PRINT ('HELLO WORLD')

A SHORT INTRODUCTION TO SOFTWARE DEVELOPMENT IN NEUROSCIENCE

Created by Heiko Borchers with reveal.js

SUMMARY

INTRODUCTION

ABOUT ME

- Bachelor student at the University of Bonn in computer science
- Focus on network communication and IT-security
- About three years now working for CENs/BonnEconLab

MOST USED LANGUAGES

- SPSS
- R
- Python
- MatLab
- Presentation
- zTree

LANGUAGE DETAILS

SPSS

Short for **S**tatistical **P**ackage for the **S**ocial **S**ciences

Used for statistical analysis and data mining

```
* create a new data file that just has "x1" in it from 1 to 20 by 1.
input program.
        loop #i = 1 to 30 by 1.
        compute x1 = #i.
        end case.
end loop.
end file.
end input program.
execute.

* fill in logistic equation below .
* say equation is -3 + .3*x1 + .1 * x2.
* and x2 has a mean of 5.
compute ylog = -3 + .3*x1 + 5*.1   .
compute py = 1 - 1/(1 + exp(ylog)) .
execute.

* show graph with prob y on y axis, and x1 on x axis.
```

R

Programming Language for statistic computing
Comparebale to MATLAB and SPSS but free software

```r
install.packages('neuralnet')
library("neuralnet")

#Going to create a neural network to perform sqare rooting
#Type ?neuralnet for more information on the neuralnet library

#Generate 50 random numbers uniformly distributed between 0 and 100
#And store them as a dataframe
traininginput <-  as.data.frame(runif(50, min=0, max=100))
trainingoutput <- sqrt(traininginput)

#Column bind the data into one variable
trainingdata <- cbind(traininginput,trainingoutput)
colnames(trainingdata) <- c("Input","Output")

#Train the neural network
#Going to have 10 hidden layers
#Threshold is a numeric value specifying the threshold for the partial
```

PYTHON

General purpose programming language
Used for almost anything from creating experiments to data evaluation

```
my_list = ['banana', 'strawberry', 'apple', 'melon', 'peach'] #a simple
sorted_list = sorted(my_list) #python includes powerful sorting algorith
for x in range(1,11,1): #a for loop which counts to ten
"""range takes up to three arguments: the start, which is
inclusive, the end, which is exclusive and the step size"""
        print(x)
print() #prints an empty line
for y in range(10,0,-1): #the step size can be negative to count backwar
        print(y)
print()
for z in range(len(sorted_list)): #you can also iterate over lists
        print (sorted_list[z]) #prints the list
```

MATLAB

Software to analyze (fMRI) data

```
% the following code models a passive neuronal membrance
% as RC-circuit.
% Note: in a membrane model, the resistor and capacitor are
% in parallel.

% this code demonstrates how a membrance responds to a constant
% current input that is turned on for a fixed time interval
% and then turned off.

% Charging and discharging curves for passive membrane patch
% R Rao 2007

clear
% input current
I = 10 % nA

% capacitance and leak resistance
```

