Lab 2 - Becoming Familiar with R

## Some simple Math

### Use R like a calculator

3 + 9 + 12 - 7

## [1] 17

12 + 17/2 - 3/4 \* 2.5

## [1] 18.625

(12 + 17/2 - 3/4) \* 2.5

## [1] 49.375

### Try it out - Type some Math

pi \* 2^3 - sqrt(4)

## [1] 23.13274

abs(12-17\*2/3-9)

## [1] 8.333333

factorial(4)

## [1] 24

log(2,10)

## [1] 0.30103

log(2,base=10)

## [1] 0.30103

log10(2)

## [1] 0.30103

log(2)

## [1] 0.6931472

exp(0.6931472)

## [1] 2

log10(2)

## [1] 0.30103

10^0.30103

## [1] 2

sin(45 \* pi / 180)

## [1] 0.7071068

asin(0.7071068) \* 180 / pi

## [1] 45

### Storing the results of calculations

ans1 = 23 + 14/2 - 18 + (7 \* pi/2)  
ans1

## [1] 22.99557

ans2 = 13 + 11 + (17 - 4/7)  
ans1 + ans2 / 2

## [1] 43.20986

ans3 = ans2 + 9 - 2 + pi  
ans4 <- 3 + 5  
ans5 <- ans1 \* ans2  
ans3 + pi / ans4 -> ans6  
  
ans1

## [1] 22.99557

ans2

## [1] 40.42857

ans3

## [1] 50.57016

ans4

## [1] 8

ans5

## [1] 929.6782

ans6

## [1] 50.96286

## Reading and Getting Data into R

### Using the combine Command for Making Data

#### Entering Numerical Items as Data

data1 = c(3, 5, 7, 5, 3, 2, 6, 8, 5, 6, 9)  
data1

## [1] 3 5 7 5 3 2 6 8 5 6 9

Here we create a new data set by appending additional values to data 1. It’s nice that you don’t have to expand or concatenate the dataset like in other programming languages. (ex. js - data2 = […data1,4,5,7,3,4])

data2 = c(data1, 4, 5, 7, 3, 4)  
data2

## [1] 3 5 7 5 3 2 6 8 5 6 9 4 5 7 3 4

Here we update the data1 dataset by prepending items and re-assigning to the data 1 variable.

data1 = c(6, 7, 6, 4, 8, data1)  
data1

## [1] 6 7 6 4 8 3 5 7 5 3 2 6 8 5 6 9

#### Entering Text Items as Data

Below is an example of creating an array of text data.

day1 = c('Mon', 'Tue', 'Wed', 'Thu')  
day1

## [1] "Mon" "Tue" "Wed" "Thu"

Mixing text and numbers coerces the numeric data to text.

mix = c(data1, day1)  
mix

## [1] "6" "7" "6" "4" "8" "3" "5" "7" "5" "3" "2" "6"   
## [13] "8" "5" "6" "9" "Mon" "Tue" "Wed" "Thu"

### Using the scan Command for Making Data

This does not work great in a notebook - running the below chunk will prompt you in R Studio’s console. Screenshots will be included in the lab report. Note that entering a non-numeric throws an error!

data3 = scan()  
#6 7 8 7 6 3 8 9 10 7  
#6 9  
data3

## numeric(0)

#### Entering Text as Data

Again, this will prompt you at the command line. Note that entering numeric data will not throw an error, but instead coerce the data to text.

day2 = scan(what='character')  
#Mon Tue Wed  
#Thu  
day2

## character(0)

### Using the Clipboard to Make Data

We copy the data from the csv file and paste into the console prompt generated by this chunk.

data4 = scan(sep = ',')  
#23,17,12.5,11,17,12,14.5,9  
#11,9,12.5,14.5,17,8,21  
data4

## numeric(0)

We do the same for text data.

data5 = scan(sep = ',', what = 'char')  
#"Jan","Feb","Mar","Apr","May","Jun"  
#"Jul","Aug","Sep","Oct","Nov","Dec"  
data5

