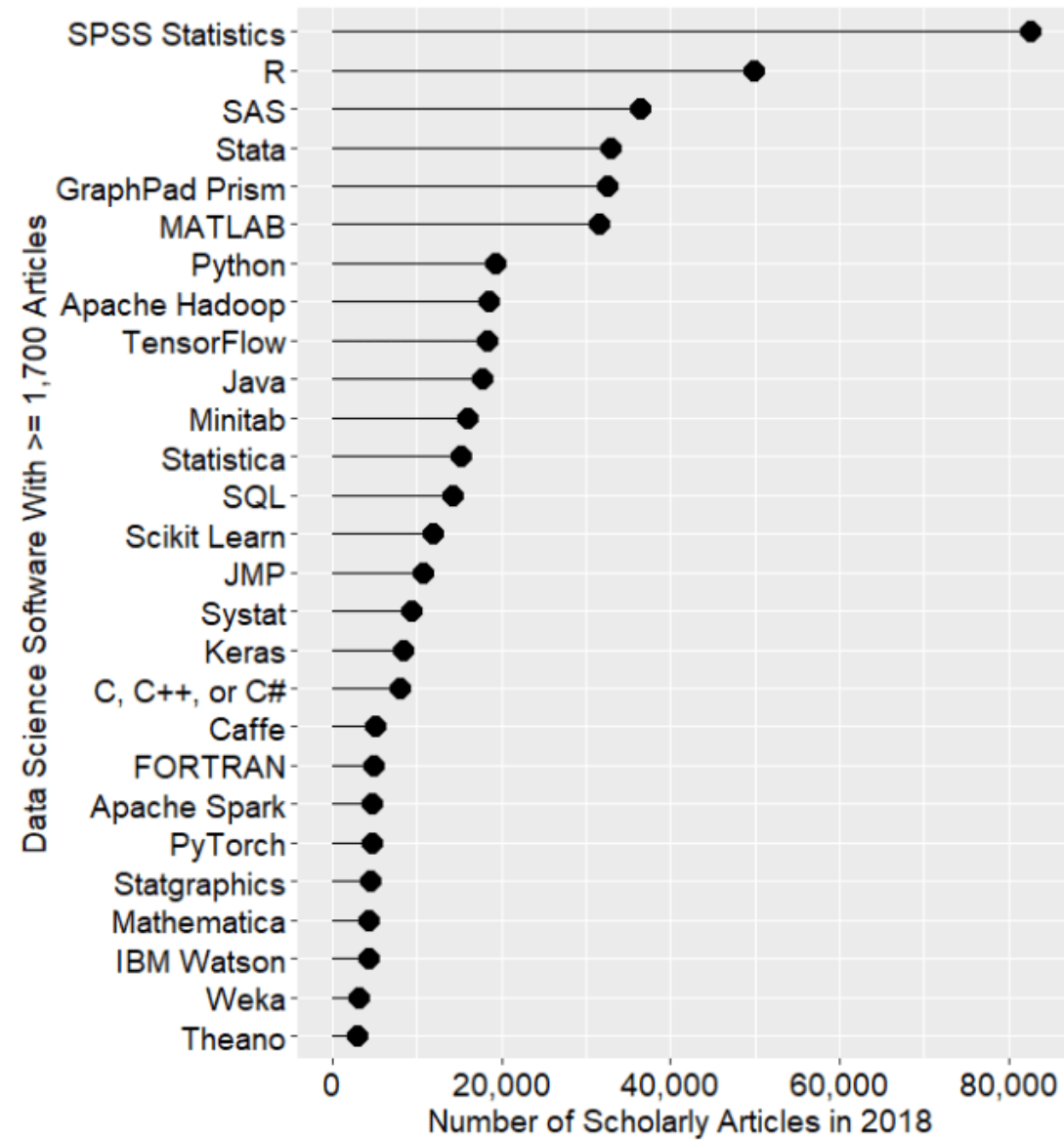
MATLAB

# Where are we?

cognitive model

statistics

computing

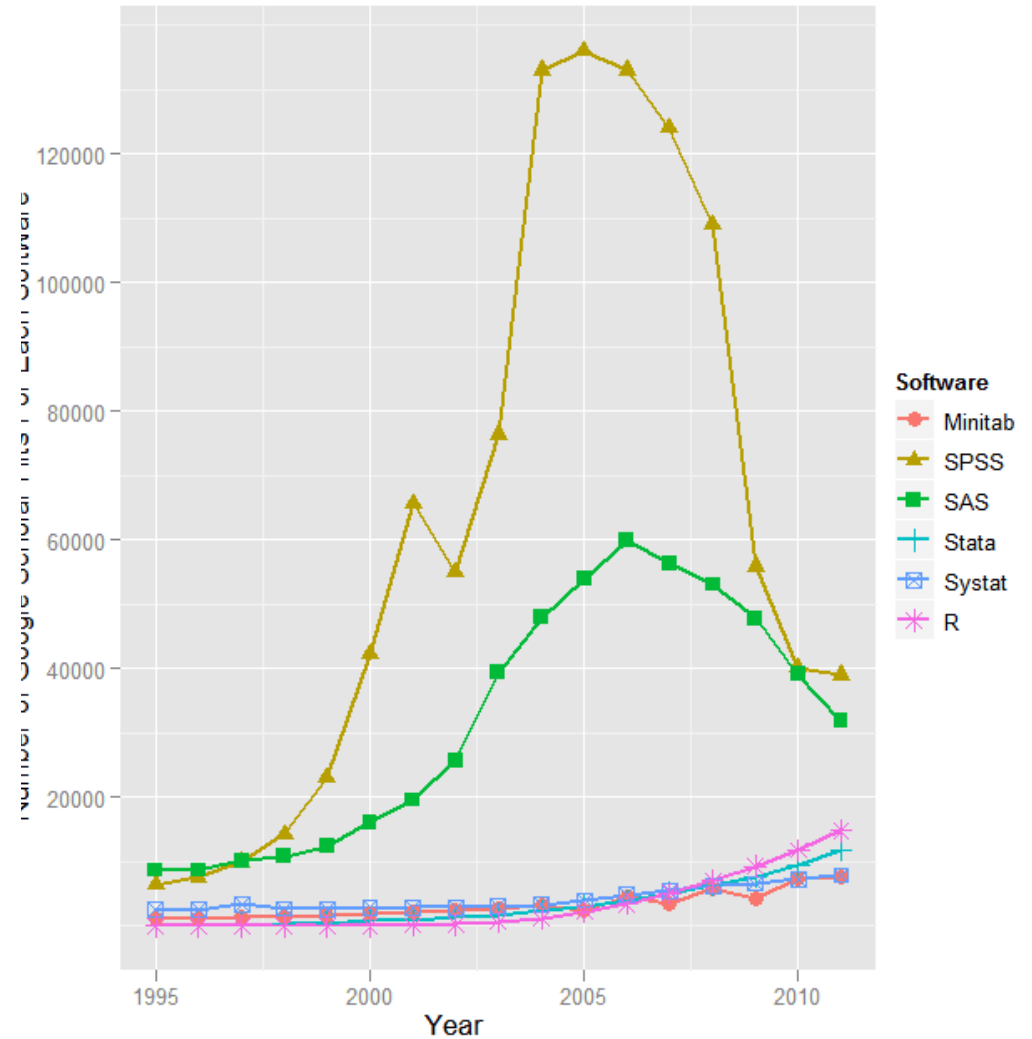
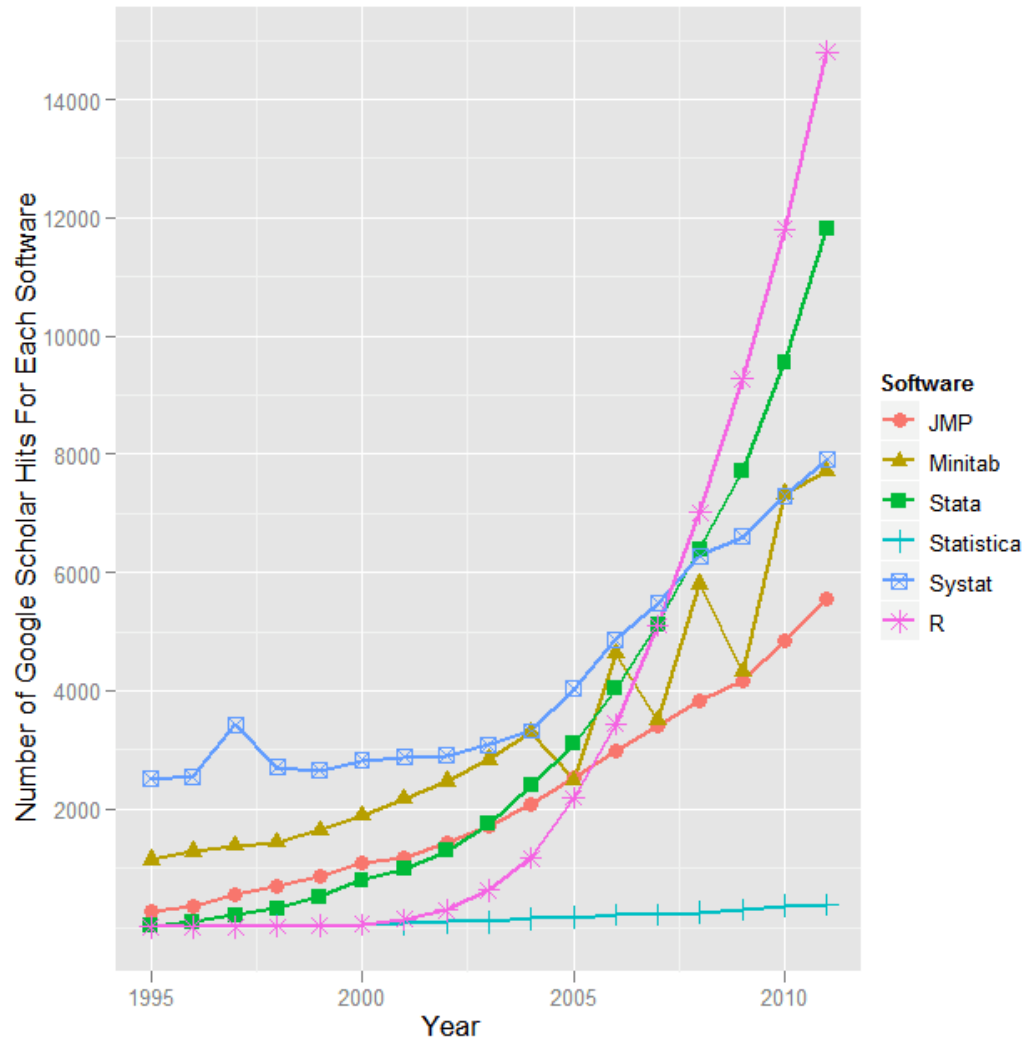