ZTREE

Windows software to conduct economic experiments
Mostly used in the BonnEconLab

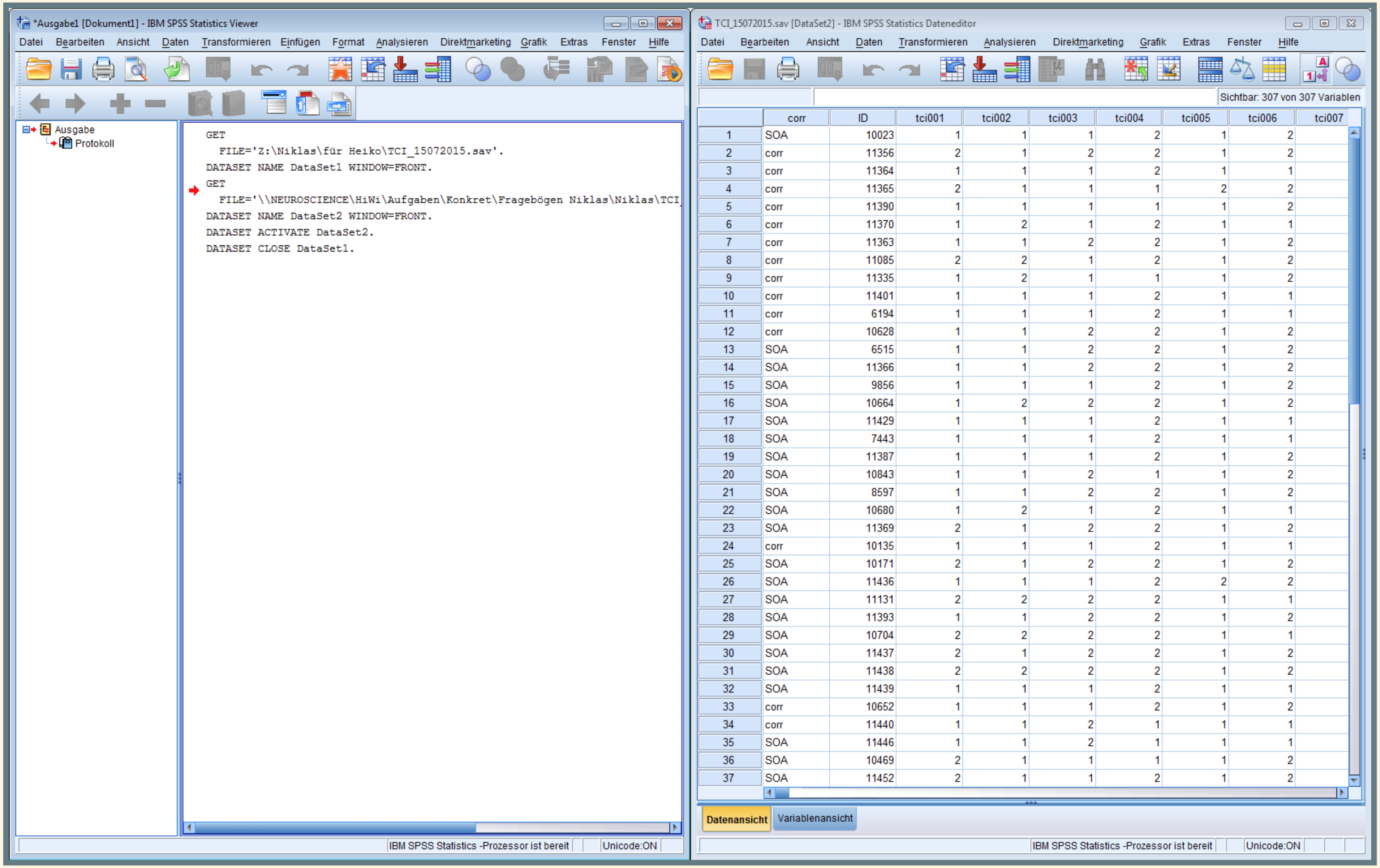
```
//Auswertung Manie
if (d4 == 1){
if (d1b == 1 | d2b == 1) {
if (d3a + d3b + d3c + d3d + d3e + d3f + d3g >= 3) {
hypomanische_episode_aktuell = 1;
LeaveStage = 1 ;
}}}

if (d4 == 1){
if (d1b == 0 & d2b ==0 ){
if (d3a + d3b + d3c + d3d + d3e + d3f + d3g >= 4){
hypomanische_episode_frueher = 1;
LeaveStage = 1 ;
}}}
/*
TODO: Manische Episoden
*/
```

PROGRAMMING ENVIRONMENTS

Most Software comes with its own Programming environment called IDE

IDE stands for **I**ntegrated **D**evelopment **E**nvironment

SPSS


*Ausgabe1 [Dokument1] - IBM SPSS Statistics Viewer
Datei Bearbeiten Ansicht Daten Transformieren Einfügen Format Analysieren Direktmarketing Grafik Extras Fenster Hilfe
Ausgabe
Protokoll
TCI_15072015.sav [DataSet2] - IBM SPSS Statistics Dateneditor
Datei Bearbeiten Ansicht Daten Transformieren Analysieren Direktmarketing Grafik Extras Fenster Hilfe
Sichtbar: 307 von 307 Variablen
Datenansicht
Variablenansicht

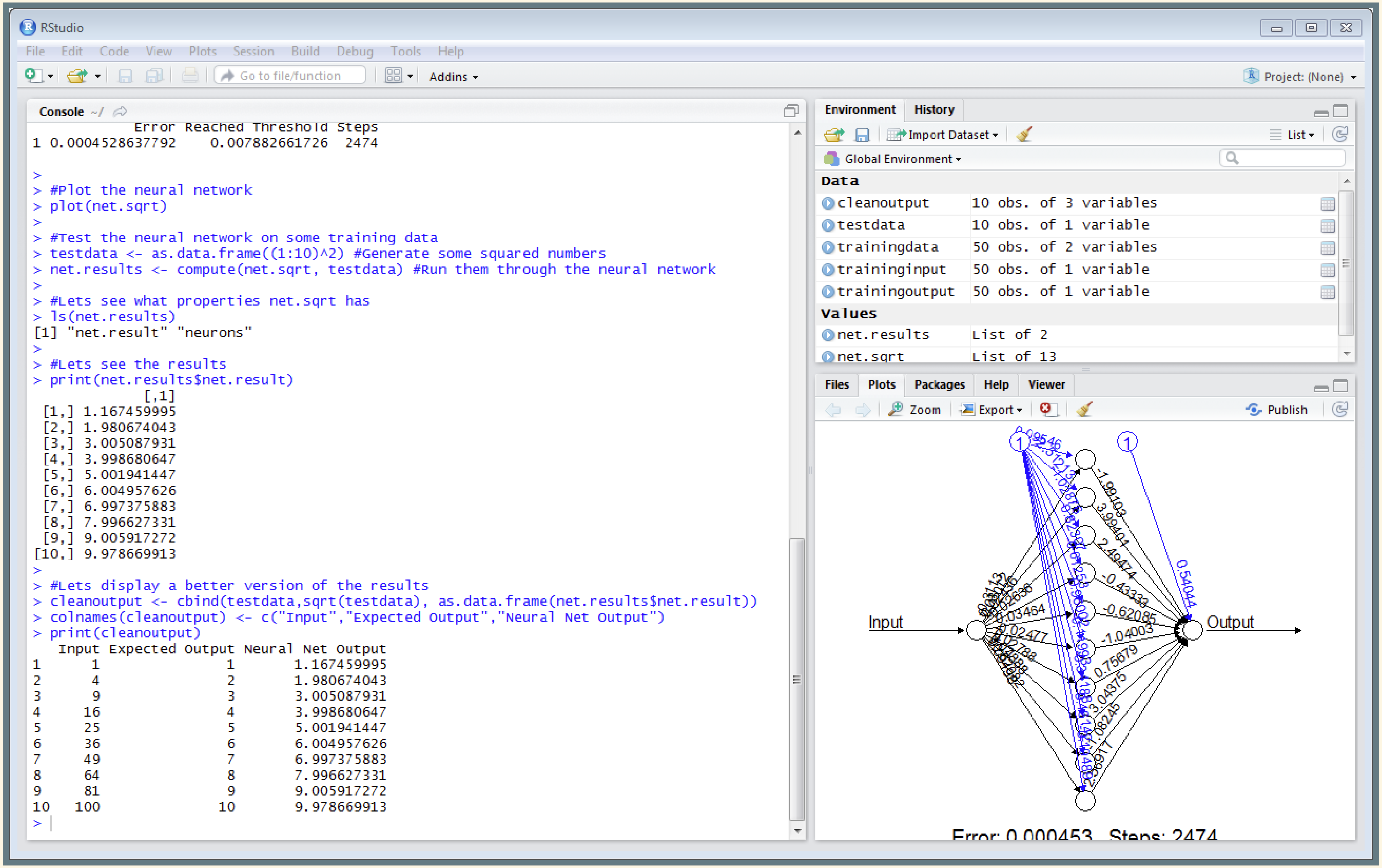
```
GET
  FILE='Z:\Niklas\für Heiko\TCI_15072015.sav'.
DATASET NAME DataSet1 WINDOW=FRONT.
GET
  FILE='\\NEUROSCIENCE\HiWi\Aufgaben\Konkret\Fragebögen Niklas\Niklas\TCI_
DATASET NAME DataSet2 WINDOW=FRONT.
DATASET ACTIVATE DataSet2.
DATASET CLOSE DataSet1.
```