## character(0)

### Reading a File of Data from a Disk

You can get the present working directory with the below command.

getwd()

## [1] "C:/Users/chris/development/CTU/CS511/Week2/Lab2"

Using the text file I created in this project, I can use a relative path to load it in with scan. Note that we have to include a separator to read individual values correctly.

data6 = scan(sep = ',', file = 'cfichman-chap2-numeric-input.txt')  
data6

## [1] 23.0 17.0 12.5 11.0 17.0 12.0 14.5 9.0 11.0 9.0 12.5 14.5 17.0 8.0 21.0

You can specify a working directory with the setwd() command. In notebooks, the working directory state is contained to each running chunk, which is good, because we don’t want to mess up the root of our project, but we can still demonstrate how setwd() works. This needed to be disabled for HTML report.

#setwd('..')  
#getwd()

You can perform various other filesystem commands with r.

Directory view - list folders AND files, including hidden files with all.files.

cwd\_contents = dir()  
all\_cwd\_contents = dir(all.files = TRUE)  
desktop\_contents = dir('Desktop')  
CTU\_courses = dir('../../')  
  
print("CWD:")

## [1] "CWD:"

print(cwd\_contents)

## [1] "cfichman-chap2-csv2.csv" "cfichman-chap2-grass.csv"   
## [3] "cfichman-chap2-na.csv" "cfichman-chap2-numeric-input.txt"  
## [5] "cfichman-chap2-space-sep.csv" "cfichman-chap2-table-2-2.csv"   
## [7] "cfichman-chap2-text-input.txt" "cfichman-chap2-tsv.csv"   
## [9] "cfichman-history.txt" "cfichman-lab2-birds.csv"   
## [11] "cfichman-lab2-birds.txt" "cfichman-lab2-data7-2.txt"   
## [13] "cfichman-lab2-data7.txt" "cfichman-lab2-image.Rdata"   
## [15] "cfichman-lab2-object-save.txt" "Cfichman-Lab2.html"   
## [17] "Cfichman-Lab2.Rmd" "cfichman-save-all.Rdata"   
## [19] "cfichman-save-file-list.Rdata" "cfichman-save-file-object.Rdata"   
## [21] "cfichman-save-file-pattern.Rdata" "cfichman-savedata-test.Rdata"   
## [23] "Lab2.Rproj"

print("ALL CWD:")

## [1] "ALL CWD:"

print(all\_cwd\_contents)

## [1] "." ".."   
## [3] ".git" ".gitignore"   
## [5] ".Rhistory" ".Rproj.user"   
## [7] "cfichman-chap2-csv2.csv" "cfichman-chap2-grass.csv"   
## [9] "cfichman-chap2-na.csv" "cfichman-chap2-numeric-input.txt"  
## [11] "cfichman-chap2-space-sep.csv" "cfichman-chap2-table-2-2.csv"   
## [13] "cfichman-chap2-text-input.txt" "cfichman-chap2-tsv.csv"   
## [15] "cfichman-history.txt" "cfichman-lab2-birds.csv"   
## [17] "cfichman-lab2-birds.txt" "cfichman-lab2-data7-2.txt"   
## [19] "cfichman-lab2-data7.txt" "cfichman-lab2-image.Rdata"   
## [21] "cfichman-lab2-object-save.txt" "Cfichman-Lab2.html"   
## [23] "Cfichman-Lab2.Rmd" "cfichman-save-all.Rdata"   
## [25] "cfichman-save-file-list.Rdata" "cfichman-save-file-object.Rdata"   
## [27] "cfichman-save-file-pattern.Rdata" "cfichman-savedata-test.Rdata"   
## [29] "Lab2.Rproj"

print("Desktop: ")

## [1] "Desktop: "

print(desktop\_contents)

## character(0)

print("CTU: ")

## [1] "CTU: "

print(CTU\_courses)

## [1] "Beginning R-The Statistical Programming Language.pdf"  
## [2] "How-to-Lie-with-Statistics.pdf"   
## [3] "Week1"   
## [4] "Week2"