# Where are we?

cognitive model

statistics

computing

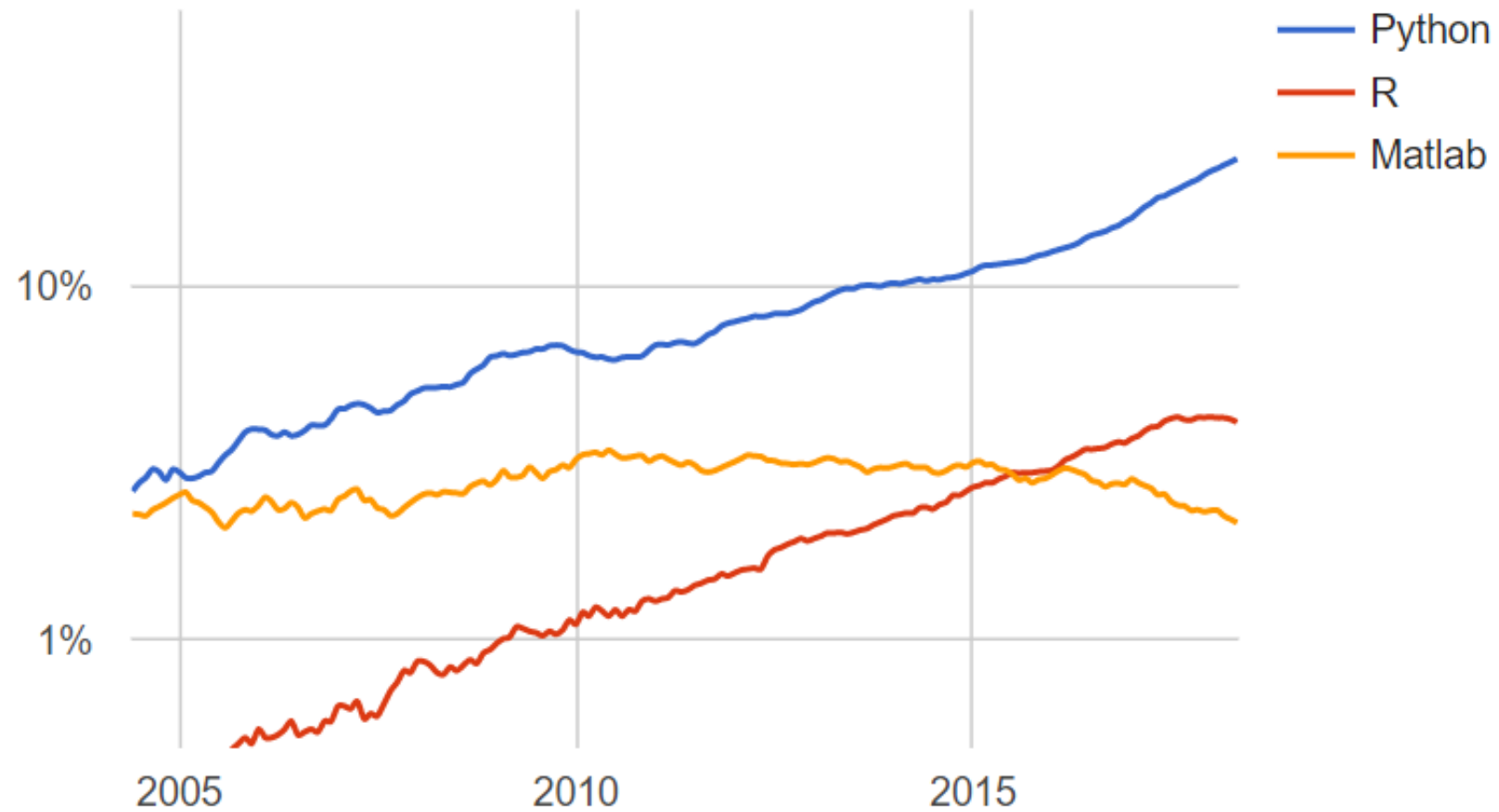


# Why should I care?

cognitive model

statistics

computing



# Why should I learn programming?

cognitive model

statistics

computing

- more flexible (type of analysis, type of visualization)
- facilitates reproducibility
- nearly default for cognitive (neuro)science
- possible to do everything all in one place
- good for career
- community is growing every day
- fun and rewarding to learn

# Do I need to make the choice?

cognitive model

statistics

computing



# Learn it all (if possible)

cognitive model

statistics

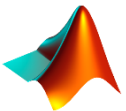
computing



- strong in statistics
- flexible way to run linear mixed-effects models (e.g., lme4)
- popular plotting library (e.g., ggplot2)
- packages available for psychological sciences (e.g., Afex)



- strong in machine learning (e.g., PyTorch, TensorFlow)
- flexible way to share scripts (e.g., Jupyter Notebook)
- packages available for neuroimaging/neuroscience



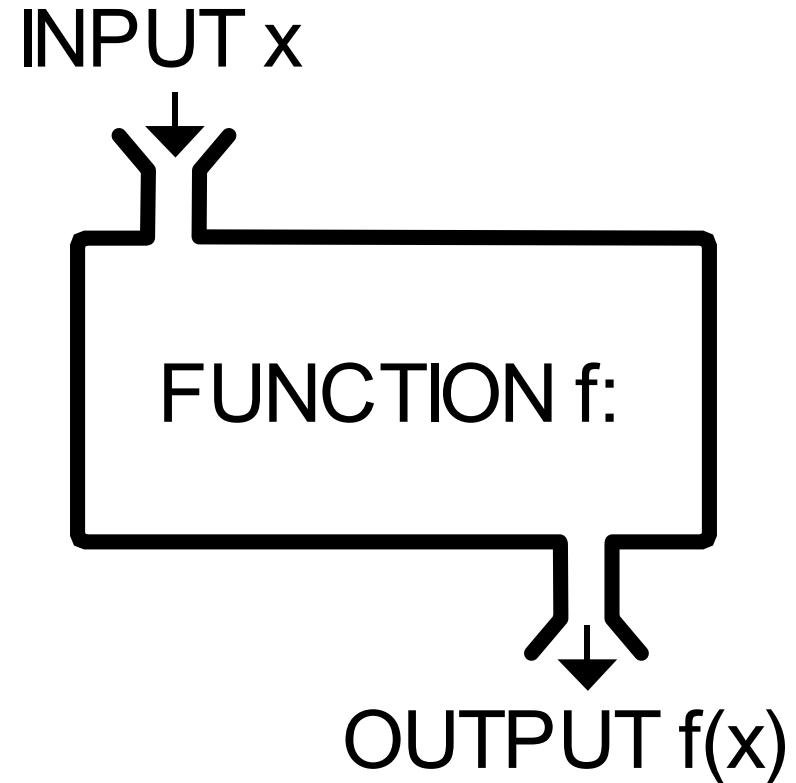
- strong in matrix calculation; detailed documentation
- commonly used platform for programming experiments (e.g., PsychToolBox)
- popular toolboxes for neuroimaging (e.g., SPM, EEGLab,)



# Functions

The operation(s) to obtain some quantity, based on another quantity.