IBM SPSS Statistics -Prozessor ist bereit | Unicode:ON

	corr	ID	tci001	tci002	tci003	tci004	tci005	tci006	tci007
1	SOA	10023	1	1	1	2	1	2	
2	corr	11356	2	1	2	2	1	2	
3	corr	11364	1	1	1	2	1	1	
4	corr	11365	2	1	1	1	2	2	
5	corr	11390	1	1	1	1	1	2	
6	corr	11370	1	2	1	2	1	1	
7	corr	11363	1	1	2	2	1	2	
8	corr	11085	2	2	1	2	1	2	
9	corr	11335	1	2	1	1	1	2	
10	corr	11401	1	1	1	2	1	1	
11	corr	6194	1	1	1	2	1	1	
12	corr	10628	1	1	2	2	1	2	
13	SOA	6515	1	1	2	2	1	2	
14	SOA	11366	1	1	2	2	1	2	
15	SOA	9856	1	1	1	2	1	2	
16	SOA	10664	1	2	2	2	1	2	
17	SOA	11429	1	1	1	2	1	1	
18	SOA	7443	1	1	1	2	1	1	
19	SOA	11387	1	1	1	2	1	2	
20	SOA	10843	1	1	2	1	1	2	
21	SOA	8597	1	1	2	2	1	2	
22	SOA	10680	1	2	1	2	1	1	
23	SOA	11369	2	1	2	2	1	2	
24	corr	10135	1	1	1	2	1	1	
25	SOA	10171	2	1	2	2	1	2	
26	SOA	11436	1	1	1	2	2	2	
27	SOA	11131	2	2	2	2	1	1	
28	SOA	11393	1	1	2	2	1	2	
29	SOA	10704	2	2	2	2	1	1	
30	SOA	11437	2	1	2	2	1	2	
31	SOA	11438	2	2	2	2	1	2	
32	SOA	11439	1	1	1	2	1	1	
33	corr	10652	1	1	1	2	1	2	
34	corr	11440	1	1	2	1	1	1	
35	SOA	11446	1	1	2	1	1	1	
36	SOA	10469	2	1	1	1	1	2	
37	SOA	11452	2	1	1	2	1	2	