List only files

list.files()

## [1] "cfichman-chap2-csv2.csv" "cfichman-chap2-grass.csv"   
## [3] "cfichman-chap2-na.csv" "cfichman-chap2-numeric-input.txt"  
## [5] "cfichman-chap2-space-sep.csv" "cfichman-chap2-table-2-2.csv"   
## [7] "cfichman-chap2-text-input.txt" "cfichman-chap2-tsv.csv"   
## [9] "cfichman-history.txt" "cfichman-lab2-birds.csv"   
## [11] "cfichman-lab2-birds.txt" "cfichman-lab2-data7-2.txt"   
## [13] "cfichman-lab2-data7.txt" "cfichman-lab2-image.Rdata"   
## [15] "cfichman-lab2-object-save.txt" "Cfichman-Lab2.html"   
## [17] "Cfichman-Lab2.Rmd" "cfichman-save-all.Rdata"   
## [19] "cfichman-save-file-list.Rdata" "cfichman-save-file-object.Rdata"   
## [21] "cfichman-save-file-pattern.Rdata" "cfichman-savedata-test.Rdata"   
## [23] "Lab2.Rproj"

Prompt user to select a file, opening a browser at a the current working directory location.

data7 = scan(sep=',', file.choose())  
data7

## [1] 23.0 17.0 12.5 11.0 17.0 12.0 14.5 9.0 11.0 9.0 12.5 14.5 17.0 8.0 21.0

Scan text data!

data8 = scan('cfichman-chap2-text-input.txt', what = 'char', sep = ',')  
data8

## [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"

## Reading Bigger Data Files

### The read.csv() Command

I’ve created a file using table2-2 from the text, and made it comma separated. Now we will read it in.

fw = read.csv('cfichman-chap2-table-2-2.csv')  
fw

## abund flow  
## 1 9 2  
## 2 25 3  
## 3 15 5  
## 4 2 9  
## 5 14 14  
## 6 25 24  
## 7 24 29  
## 8 47 34

####Alternative Commands for Reading Data in R

Note that read.table must have header set to true if the input file has a header and you want to use it. Note it seems like there was a bad edit in the text and the example doesn’t line up with the description, so I followed the example which shows how to include row names.

ssv\_path = 'cfichman-chap2-space-sep.csv'  
ssv = read.table(ssv\_path, header = TRUE)  
ssv = read.csv(ssv\_path, sep = ' ')  
ssv

## data1 data2 data3  
## 1 1 2 4  
## 2 4 5 3  
## 3 3 4 5  
## 4 3 6 6  
## 5 4 5 9

tsv\_path = 'cfichman-chap2-tsv.csv'  
tsv = read.delim(tsv\_path)  
tsv = read.csv(tsv\_path, sep = '\t')  
tsv = read.table(tsv\_path, header=TRUE, sep = '\t')  
tsv

## data1 data2 data3  
## 1 1 2 4  
## 2 4 5 3  
## 3 3 4 5  
## 4 3 6 6  
## 5 4 5 9

csv2\_path = 'cfichman-chap2-csv2.csv'  
csv2\_data = read.csv(csv2\_path, row.names = 1, sep = '\t')  
csv2\_data

## data1 data2 data3  
## mon 1 2 4  
## tue 4 5 3  
## wed 3 4 5  
## thu 3 6 6  
## fri 4 5 9

#### Missing Values in Data Files

grass = read.csv('cfichman-chap2-na.csv', row.names = 1, sep = '\t')  
grass

## mow unmow  
## 1 12 8  
## 2 15 9  
## 3 17 7  
## 4 11 9  
## 5 15 NA

## Viewing Named Objects

### Viewing previously Loaded Named Objects

#### Viewing All Objects

ls()