- built-in functions
- external functions (packages)
- user-defined functions



# A walk-through example

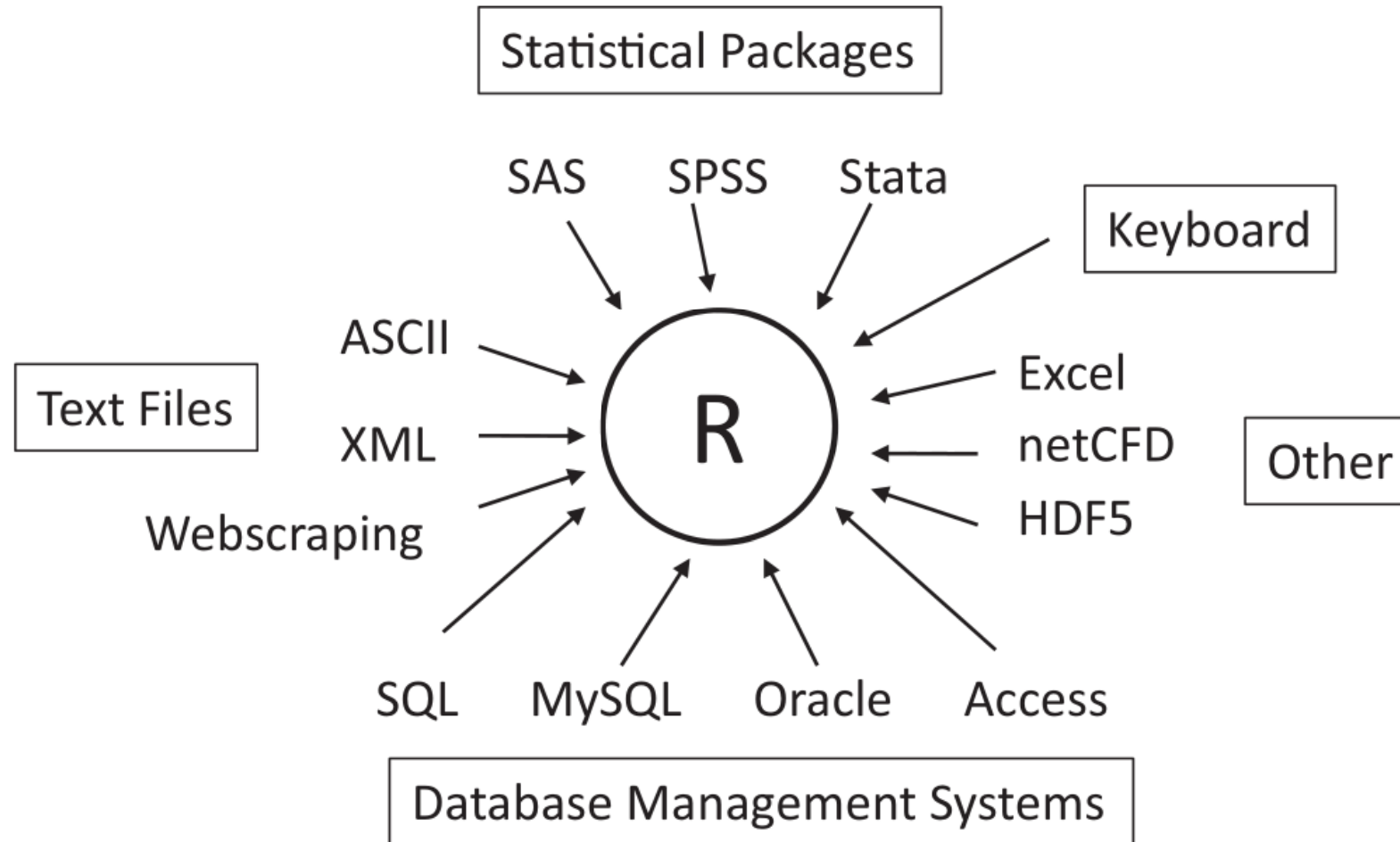
cognitive model

statistics

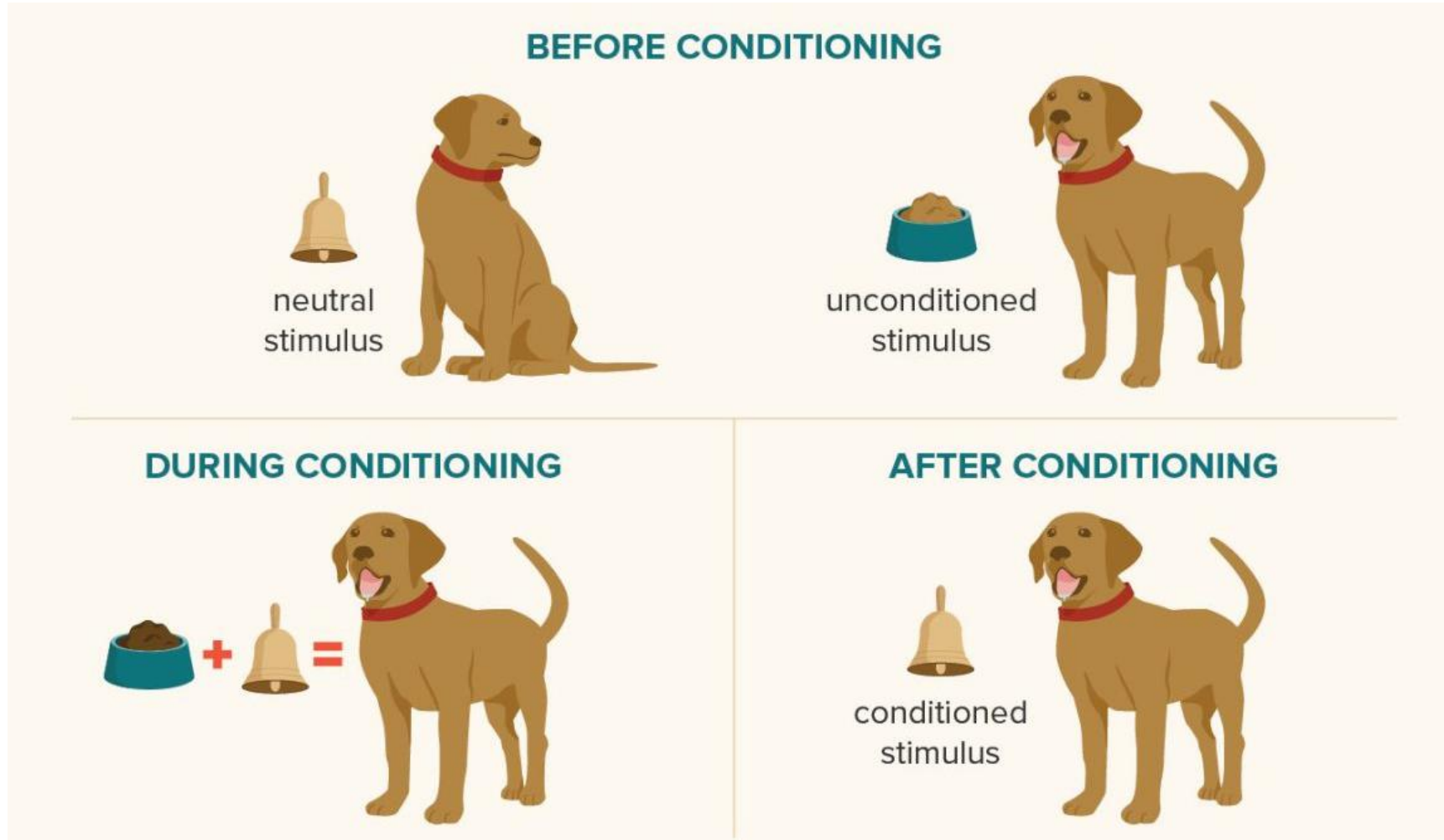
computing

- real-life problem in psychological science
- detailed walk-through in R
- brief demo in Python/Matlab
- Keep in mind: this is only to give you a rough idea of how each programming language looks like, in case you will encounter them in the future

# Data management

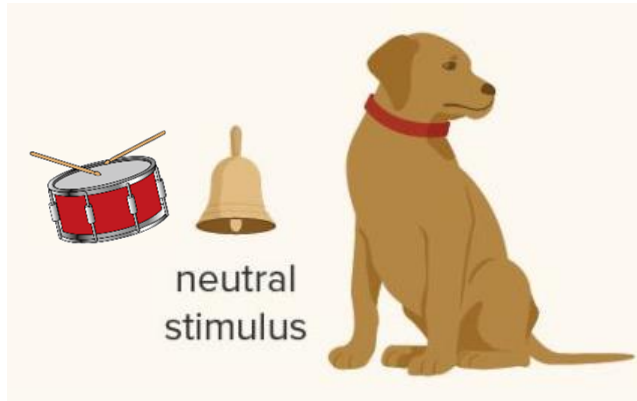


# Associative learning

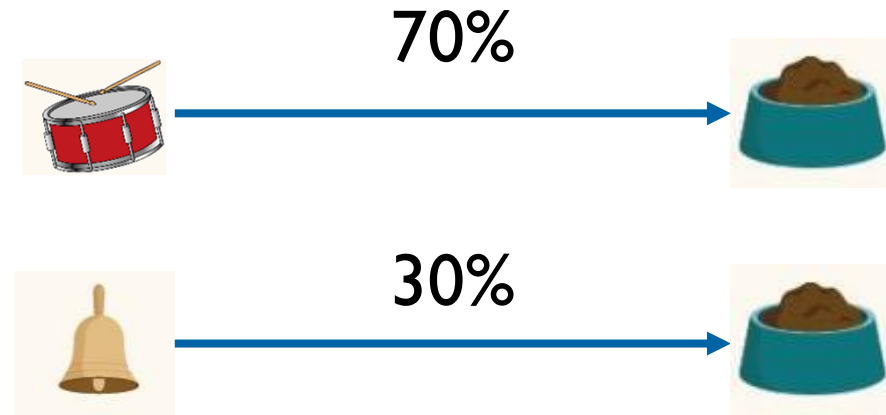


# Associative learning

pre-conditioning



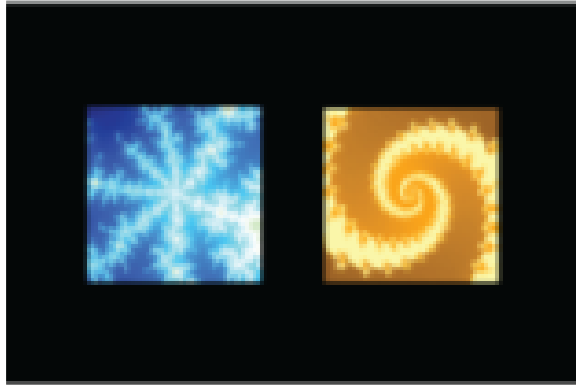
conditioning/training



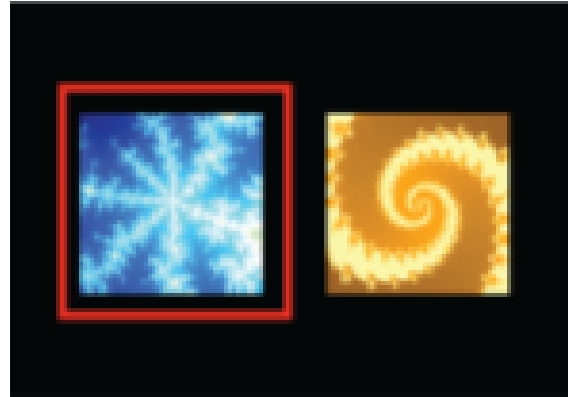
testing



# One simple experiment



choice  
presentation



action  
selection



outcome

reward contingency – 80:20

# The data

- nSub = 10
- nTrial = 80

./\_data/\_raw\_data/sub01/raw\_data\_sub01.txt

sub01
sub02
sub03
sub04
sub05
sub06
sub07
sub08
sub09
sub10

subjID	trialID	choice	outcome	correct
1	1	2	-1	1
1	2	1	1	1
1	3	1	1	1
1	4	1	1	1
1	5	2	-1	1
1	6	1	1	1
1	7	1	1	1
1	8	1	1	1
1	9	1	-1	1
1	10	2	-1	1
1	11	1	1	1
1	12	1	1	1
1	13	1	-1	2