IBM SPSS Statistics -Prozessor ist bereit | Unicode:ON

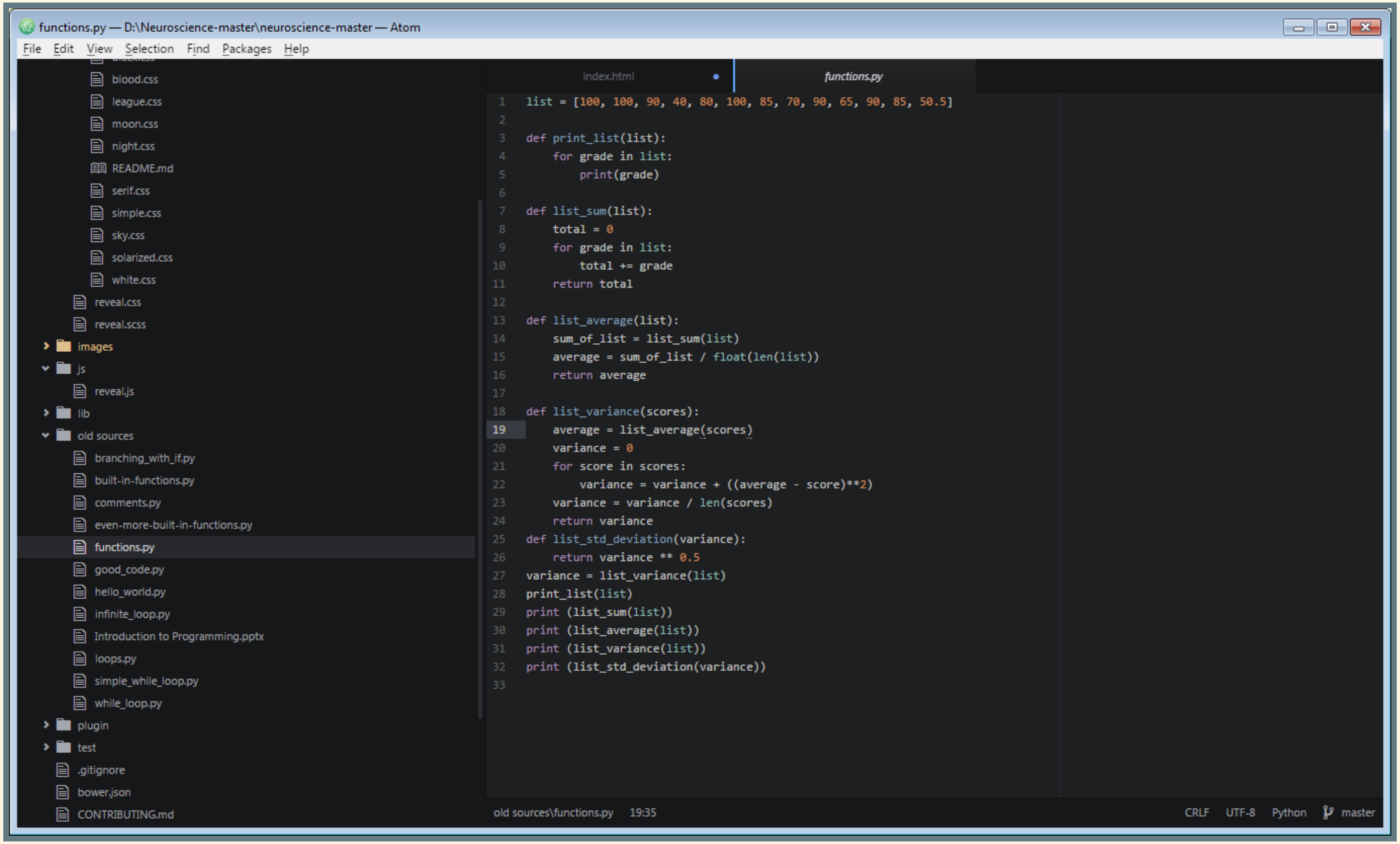
R



```
        Error Reached Threshold Steps
1 0.0004528637792    0.007882661726  2474

>
> #Plot the neural network
> plot(net.sqrt)
>
> #Test the neural network on some training data
> testdata <- as.data.frame((1:10)^2) #Generate some squared numbers
> net.results <- compute(net.sqrt, testdata) #Run them through the neural network
>
> #Lets see what properties net.sqrt has
> ls(net.results)
[1] "net.result" "neurons"
>
> #Lets see the results
> print(net.results$net.result)
             [,1]
 [1,] 1.167459995
 [2,] 1.980674043
 [3,] 3.005087931
 [4,] 3.998680647
 [5,] 5.001941447
 [6,] 6.004957626
 [7,] 6.997375883
 [8,] 7.996627331
 [9,] 9.005917272
[10,] 9.978669913
>
> #Lets display a better version of the results
> cleanoutput <- cbind(testdata,sqrt(testdata), as.data.frame(net.results$net.result))
> colnames(cleanoutput) <- c("Input","Expected Output","Neural Net Output")
> print(cleanoutput)
   Input Expected Output Neural Net Output
1      1               1       1.167459995
2      4               2       1.980674043
3      9               3       3.005087931
4     16               4       3.998680647
5     25               5       5.001941447
6     36               6       6.004957626
7     49               7       6.997375883
8     64               8       7.996627331
9     81               9       9.005917272
10   100              10       9.978669913
> |
```

```
Data
cleanoutput      10 obs. of 3 variables
testdata         10 obs. of 1 variable
trainingdata     50 obs. of 2 variables
traininginput    50 obs. of 1 variable
trainingoutput   50 obs. of 1 variable
Values
net.results      List of 2
net.sqrt         List of 13
```