## [1] "all\_cwd\_contents" "ans1" "ans2" "ans3"   
## [5] "ans4" "ans5" "ans6" "csv2\_data"   
## [9] "csv2\_path" "CTU\_courses" "cwd\_contents" "data1"   
## [13] "data2" "data3" "data4" "data5"   
## [17] "data6" "data7" "data8" "day1"   
## [21] "day2" "desktop\_contents" "fw" "grass"   
## [25] "mix" "ssv" "ssv\_path" "tsv"   
## [29] "tsv\_path"

#### Viewing Only Matching Names

ls(pattern = 'd')

## [1] "all\_cwd\_contents" "csv2\_data" "cwd\_contents" "data1"   
## [5] "data2" "data3" "data4" "data5"   
## [9] "data6" "data7" "data8" "day1"   
## [13] "day2" "desktop\_contents"

ls(pattern = 'da')

## [1] "csv2\_data" "data1" "data2" "data3" "data4" "data5"   
## [7] "data6" "data7" "data8" "day1" "day2"

Pattern examples. This is similar to Regexp, but lacks much of it’s valuable functionality.

ls(pattern = '^d') # only data starting with d

## [1] "data1" "data2" "data3" "data4"   
## [5] "data5" "data6" "data7" "data8"   
## [9] "day1" "day2" "desktop\_contents"

ls(pattern = '1$') # only data ending with a 1

## [1] "ans1" "data1" "day1"

ls(pattern = '^d|^c') #only data starting with a d or a c

## [1] "csv2\_data" "csv2\_path" "cwd\_contents" "data1"   
## [5] "data2" "data3" "data4" "data5"   
## [9] "data6" "data7" "data8" "day1"   
## [13] "day2" "desktop\_contents"

ls(pattern = 'data.') # wildcard for one place

## [1] "data1" "data2" "data3" "data4" "data5" "data6" "data7" "data8"

### Removing Objects from R

You can remove single objects by using their variable names, or you can pass a list of named objects. Here we’re going to clean out all our ^list.$ variables.

list = c(1,2,3,4)  
list1 = c("hi","hello")  
list2 = c(5,6,7)  
ls(pattern = 'list')

## [1] "list" "list1" "list2"

rm(list)  
ls(pattern = '^list.$')

## [1] "list1" "list2"

remove(list = ls(pattern = '^list.$'))  
ls(pattern = '^list.$')

## character(0)

## Types of Data Items

Here we read in a mix of data that interprets which data type to use.

grass2 = read.csv('cfichman-chap2-grass.csv', sep='\t', as.is = 2)  
grass2

## idx species cut  
## 1 1 12 mow  
## 2 2 15 mow  
## 3 3 17 mow  
## 4 4 11 mow  
## 5 5 15 mow  
## 6 6 8 unmow  
## 7 7 9 unmow  
## 8 8 7 unmow  
## 9 9 9 unmow

### Converting between Number and Text data

Here we convert the data we loaded from a file to various types.

data7i = as.integer(data7)  
data7i

## [1] 23 17 12 11 17 12 14 9 11 9 12 14 17 8 21

data7n = as.numeric(data7i)  
data7n

## [1] 23 17 12 11 17 12 14 9 11 9 12 14 17 8 21

data7c = as.character(data7)  
data7c

## [1] "23" "17" "12.5" "11" "17" "12" "14.5" "9" "11" "9"   
## [11] "12.5" "14.5" "17" "8" "21"

data7nt = as.numeric(data7c)  
data7nt

## [1] 23.0 17.0 12.5 11.0 17.0 12.0 14.5 9.0 11.0 9.0 12.5 14.5 17.0 8.0 21.0

Now we try with a factor

cut\_data = c(grass2['cut'])  
cut\_data

## $cut  
## [1] mow mow mow mow mow unmow unmow unmow unmow  
## Levels: mow unmow

cut\_n = as.numeric(as.factor(cut\_data))  
cut\_n

## [1] 1

Converting from text to numeric results in NA values. To get something useful, we must first convert to a factor, and then convert to a numeric to get numeric categorical data, which is very useful in situations where the data will be used for data analysis/machine learning.

data8

## [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" "Nov" "Dec"

data8n = as.numeric(data8)