# Import some data!

```
data_dir = ('_data/RL_raw_data/sub01/raw_data_sub01.txt')  
data = read.table(data_dir, header = T, sep = ",")  
head(data)
```

	subjID	trialID	choice	outcome	correct
1	1	1	1	1	1
2	1	2	1	1	1
3	1	3	1	1	1
4	1	4	NA	1	1
5	1	5	1	-1	1
6	1	6	2	-1	1



# Indexing

```
data[1,1]  
data[1,]  
data[,1]  
data[1:10,]  
data[,1:2]  
data[1:10, 1:2]  
data[c(1,3,5,6), c(2,4)]  
  
data$choice
```

```
> data  
  subjID trialID choice outcome correct  
1      1      1      1        1       1  
2      1      2      1        1       1  
3      1      3      1        1       1  
5      1      5      1       -1       1  
6      1      6      2       -1       1  
7      1      7      1        1       1  
8      1      8      1        1       1  
9      1      9      1        1       1  
10     1     10      1        1       1  
11     1     11      1        1       1
```

# Import some data!

```
data_dir = ('_data/RL_raw_data/sub01/raw_data_sub01.txt')  
data = read.table(data_dir, header = T, sep = ",")  
head(data)
```

	subjID	trialID	choice	outcome	correct
1	1	1	1	1	1
2	1	2	1	1	1
3	1	3	1	1	1
4	1	4	NA	1	1
5	1	5	1	-1	1
6	1	6	2	-1	1

```
sum(complete.cases(data)) # number of valid trials  
data = data[complete.cases(data),]  
dim(data[complete.cases(data),])
```

## Exercise III

.../01.R\_basics/\_scripts/R\_basics.R

### TASK:

write a for loop

... which reads in each participant's raw data

... and reshape it in the “long format” by subj

TIP: complete line 173

```
for ( j in 1:n ) {  
  read.table(file, header = T, sep = ",")  
}
```

subID	Choice
sub01	1
sub01	2
...	
sub02	2
sub02	2
...	
sub10	2
sub10	1

# Read all the data!

```
ns = 10
data_dir = '_data/RL_raw_data'

rawdata = c();
for (s in 1:ns) {
  sub_file = file.path(data_dir, sprintf('sub%02i/raw_data_sub%02i.txt',s,s))
  sub_data = read.table(sub_file, header = T, sep = ",")
  rawdata = rbind(rawdata, sub_data)
}
rawdata = rawdata[complete.cases(rawdata),]
rawdata$accuracy = (rawdata$choice == rawdata$correct) * 1.0

acc_mean = aggregate(rawdata$accuracy, by = list(rawdata$subjID), mean)[,2]
```



mean choice accuracy across trials, per participant.

# Basic stats

```
mean(acc_mean)
sd(acc_mean)
sem(acc_mean)
```

```
t.test(acc_mean, mu = 0.5) # one sample t-test
```

One Sample t-test

```
data: acc_mean
```

```
t = 13.788, df = 9, p-value = 2.34e-07
```

```
alternative hypothesis: true mean is not equal to 0.5
```

```
95 percent confidence interval:
```

```
0.6962988 0.7733565
```

```
sample estimates:
```

```
mean of x
```

```
0.7348277
```

```
> as.matrix(acc_mean, 10, 1)
      [,1]
[1,] 0.8076923
[2,] 0.7125000
[3,] 0.6875000
[4,] 0.6493506
[5,] 0.7750000
[6,] 0.7250000
[7,] 0.7662338
[8,] 0.8000000
[9,] 0.7500000
[10,] 0.6750000
```

# Basic correlation

```
load('_data/RL_descriptive.RData')
descriptive$acc = acc_mean
df = descriptive
```

```
cor.test(df$IQ, df$acc)
```

Pearson's product-moment correlation

```
data: df$IQ and df$acc
```

```
t = 4.8347, df = 8, p-value = 0.001297
```

```
alternative hypothesis: true correlation is not equal to 0
```

```
95 percent confidence interval:
```

```
0.5114810 0.9671586
```

```
sample estimates:
```

```
cor
```

```
0.8631401
```

```
> descriptive
  subjID      IQ      Age      acc
1      1 123.98691 31.07218 0.8125
2      2  87.63187 30.13800 0.7125
3      3  89.39930 23.44219 0.6875
4      4  84.34607 27.44848 0.6500
5      5 134.72208 23.30624 0.7750
6      6  84.60797 25.67858 0.7250
7      7 111.10238 24.36375 0.7750
8      8 117.89599 32.74026 0.8000
9      9  96.88233 22.80211 0.7500
10     10  76.01652 30.44258 0.6750
```

## Exercise IV

```
.../01.R_basics/_scripts/R_basics.R
```

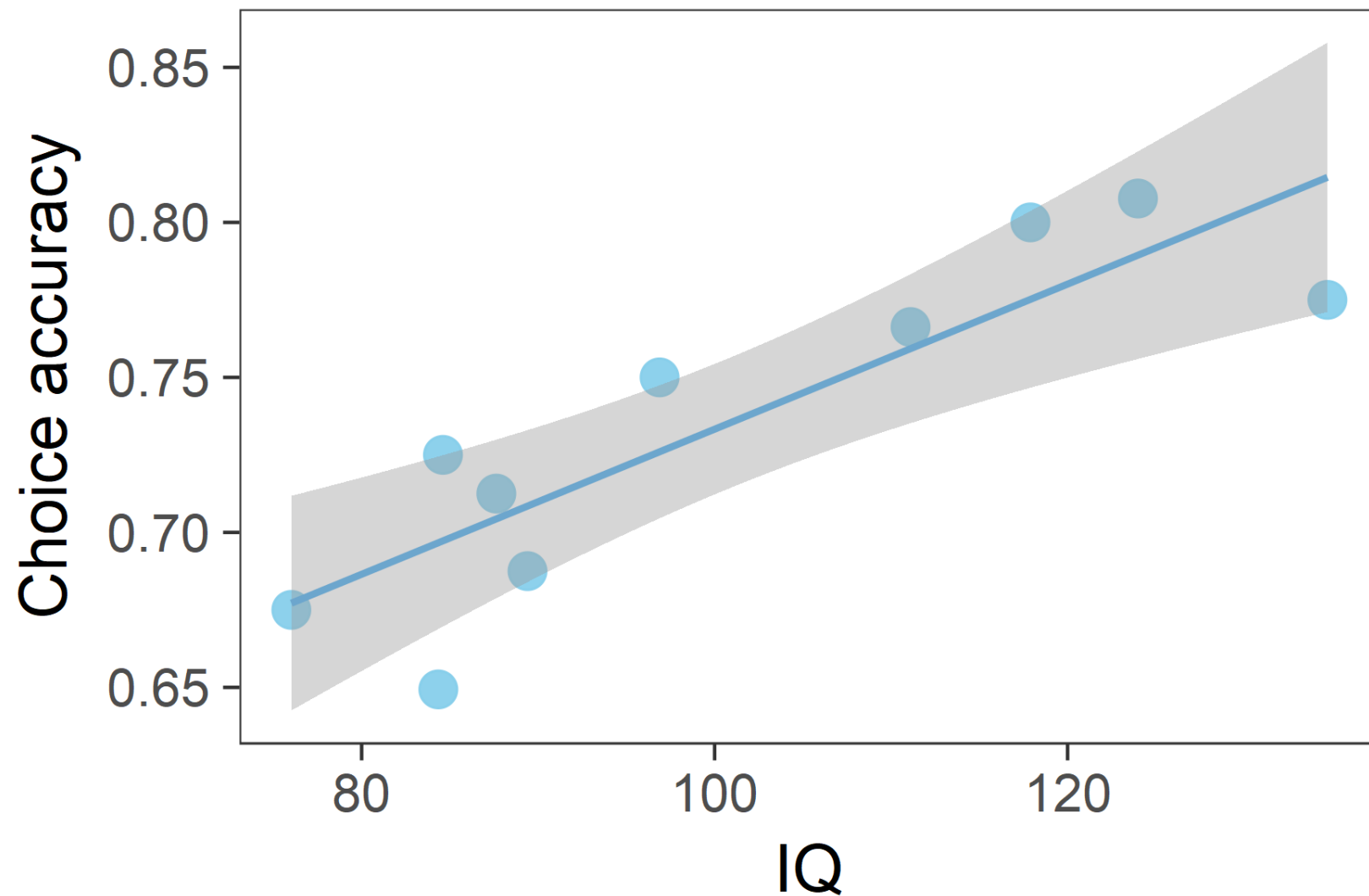
### TASK:

Read in the descriptive data: `_data/descriptive.RData`  
...include 'acc\_mean' as a new column, and  
...rename 'descriptive' as df.