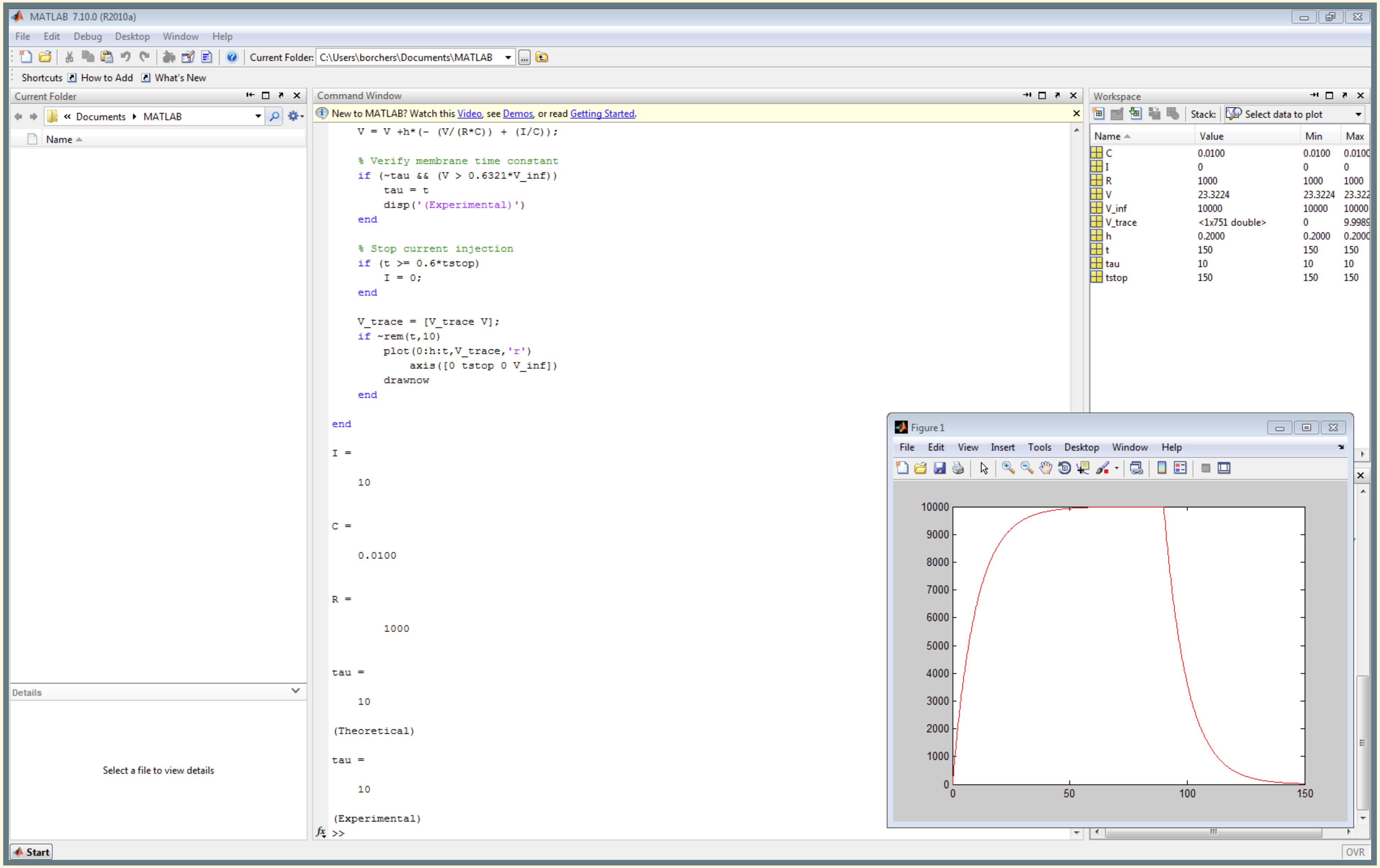
PYTHON



functions.py — D:\Neuroscience-master\neuroscience-master — Atom

```python
list = [100, 100, 90, 40, 80, 100, 85, 70, 90, 65, 90, 85, 50.5]

def print_list(list):
    for grade in list:
        print(grade)

def list_sum(list):
    total = 0
    for grade in list:
        total += grade
    return total

def list_average(list):
    sum_of_list = list_sum(list)
    average = sum_of_list / float(len(list))
    return average

def list_variance(scores):
    average = list_average(scores)
    variance = 0
    for score in scores:
        variance = variance + ((average - score)**2)
    variance = variance / len(scores)
    return variance
def list_std_deviation(variance):
    return variance ** 0.5
variance = list_variance(list)
print_list(list)
print (list_sum(list))
print (list_average(list))
print (list_variance(list))
print (list_std_deviation(variance))
```