## Warning: NAs introduced by coercion

data8n

## [1] NA NA NA NA NA NA NA NA NA NA NA NA

data8c = as.numeric(as.factor(data8))  
data8c

## [1] 5 4 8 1 9 7 6 2 12 11 10 3

## Examining Data Structure

str(grass2)

## 'data.frame': 9 obs. of 3 variables:  
## $ idx : int 1 2 3 4 5 6 7 8 9  
## $ species: int 12 15 17 11 15 8 9 7 9  
## $ cut : Factor w/ 2 levels "mow","unmow": 1 1 1 1 1 2 2 2 2

str(grass2[1])

## 'data.frame': 9 obs. of 1 variable:  
## $ idx: int 1 2 3 4 5 6 7 8 9

birds <- matrix(c(47,10,40,2,2,19,3,5,0,2,50,0,10,7,0,46,16,8,4,0,9,3,0,0,2,4,0,6,0,0), nrow = 6, ncol = 5)  
colnames(birds) = c("Garden","Hedgerow","Parkland","Pasture","Woodland")  
rownames(birds) = c("Blackbird","Caffinch","Great Tit","House Sparrow","Robin","Song Thrush")  
  
str(birds)

## num [1:6, 1:5] 47 10 40 2 2 19 3 5 0 2 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : chr [1:6] "Blackbird" "Caffinch" "Great Tit" "House Sparrow" ...  
## ..$ : chr [1:5] "Garden" "Hedgerow" "Parkland" "Pasture" ...

List all objects and show their structure

ls.str(pattern = 'data')

## csv2\_data : 'data.frame': 5 obs. of 3 variables:  
## $ data1: int 1 4 3 3 4  
## $ data2: int 2 5 4 6 5  
## $ data3: int 4 3 5 6 9  
## cut\_data : List of 1  
## $ cut: Factor w/ 2 levels "mow","unmow": 1 1 1 1 1 2 2 2 2  
## data1 : num [1:16] 6 7 6 4 8 3 5 7 5 3 ...  
## data2 : num [1:16] 3 5 7 5 3 2 6 8 5 6 ...  
## data3 : num(0)   
## data4 : num(0)   
## data5 : chr(0)   
## data6 : num [1:15] 23 17 12.5 11 17 12 14.5 9 11 9 ...  
## data7 : num [1:15] 23 17 12.5 11 17 12 14.5 9 11 9 ...  
## data7c : chr [1:15] "23" "17" "12.5" "11" "17" "12" "14.5" "9" "11" "9" "12.5" ...  
## data7i : int [1:15] 23 17 12 11 17 12 14 9 11 9 ...  
## data7n : num [1:15] 23 17 12 11 17 12 14 9 11 9 ...  
## data7nt : num [1:15] 23 17 12.5 11 17 12 14.5 9 11 9 ...  
## data8 : chr [1:12] "Jan" "Feb" "Mar" "Apr" "May" "Jun" "Jul" "Aug" "Sep" "Oct" ...  
## data8c : num [1:12] 5 4 8 1 9 7 6 2 12 11 ...  
## data8n : num [1:12] NA NA NA NA NA NA NA NA NA NA ...

See what type of data structure the data is:

class(grass2)

## [1] "data.frame"

class(c(grass2[2]))

## [1] "list"

class(birds)

## [1] "matrix" "array"

class(day1)

## [1] "character"

class(data7i)

## [1] "integer"

## Working with History Commands

### Using History Files

#### Viewing the previous command history

#history(max.show = 25)

####Saving and Recalling Lists of Commands These don’t work when building out the HTML for result so they’ve been commented out.

#savehistory(file = 'cfichman-history.txt')  
#loadhistory(file = 'cfichman-history.txt')  
#history()

### Editing History Files

You can set the max number of history entries with the following in windows.

#Sys.setenv('R\_HISTSIZE' = 512)

## Saving your work in R

I am using projects, so this section won’t work for me. But underneath the hood there’s workspace in the project.