Practice all the basic stats.

```
df$new_Col = new_Col
```

## A simple linear regression





# What is exactly the regression line in R?

```
fit1 = lm(acc ~ IQ, data = df)
summary(fit1)
```

```
Call:
lm(formula = acc ~ IQ, data = df)
```

```
Residuals:
```

	Min	1Q	Median	3Q	Max
	-0.047305	-0.016277	0.007562	0.022577	0.027731

```
Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	0.499292	0.049565	10.073	8.04e-06	***
IQ	0.002340	0.000484	4.835	0.0013	**

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.02885 on 8 degrees of freedom
```

```
Multiple R-squared:  0.745, Adjusted R-squared:  0.7131
```

```
F-statistic: 23.37 on 1 and 8 DF, p-value: 0.001297
```

$$\mu_i = \alpha + \beta x_i$$

$$y_i = \mu_i + \varepsilon$$

$$0.8631^2 = 0.7131$$

## Exercise V

```
.../01.R_basics/_scripts/R_basics.R
```

### TASK:

Read and make sense of the ggplot functions,  
... experiment make some adjustments (color marker size etc. ), and  
... run the `lm(acc ~ IQ)`

# BASICS OF R PROGRAMMING



# R Basics

cognitive model

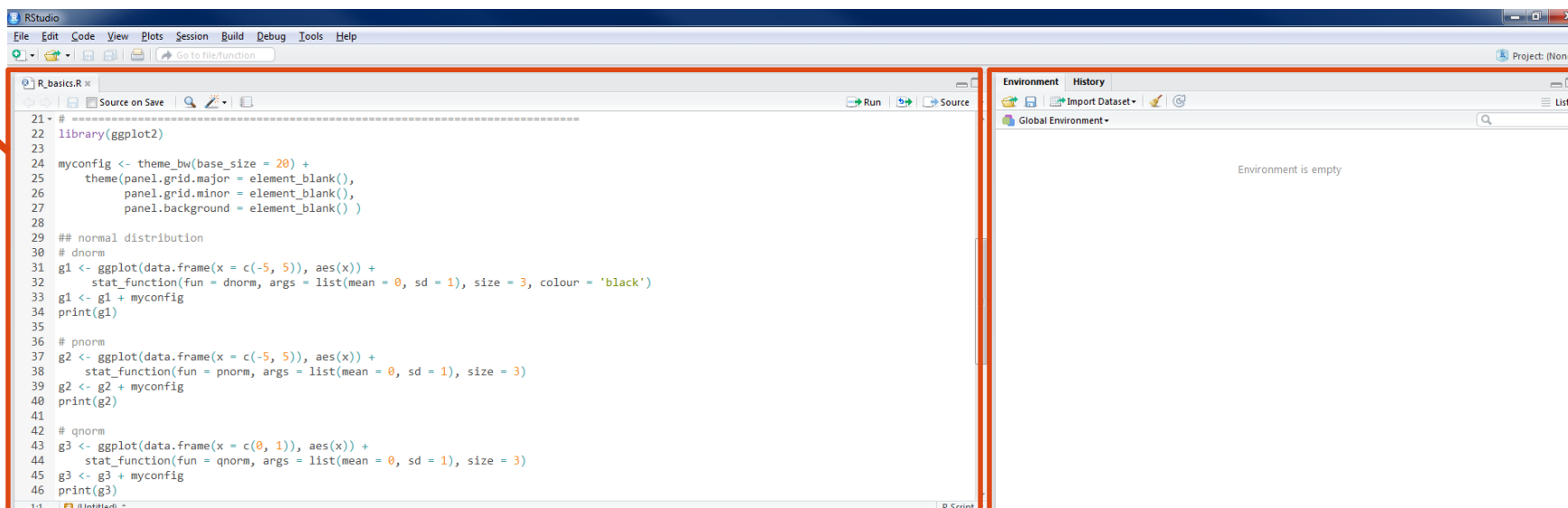
statistics

computing

- R
  - a programming language for statistical computing
  - R has its own user interface
  - freely available on Windows, Mac, and Linux
- R Studio
  - integrated development environment (IDE) for R
  - a more sophisticated R-friendly editor, with helpful syntax highlight



script editor

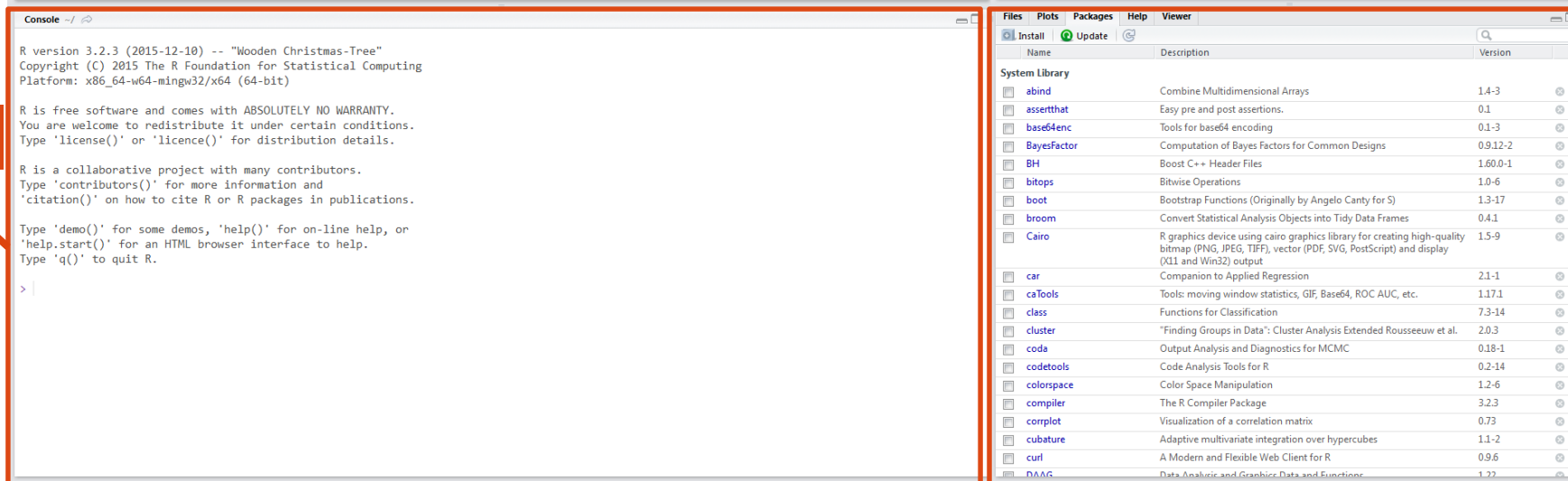


The image shows the RStudio interface. The top-left pane is the script editor, displaying R code for creating a ggplot2 theme and three plots (g1, g2, g3). The top-right pane is the Environment/History pane, which is currently empty, showing 'Global Environment' and 'Environment is empty'.

```
21 #  
22 library(ggplot2)  
23  
24 myconfig <- theme_bw(base_size = 20) +  
25   theme(panel.grid.major = element_blank(),  
26         panel.grid.minor = element_blank(),  
27         panel.background = element_blank())  
28  
29 ## normal distribution  
30 # dnorm  
31 g1 <- ggplot(data.frame(x = c(-5, 5)), aes(x)) +  
32   stat_function(fun = dnorm, args = list(mean = 0, sd = 1), size = 3, colour = 'black')  
33 g1 <- g1 + myconfig  
34 print(g1)  
35  
36 # pnorm  
37 g2 <- ggplot(data.frame(x = c(-5, 5)), aes(x)) +  
38   stat_function(fun = pnorm, args = list(mean = 0, sd = 1), size = 3)  
39 g2 <- g2 + myconfig  
40 print(g2)  
41  
42 # qnorm  
43 g3 <- ggplot(data.frame(x = c(0, 1)), aes(x)) +  
44   stat_function(fun = qnorm, args = list(mean = 0, sd = 1), size = 3)  
45 g3 <- g3 + myconfig  
46 print(g3)
```

environment/  
command history

console



The image shows the RStudio interface. The bottom-left pane is the console, displaying the R version information and welcome message. The bottom-right pane is the Packages pane, showing a list of installed and available packages.

R version 3.2.3 (2015-12-10) -- "Wooden Christmas-Tree"  
Copyright (C) 2015 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.  
You are welcome to redistribute it under certain conditions.  
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.  
Type 'contributors()' for more information and  
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or  
'help.start()' for an HTML browser interface to help.  
Type 'q()' to quit R.

> |

Name	Description	Version
<b>System Library</b>		
abind	Combine Multidimensional Arrays	1.4-3
assertthat	Easy pre and post assertions.	0.1
base64enc	Tools for base64 encoding	0.1-3
BayesFactor	Computation of Bayes Factors for Common Designs	0.9.12-2
BH	Boost C++ Header Files	1.60.0-1
bitops	Bitwise Operations	1.0-6
boot	Bootstrap Functions (Originally by Angelo Canty for S)	1.3-17
broom	Convert Statistical Analysis Objects into Tidy Data Frames	0.4.1
Cairo	R graphics device using cairo graphics library for creating high-quality bitmap (PNG, JPEG, TIFF), vector (PDF, SVG, PostScript) and display (X11 and Win32) output	1.5-9
car	Companion to Applied Regression	2.1-1
caTools	Tools: moving window statistics, GIF, Base64, ROC AUC, etc.	1.17.1
class	Functions for Classification	7.3-14
cluster	"Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.	2.0.3
coda	Output Analysis and Diagnostics for MCMC	0.18-1
codetools	Code Analysis Tools for R	0.2-14
colorspace	Color Space Manipulation	1.2-6
compiler	The R Compiler Package	3.2.3
corplot	Visualization of a correlation matrix	0.73
cubature	Adaptive multivariate integration over hypercubes	1.1-2
curl	A Modern and Flexible Web Client for R	0.9.6
DAAG	Data Analysis and Graphics: Data and Functions	1.22

file/pkg/img/  
etc.