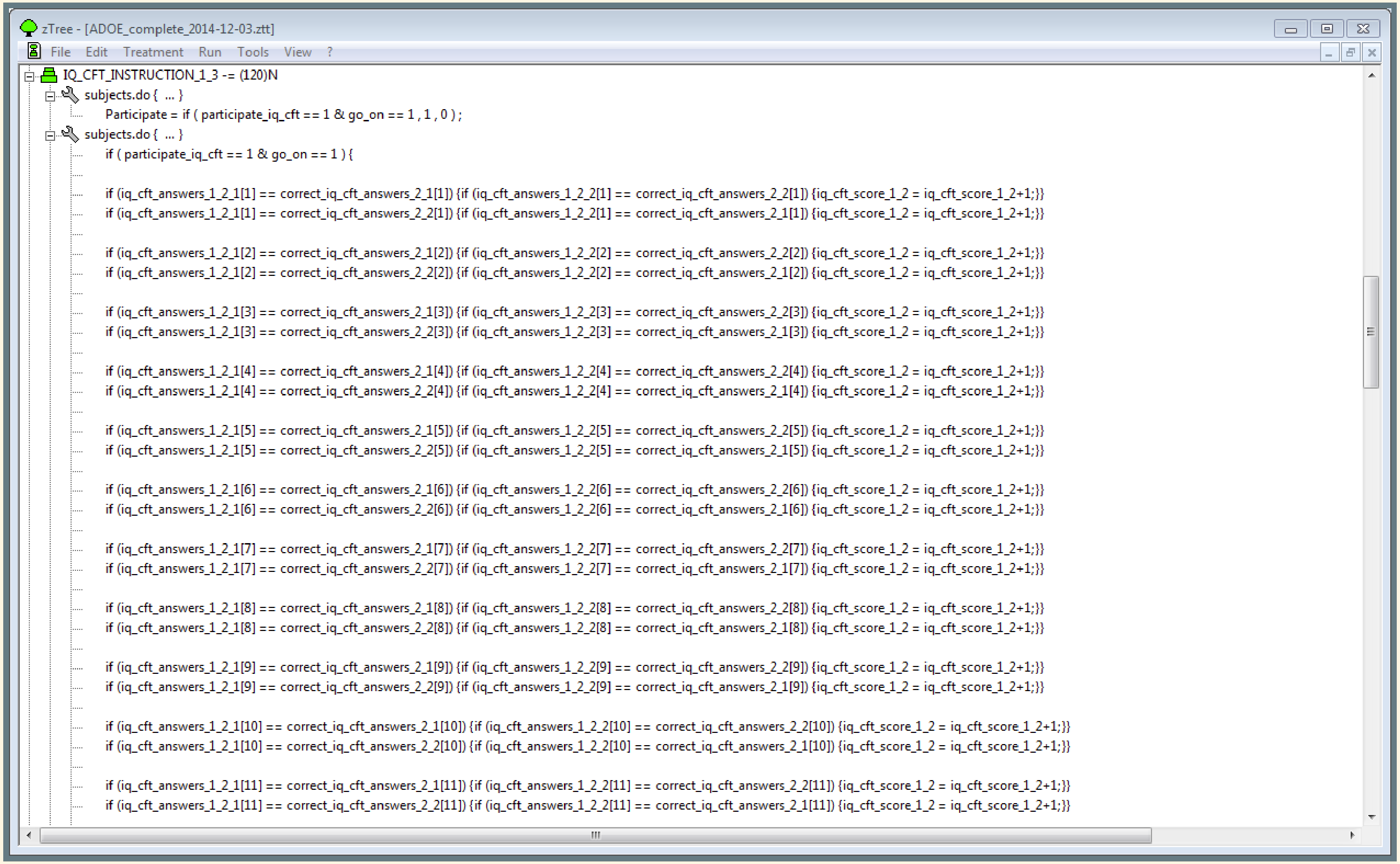
MATLAB


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
Current Folder: C:\Users\borchers\Documents\MATLAB
Shortcuts How to Add What's New
Current Folder
« Documents ▸ MATLAB
Name
Details
Select a file to view details
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Stack:
Select data to plot
Figure 1
File Edit View Insert Tools Desktop Window Help
10000
9000
8000
7000
6000
5000
4000
3000
2000
1000
0
0
50
100
150
Start
OVR

```
    V = V +h*(- (V/(R*C)) + (I/C));

    % Verify membrane time constant
    if (~tau && (V > 0.6321*V_inf))
        tau = t
        disp('(Experimental)')
    end

    % Stop current injection
    if (t >= 0.6*tstop)
        I = 0;
    end

    V_trace = [V_trace V];
    if ~rem(t,10)
        plot(0:h:t,V_trace,'r')
            axis([0 tstop 0 V_inf])
        drawnow
    end

end

I =

    10


C =

    0.0100


R =

        1000


tau =

    10

(Theoretical)

tau =

    10

(Experimental)
>>
```

Name	Value	Min	Max
C	0.0100	0.0100	0.0100
I	0	0	0
R	1000	1000	1000
V	23.3224	23.3224	23.322
V_inf	10000	10000	10000
V_trace	<1x751 double>	0	9.9989
h	0.2000	0.2000	0.2000
t	150	150	150
tau	10	10	10
tstop	150	150	150

ZTREE



zTree - [ADOE_complete_2014-12-03.ztt]