#### Saving Named Objects

save(grass2, file = 'cfichman-save-file-object.Rdata')  
save(data7i,data7nt,data7c, file = 'cfichman-save-file-list.Rdata')  
save(list = ls(pattern = '^data.$'), file = 'cfichman-save-file-pattern.Rdata')

#### Save ALL THE THINGS

save(list = ls(all=TRUE), file = 'cfichman-save-all.Rdata')  
save.image(file = "cfichman-lab2-image.Rdata")

### Reading Data Files from Disk

#### Try it out - Save and Read a Binary Data file to and From Disk

ls()

## [1] "all\_cwd\_contents" "ans1" "ans2" "ans3"   
## [5] "ans4" "ans5" "ans6" "birds"   
## [9] "csv2\_data" "csv2\_path" "CTU\_courses" "cut\_data"   
## [13] "cut\_n" "cwd\_contents" "data1" "data2"   
## [17] "data3" "data4" "data5" "data6"   
## [21] "data7" "data7c" "data7i" "data7n"   
## [25] "data7nt" "data8" "data8c" "data8n"   
## [29] "day1" "day2" "desktop\_contents" "fw"   
## [33] "grass" "grass2" "mix" "ssv"   
## [37] "ssv\_path" "tsv" "tsv\_path"

savedata = c(9,2,4,6,5,9,2,1,1,7)

save(savedata, file = 'cfichman-savedata-test.Rdata')

rm(savedata)  
load(file='cfichman-savedata-test.Rdata')

load(file = file.choose())

savedata

## [1] 9 2 4 6 5 9 2 1 1 7

### Saving Data to Disk as Text Files

data7

## [1] 23.0 17.0 12.5 11.0 17.0 12.0 14.5 9.0 11.0 9.0 12.5 14.5 17.0 8.0 21.0

write(data7, file = 'cfichman-lab2-data7.txt', sep = ',')

write(data7, file = 'cfichman-lab2-data7-2.txt', sep = ',', ncolumns = length(data7))  
cat(data7, file='cfichman-lab2-data7-2.txt')

### Writing Matrix and Frame Data to Disk

write.table(birds, file='cfichman-lab2-birds.txt', row.names = TRUE, sep = ' ', col.names = TRUE)  
write.csv(birds, file ='cfichman-lab2-birds.csv', row.names = TRUE)

### Writing List Objects to Disk

dput(data1, file = 'cfichman-lab2-object-save.txt')  
dget('cfichman-lab2-object-save.txt')

## [1] 6 7 6 4 8 3 5 7 5 3 2 6 8 5 6 9

### Converting List Objects to Data Frames

ex\_list = list(c(12,15,17,11,15),c(8,9,7,9))  
names(ex\_list) = c('mow', 'unmow')  
ex\_list

## $mow  
## [1] 12 15 17 11 15  
##   
## $unmow  
## [1] 8 9 7 9

str(ex\_list)

## List of 2  
## $ mow : num [1:5] 12 15 17 11 15  
## $ unmow: num [1:4] 8 9 7 9

ex\_stack = stack(ex\_list)  
names(ex\_stack) = c('numbers','sample')  
ex\_stack

## numbers sample  
## 1 12 mow  
## 2 15 mow  
## 3 17 mow  
## 4 11 mow  
## 5 15 mow  
## 6 8 unmow  
## 7 9 unmow  
## 8 7 unmow  
## 9 9 unmow

unstack(ex\_stack)

## $mow  
## [1] 12 15 17 11 15  
##   
## $unmow  
## [1] 8 9 7 9

new\_list = as.list(unstack(ex\_stack))

Creating a list from dataframe

list2 = as.list(grass2)  
names(list2) = c('idx','species','cut')  
list2

## $idx  
## [1] 1 2 3 4 5 6 7 8 9  
##   
## $species  
## [1] 12 15 17 11 15 8 9 7 9  
##   
## $cut  
## [1] mow mow mow mow mow unmow unmow unmow unmow  
## Levels: mow unmow