# Know your R

```
>R.version
```

```
platform      _  
arch          x86_64-w64-mingw32  
os            mingw32  
system        x86_64, mingw32  
status  
major         3  
minor         5.1  
year          2018  
month         07  
day           02  
svn rev       74947  
language      R  
version.string R version 3.5.1 (2018-07-02)  
nickname      Feather Spray
```

# R Console as a Calculator

cognitive model

statistics

computing

## Addition and Subtraction

```
> 3+2  
[1] 5
```

```
> 3-2  
[1] 1
```

## Multiplication and Division

```
> 3*2  
[1] 6
```

```
> 3/2  
[1] 1.5
```

## Exponents in R

```
> 3^2  
[1] 9
```

```
> 2^3  
[1] 8
```

## Constants in R

```
> pi  
[1] 3.141593
```

```
> exp(1)    base of the natural logarithm  
[1] 2.718282
```

# Special values

## Infinite Values

```
> Inf  
[1] Inf
```

```
> 1+Inf  
[1] Inf
```

## Machine Epsilon

```
> .Machine$double.eps  
[1] 2.220446e-16
```

```
> 0>.Machine$double.eps  
[1] FALSE
```

## Empty Values

```
> NULL  
NULL
```

```
> 1+NULL  
numeric(0)
```

## Missing Values

```
> NA  
[1] NA
```

```
> 1+NA  
[1] NA
```



# Storing and manipulating variables

Define objects `x` and `y` with values of 3 and 2, respectively:

```
> x=3
```

```
> y=2
```

Some calculations with the defined objects `x` and `y`:

```
> x+y
```

```
[1] 5
```

```
> x*y
```

```
[1] 6
```

Warning: R is case sensitive, so `x` and `X` are not the same object.

# Basic R functions

## Combine

```
> c(1, 3, -2)
[1] 1 3 -2
```

```
> c("a", "a", "b", "b", "a")
[1] "a" "a" "b" "b" "a"
```

## Sum and Mean

```
> sum(c(1, 3, -2))
[1] 2
```

```
> mean(c(1, 3, -2))
[1] 0.6666667
```

## Variance and Std. Dev.

```
> var(c(1, 3, -2))
[1] 6.333333
```

```
> sd(c(1, 3, -2))
[1] 2.516611
```

## Minimum and Maximum

```
> min(c(1, 3, -2))
[1] -2
```

```
> max(c(1, 3, -2))
[1] 3
```

# Basic R functions (cont.)

Define objects `x` and `y`:

```
> x=c(1,3,4,6,8)
```

```
> y=c(2,3,5,7,9)
```

Calculate the correlation:

```
> cor(x,y)
```

```
[1] 0.988765
```

Calculate the covariance:

```
> cov(x,y)
```

```
[1] 7.65
```

Combine as columns

```
> cbind(x,y)
```

	x	y
[1,]	1	2
[2,]	3	3
[3,]	4	5
[4,]	6	7
[5,]	8	9

Combine as rows

```
> rbind(x,y)
```

	[,1]	[,2]	[,3]	[,4]	[,5]
x	1	3	4	6	8
y	2	3	5	7	9

# Basic Commands

```
getwd()
setwd('E:/teaching/BayesCog_Wien/')
dir() # folders/files in the wd
ls()  # anything in the environment/workspace
print('Hello World!')
cat('Hello', 'World!')
paste0('C:/', 'Group1')
help(func)
? func # and Google!
a <- 5
a = 5
head(d) # first 6 entries
tail(d) # last 6 entries
save(varname, file = "pathname/varname.RData")
load("pathname/varname.RData")
rm(list = ls())
q()
```

# RStudio - Shortcuts

cognitive model

statistics

computing

Ctrl + L: clean console

Ctrl + Shift + N: create a new script

↑: command history

Ctrl(hold) + ↑: command history with certain starts

Ctrl + Enter: execute selected codes (in a script)

# Editor (WIN general) - Shortcuts

cognitive model

statistics

computing

Ctrl + home/Pos1: go to the very top of a script

Ctrl + end/Ende: go to the very end of a script

Shift(hold) + ↑/↓: select line(s)

Ctrl(hold) + ←/→: select word(s)

# Data Classes

numeric: 1.1 2.0

integer: 1 2 3

character / string: "hello world!"

logical: TRUE FALSE

factors: "male" / "female"

(complex: 1+2i)

# Data Types

cognitive model

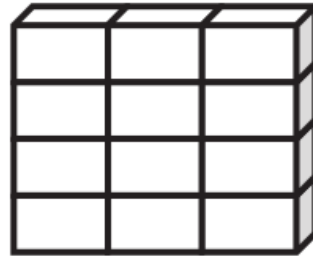
statistics

computing

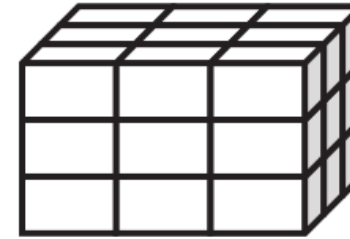
(a) Vector



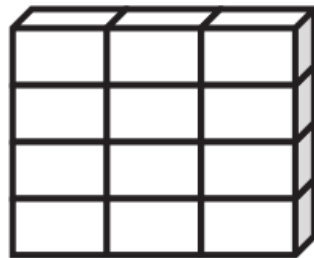
(b) Matrix



(c) Array

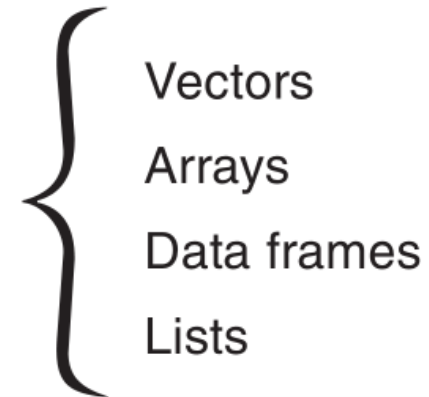


(d) Data frame



**Columns can be different modes**

(e) List





# Exercise I

cognitive model

statistics

computing

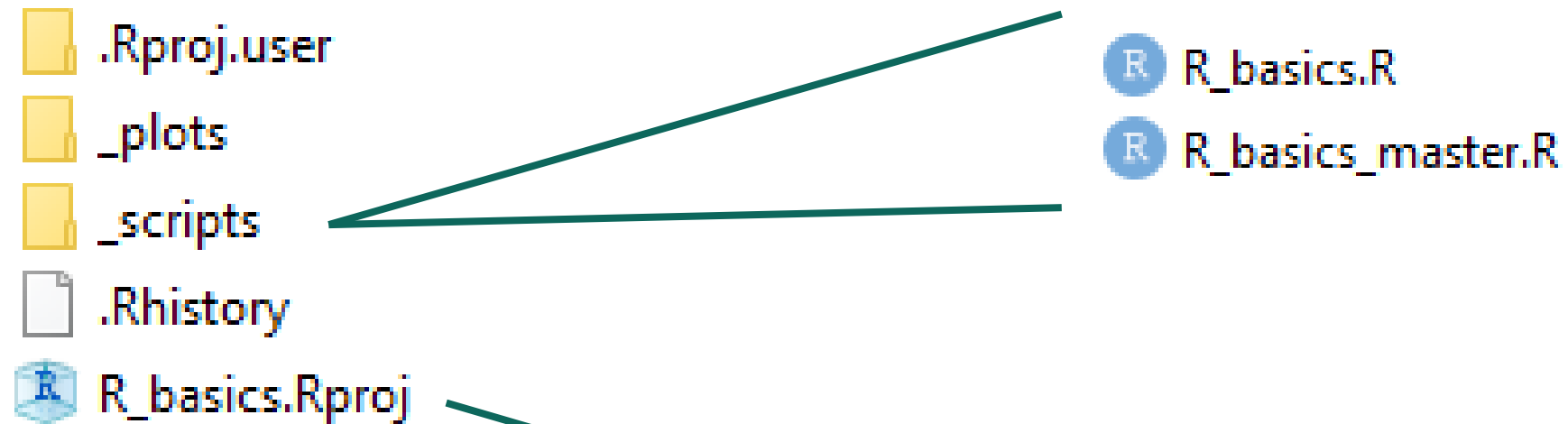
```
.../01.R_basics/_scripts/R_basics.R
```

up to “Control Flow”

TASK: practise basic R commands and data type

TIP: `class()`, `str()`

## Side note: folder structure



click this to start each exercise,  
then no need to set directory

# Logical Operators

Operator	Summary
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not equal to
!x	NOT x
x y	x OR y
x&y	x AND y

# Control Flow

- if-else

```
if (cond) {  
    ..statement..  
}
```

```
if (cond) {  
    ..statement..  
} else {  
    ..statement..  
}
```

```
if (cond) {  
    ..statement..  
} else if (cond) {  
    ..statement..  
} else {  
    ..statement..  
}
```