File Edit Treatment Run Tools View ?

```
IQ_CFT_INSTRUCTION_1_3 -= (120)N
  subjects.do { ... }
    Participate = if ( participate_iq_cft == 1 & go_on == 1 , 1 , 0 ) ;
  subjects.do { ... }
    if ( participate_iq_cft == 1 & go_on == 1 ) {

    if (iq_cft_answers_1_2_1[1] == correct_iq_cft_answers_2_1[1]) {if (iq_cft_answers_1_2_2[1] == correct_iq_cft_answers_2_2[1]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[1] == correct_iq_cft_answers_2_2[1]) {if (iq_cft_answers_1_2_2[1] == correct_iq_cft_answers_2_1[1]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[2] == correct_iq_cft_answers_2_1[2]) {if (iq_cft_answers_1_2_2[2] == correct_iq_cft_answers_2_2[2]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[2] == correct_iq_cft_answers_2_2[2]) {if (iq_cft_answers_1_2_2[2] == correct_iq_cft_answers_2_1[2]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[3] == correct_iq_cft_answers_2_1[3]) {if (iq_cft_answers_1_2_2[3] == correct_iq_cft_answers_2_2[3]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[3] == correct_iq_cft_answers_2_2[3]) {if (iq_cft_answers_1_2_2[3] == correct_iq_cft_answers_2_1[3]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[4] == correct_iq_cft_answers_2_1[4]) {if (iq_cft_answers_1_2_2[4] == correct_iq_cft_answers_2_2[4]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[4] == correct_iq_cft_answers_2_2[4]) {if (iq_cft_answers_1_2_2[4] == correct_iq_cft_answers_2_1[4]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[5] == correct_iq_cft_answers_2_1[5]) {if (iq_cft_answers_1_2_2[5] == correct_iq_cft_answers_2_2[5]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[5] == correct_iq_cft_answers_2_2[5]) {if (iq_cft_answers_1_2_2[5] == correct_iq_cft_answers_2_1[5]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[6] == correct_iq_cft_answers_2_1[6]) {if (iq_cft_answers_1_2_2[6] == correct_iq_cft_answers_2_2[6]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[6] == correct_iq_cft_answers_2_2[6]) {if (iq_cft_answers_1_2_2[6] == correct_iq_cft_answers_2_1[6]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[7] == correct_iq_cft_answers_2_1[7]) {if (iq_cft_answers_1_2_2[7] == correct_iq_cft_answers_2_2[7]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[7] == correct_iq_cft_answers_2_2[7]) {if (iq_cft_answers_1_2_2[7] == correct_iq_cft_answers_2_1[7]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[8] == correct_iq_cft_answers_2_1[8]) {if (iq_cft_answers_1_2_2[8] == correct_iq_cft_answers_2_2[8]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[8] == correct_iq_cft_answers_2_2[8]) {if (iq_cft_answers_1_2_2[8] == correct_iq_cft_answers_2_1[8]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[9] == correct_iq_cft_answers_2_1[9]) {if (iq_cft_answers_1_2_2[9] == correct_iq_cft_answers_2_2[9]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[9] == correct_iq_cft_answers_2_2[9]) {if (iq_cft_answers_1_2_2[9] == correct_iq_cft_answers_2_1[9]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[10] == correct_iq_cft_answers_2_1[10]) {if (iq_cft_answers_1_2_2[10] == correct_iq_cft_answers_2_2[10]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[10] == correct_iq_cft_answers_2_2[10]) {if (iq_cft_answers_1_2_2[10] == correct_iq_cft_answers_2_1[10]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}

    if (iq_cft_answers_1_2_1[11] == correct_iq_cft_answers_2_1[11]) {if (iq_cft_answers_1_2_2[11] == correct_iq_cft_answers_2_2[11]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
    if (iq_cft_answers_1_2_1[11] == correct_iq_cft_answers_2_2[11]) {if (iq_cft_answers_1_2_2[11] == correct_iq_cft_answers_2_1[11]) {iq_cft_score_1_2 = iq_cft_score_1_2+1;}}
```

SOFTWARE DEVELOPMENT BASICS

From here on all code will be Python code

PYTHON HAS A CORE PHILOSOPHY

1. Beautiful is better than ugly
2. Explicit is better than implicit
3. Simple is better than complex
4. Complex is better than complicated
5. Readability counts

YOUR FIRST PROGRAM

```
print ('Hello World')
```

print() is a defined function which writes its input to the standard output, e.g. your command line

Each function in python ends with () the brackets can contain one or more parameters

COMMENTING YOUR CODE

```
print(‘Single line comment’) # This symbol indicates a single line commer
print(‘Multiline comments are different’)
"""This is a comment
which spans over multiple
lines."""
# But single line comments are
# favored by the python style guide PEP8
```

Commenting your code makes it more read- and maintainable

DATA TYPES

```
numbers = 5
floats = 0.5
strings = “Hello World”
boolean = True #or "False"
lists = [list_item_1, list_item_2 .. ]
dictionaries = { ‘key one’ : value, ‘key two’ : value .. }
```

OPERATORS 1/2

Python has many built-in operators

The most used arithmetic Operators are

```
a = 12
b = 5
c = a + b # addition
d = a - b # subtraction
e = a * b # multiplication
f = a / b # division
g = a % b # modulus
h = a ** b # exponent
```

OPERATORS 2/2

Typical comparison operators are

```
== # equal
!= # not equal
<> # not equal
> # more
< # less
>= # more or equal
<= # less or equal
```

BUILT-IN FUNCTIONS

```
a = ‘Hello ‘
b = ‘World!’
c = len(a+b) # Gives us the length of "Hello World!" and stores it in c
print (a + b) # Prints out Hello World!
print (c) # Prints out the Lenght of Hello World
```

CODE FLOW AND LOOPS

- Controlling the flow of a program is often necessary to reduce complexity
- It also reduces the amount of work to complete specific tasks e.g. filling a list with numbers, you don’t have to do it by hand but can use loops, for instance with “for” or “while”
- Branching your code allows for different outcomes, this is done via “if” statements

In python there are two kinds of loops

CODE FLOW WITH "FOR" LOOPS

for loops which execute instructions FOR a specific time or a specific number of times

```
my_list = ['banana', 'strawberry', 'apple', 'lemon', 'peach']
sorted_list = sorted(my_list)
for x in range(1,11,1):
        print(x)
print()
for y in range(10,0,-1):
        print(y)
print()
for z in range(len(sorted_list)):
        print (sorted_list[z])
```

CODE FLOW WITH "WHILE" LOOPS

while loops which execute instructions WHILE a specific condition is met

```
while condition_one == another_condition:
        do_something()
        do_something_more() # optional
        break_condition == True # optional
a = 0
while a < 10:
        a =+ 1
        print(a)
```