- for-loop

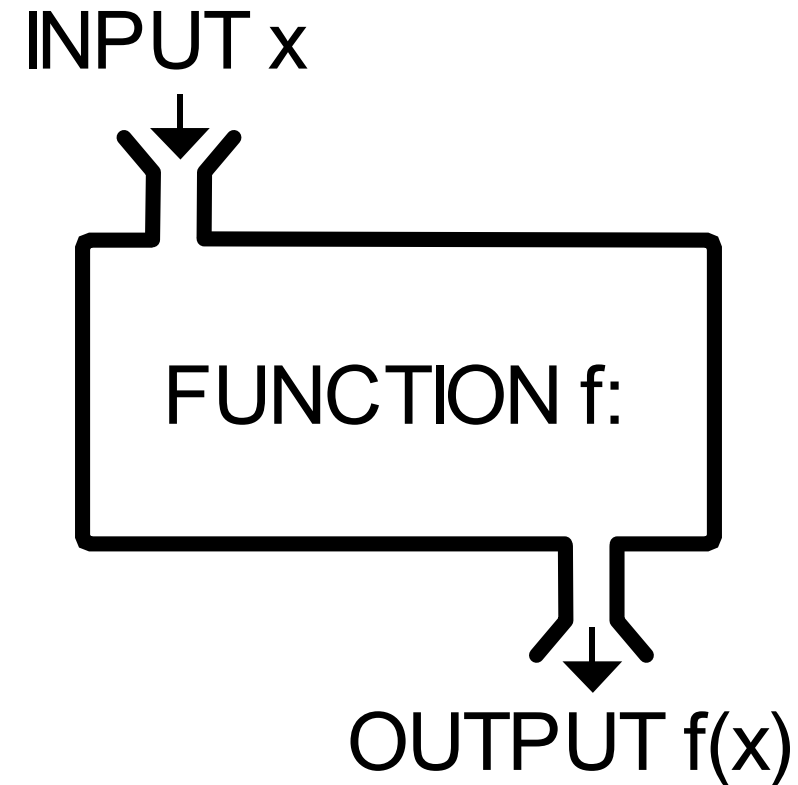
```
for ( j in 1:J ) {  
    ..statement..  
}
```

```
for ( j in 1:J ) {  
    for ( k in 1:K ) {  
        ..statement..  
    }  
}
```

# Functions

The operation(s) to obtain some quantity, based on another quantity.

- built-in functions
- external functions (packages)
- user-defined functions



# User-defined Function

```
funname <- function (input_args) {  
  .. function body ..  
  .. function body ..  
  return(output_args)  
}
```

$$sem = \sqrt{\frac{s^2}{n-1}}$$

```
sem <- function(x) {  
  sqrt( var(x,na.rm=TRUE) / (length(na.omit(x))-1) )  
}
```

# Exercise II

cognitive model

statistics

computing

```
.../01.R_basics/_scripts/R_basics.R
```

TASK: practise control flow and user-defined function

## Exercise II

- Generate a random number between 0 and 1
- Compare it against  $1/3$  and  $2/3$
- Print the random number and its position relative to  $1/3$  and  $2/3$ .

```
# if-else
t <- runif(1) # random number between 0 and 1
if (t <= 1/3) {
  cat("t =", , ", t <= 1/3. \n")
} else if () {
  cat("t =", t, ", t > 2/3. \n")
} else {
  cat("t =", t, ", 1/3 < t <= 2/3. \n")
}
```

Example outcome:

t = 0.895 , t > 2/3.

- Get the name of each month
- Print it one by one

```
# for-loop
month_name <- format(ISOdate(2018,1:12,1), "%B")
for (j in 1:length(month_name)) {
  cat()
}
```

```
The month is January
The month is February
The month is March
The month is April
The month is May
The month is June
The month is July
The month is August
The month is September
The month is October
The month is November
The month is December
```



# Packages in R

R packages are collections of functions and data sets developed by the community, to make your life a lot easier!

```
install.packages('ggplot2')  
library(ggplot2)  
detach('package:ggplot2')
```

# Visualization

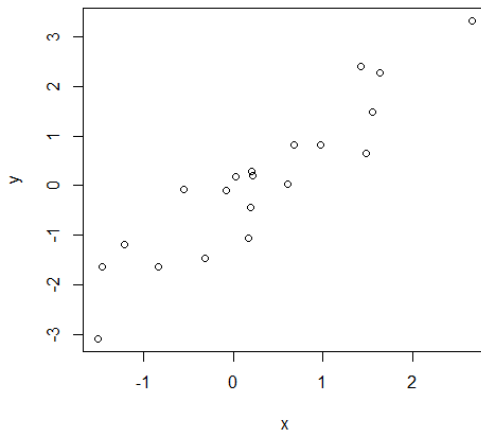
cognitive model

statistics

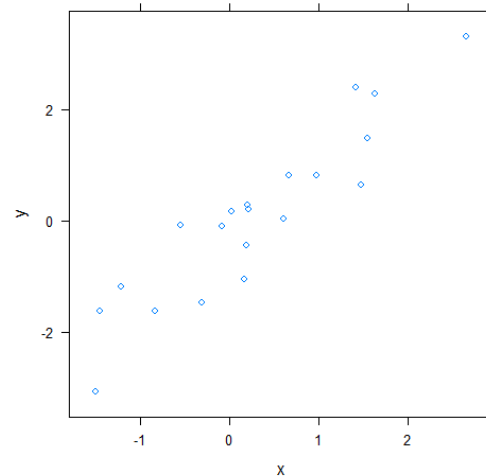
computing

- **built-in** plotting functions – first attempt / quick look / exploratory
- **{lattice}** – making nicer, similar to basic plotting functions (takes lm formulae)
- **{ggplot2}** – making nicer, a layering philosophy

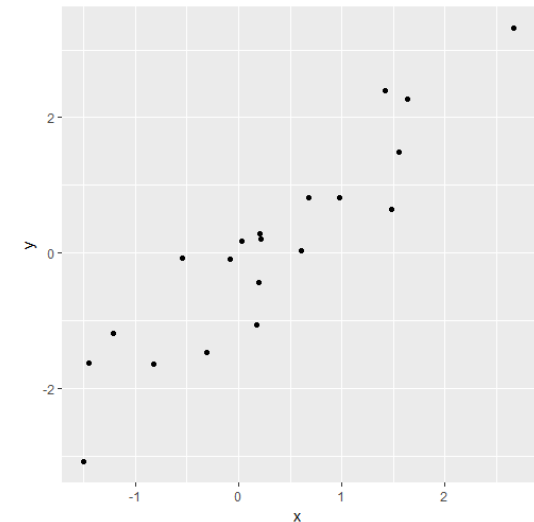
`plot(x,y)`



`lattice::xyplot(y~x)`



`ggplot2::qplot(x,y)`



# Brief Intro to **ggplot2**

cognitive model

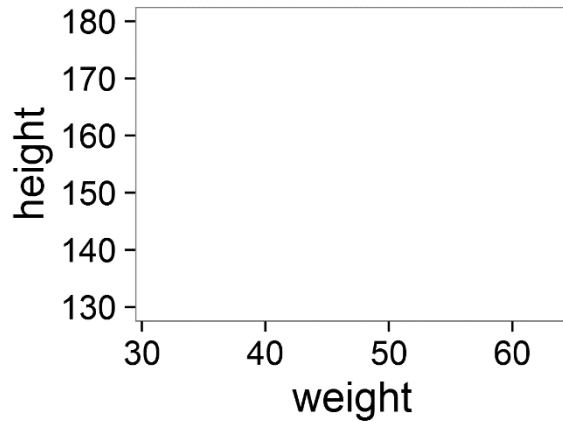
statistics

computing

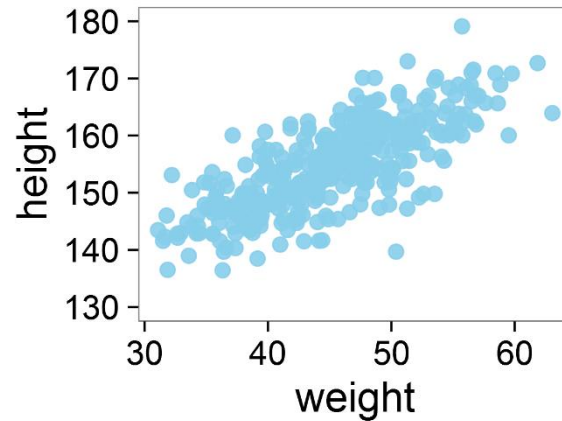
`plot` = `geometric` (points, lines, bars) + `aesthetic` (color, shape, size)

game of adding layers!

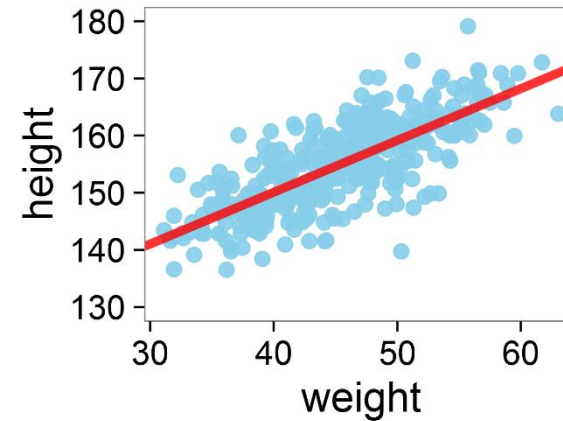
background



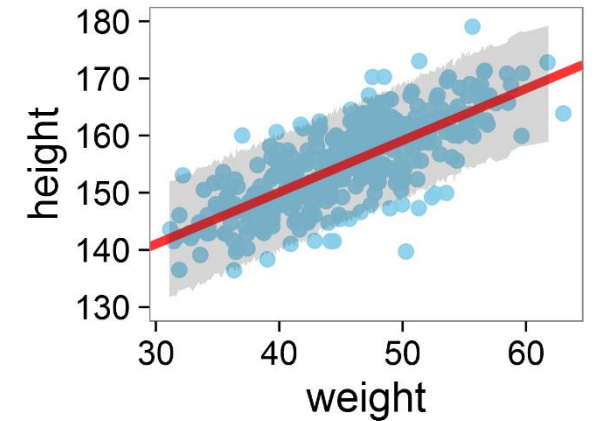
add scatters



add regression line



add uncertainty

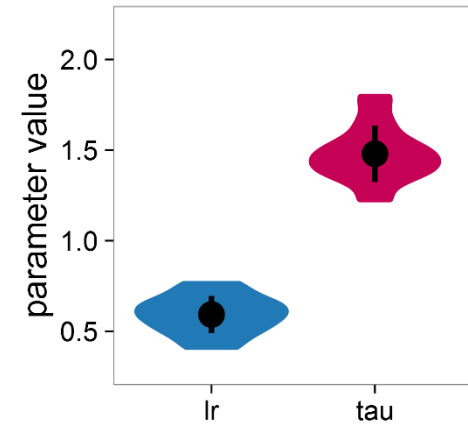
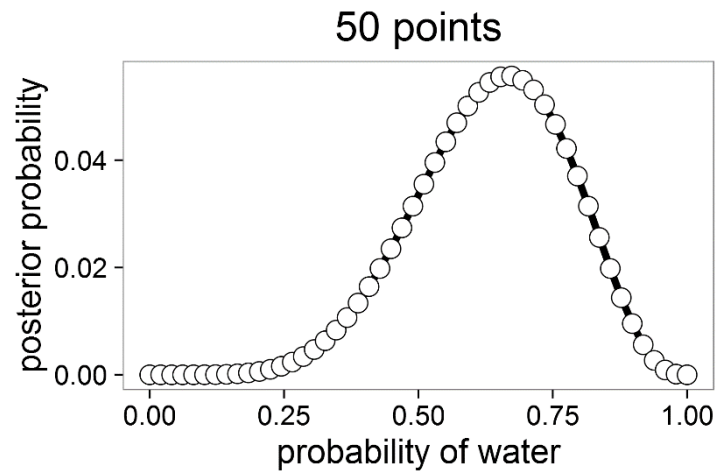
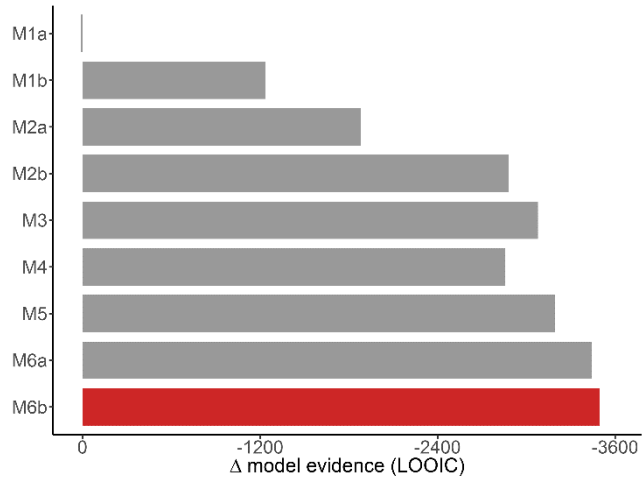
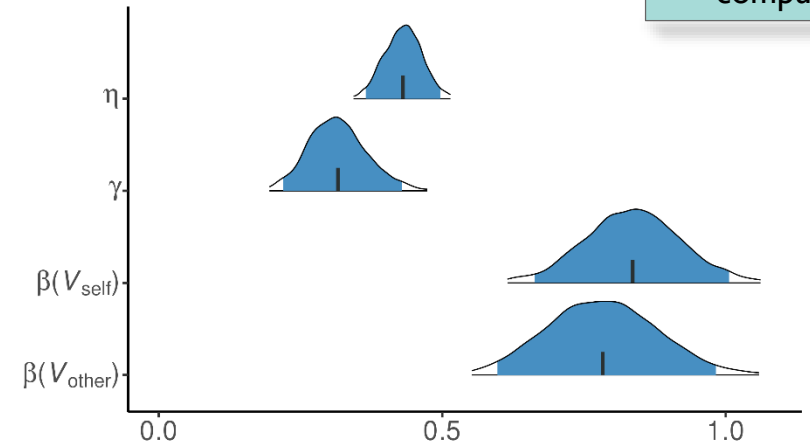
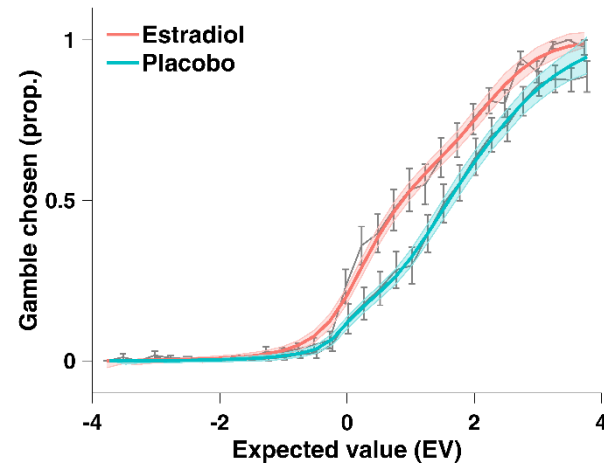
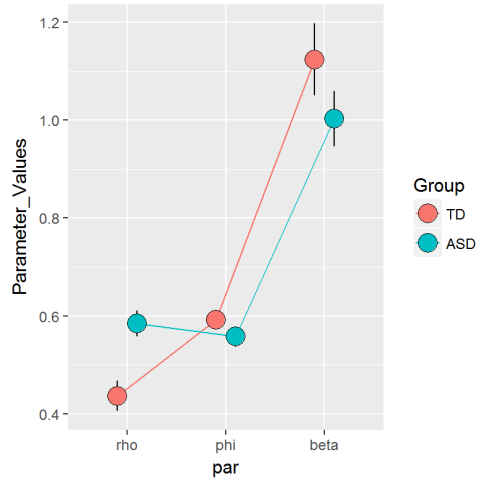


# A taste of ggplot2

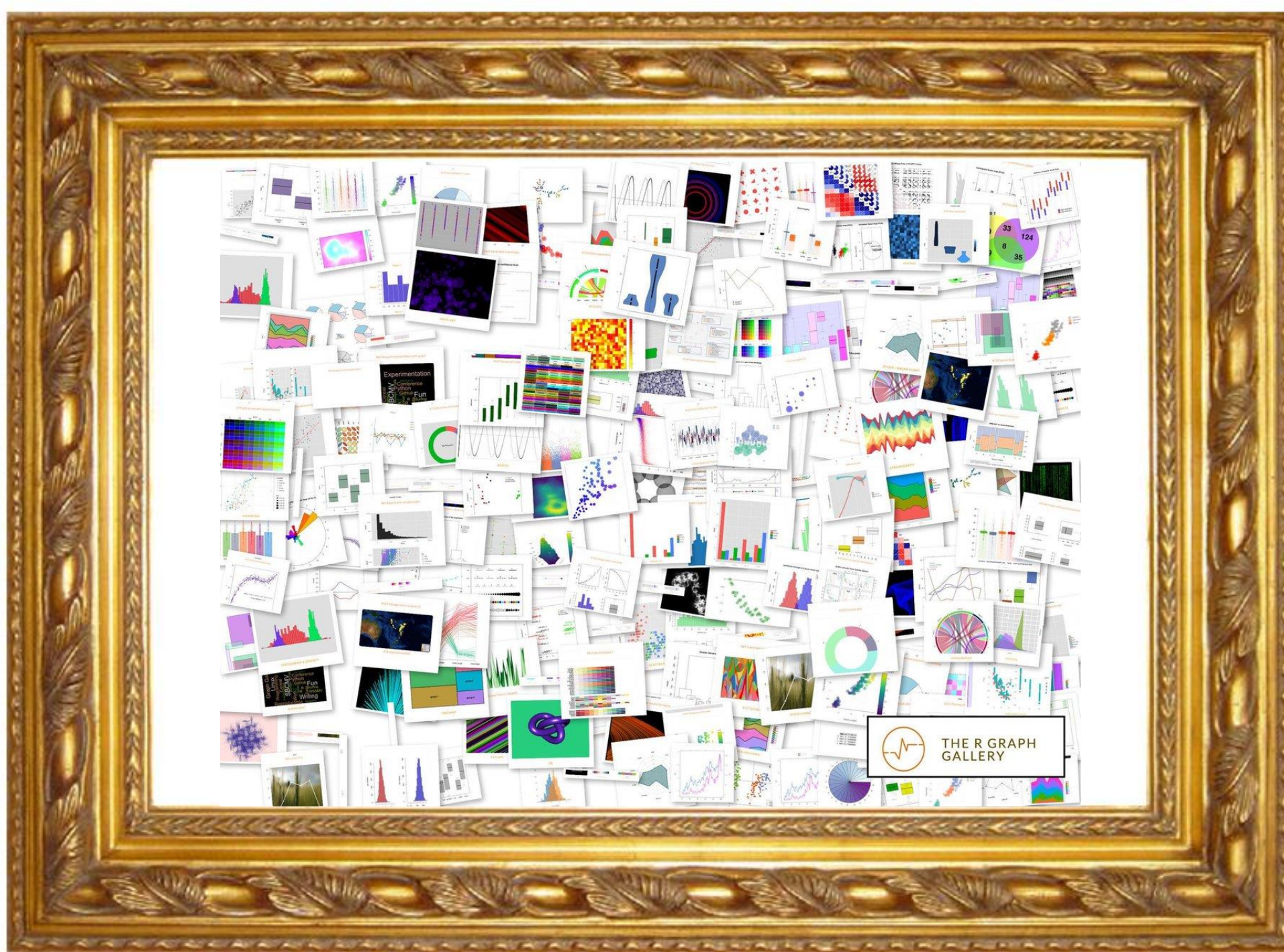
cognitive model

statistics

computing







<https://www.r-graph-gallery.com/>



ANY  
QUESTIONS  
?

Happy Computing!