BRANCHING WITH "IF"

The following program prints out the numbers from 0 to 100 and if they are even, odd or null

```
for x in range(0, 101, 1):
        if x == 0:
                print(x, ' is null')
        elif x % 2 != 0:
                print(x, ' is odd')
        else:
                print(x, ' is even')
```

FUNCTIONS

- Functions are necessary to reduce the lines of code
- There are two types of functions
- built-in functions like print(), sort() and so on
- and self defined functions

```
def answer():
        return 42
print(answer()) #This simple function just returns 42

def print_list(list):
        for item in list:
                print(item) # This function takes a
                # list as an argument and prints
                # every item in a new line"""
print_list(my_list)
```

INCLUDING MODULES AND LIBRARIES

- Sometimes python alone does not have all functions implemented you need
- This is what modules are for
- You can chose to import a whole modules or just some specific functions from it
- To import a complete module add "import module_name" to the start of your program
- If you just need one or more specific functions from a module you can tell python this via "from module_name import function"

```
import math
print (math.sqrt(25))

from math import sqrt
print (sqrt(25))
```

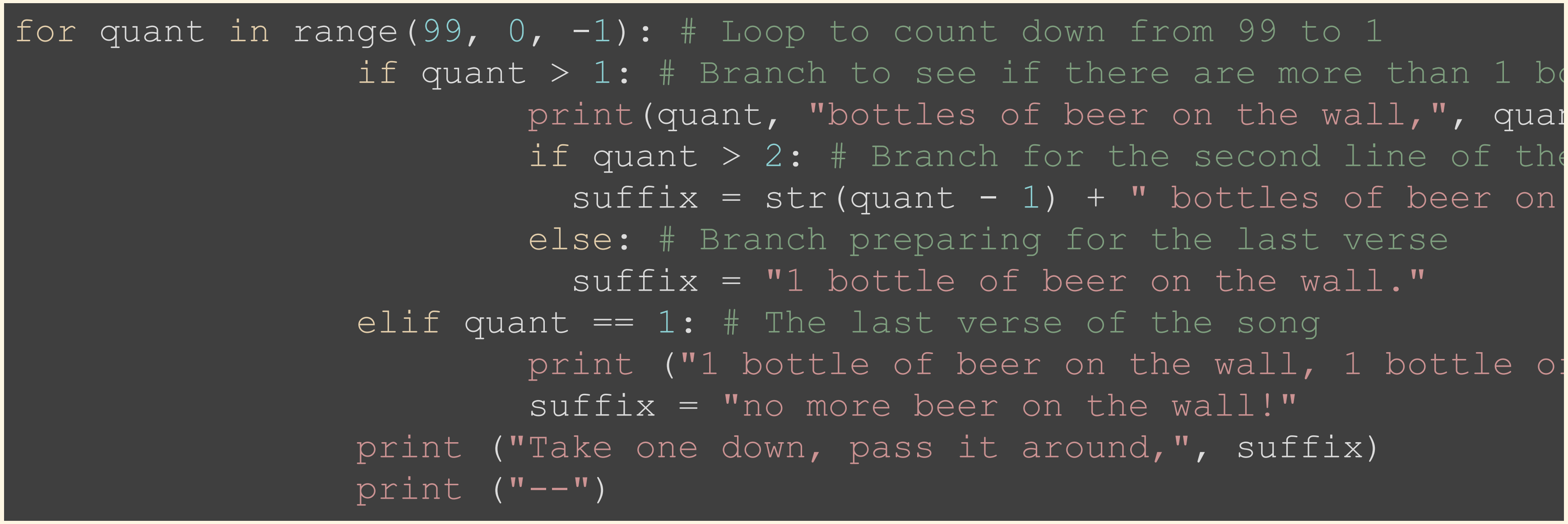
The result of both code fragments is the same

DIFFERENCES BETWEEN GOOD AND BAD CODE

GOOD CODE IS

- readable
- maintainable
- easy to understand

```python
a,t="\n%s bottles of beer on the wall","\nTake one down, pass it around"
for d in range(99,0,-1):print((a%d*2)[:-12]+t+a%(d-1 or'No'))
```



```
for quant in range(99, 0, -1): # Loop to count down from 99 to 1
                if quant > 1: # Branch to see if there are more than 1 bo
                        print(quant, "bottles of beer on the wall,", quan
                        if quant > 2: # Branch for the second line of the
                          suffix = str(quant - 1) + " bottles of beer on
                        else: # Branch preparing for the last verse
                          suffix = "1 bottle of beer on the wall."
                elif quant == 1: # The last verse of the song
                        print ("1 bottle of beer on the wall, 1 bottle o
                        suffix = "no more beer on the wall!"
                print ("Take one down, pass it around,", suffix)
                print ("--")
```

SOURCES

- MATLAB Code
- R Code
- SPSS Code
- Python Code [my own work]
- zTree Code [my own work]
- View the full source of this presentation

FURTHER READING

- Google for Education
- Codecademey Python course
- Objektorientierte Softwareentwicklung 2012 (German)
- Python in Neuroscience (from Frontiers in Neuroinformatics)
- Video2Brain (via Uni-ID, German)

THANK YOU FOR YOUR ATTENTION