Appendices

mcc

3/13/2020

Appendices

```
knitr::opts_chunk$set(cache = TRUE)
```

Word To The Wise

- LEARN GIT. Get a Github account!
  - Put Every computer program you write and Every stick of knowledge related to your work in a clean format on Git. You will benefit from it in the long run…
- To Biologists and Biochemists (which I also consider myself), it is your task to become more familiar with computer languages, computer concepts and math/statistics.

As of the writing of this booklet, Data Science and Bioinformatics are now centered around the computer languages R and Python.

To many researchers in data science and bioinformatics the field now includes such languages as, *in no particular order*,

- R
- Python
- Bash shell scripting
- SQL
- Julia
- Javascript
- RMarkdown, (This one is fun and easy)

Just start with one language

- Stick with it for a time and learn it. Learn the ins-and-outs of one language first.

Is R or Python better?

1. R & Python are both FREE,
2. Both have great integrated development environments (IDEs),
    - RStudio is great & FREE,
    - Spyder is also great & FREE,
    - PyCharm
    - Sublime
    - Visual Studio Code
    - Jupyter Notebook
3. Both languages have been around for > 20 years, therefore both have tons of FREE information & tutorials on YouTube,

Install R & RStudio

NOTE: I use Ubuntu.

1. Go to: https://cran.r-project.org/
2. Choose R for your operating system.
3. If you are using Linux, I recommend that you download/install 4 R files.
    1. r-base-core_#.#.#.*.deb (approx. 30 MB)
    2. r-base-dev_#.#.#.*.deb (approx. 45 KB)
    3. r-base_#.#.#*.deb (approx. 90 KB)
    4. r-base-html-#.#.#*.deb (approx. 90 KB)
4. If you do have Linux you may try this video: How to install R.
5. Go to: https://www.rstudio.com/
6. From RStudio's homepage click, *Products* then click *RStudio* from the drop-down menu.
7. Click the Download of the FREE version of RStudio Desktop
8. Click *RStudio #.#* to download a version for your machine
9. Have Linux? Try this - How to install RStudio.
10. If you are looking for instructions for Mac & Windows machines try:
    - FreeCodeCamp

Load Libraries Used In This Project

If you are using Ubuntu/Linux you may need these Linux libraries first.

```
sudo apt-get install libcurl4-openssl-dev libssl-dev libxml2-dev build-essential
```

To install `car` & `rgl`

```
sudo apt-get install xorg libx11-dev libglu1-mesa-dev libfreetype6-dev
```

```
install.packages("rlang")
library(rlang)

load_or_install <- function(package_names) {
    for(package_name in package_names) {
        if(!is_installed(package_name)) {
            install.packages(package_name,
            repos = "http://lib.stat.cmu.edu/R/CRAN",
            dependencies = TRUE)
        }
        library(package_name,character.only=TRUE,quietly=TRUE,verbose=FALSE)
        print("OK")
        }
}

load_or_install(c("doMC", "corrplot", "knitr", "caret", "tidyverse"))

load_or_install(c("ggplot2", "rmarkdown", "bookdown", "blogdown", "kernlab"))

load_or_install(c("e1071", "plyr", "RColorBrewer", "neuralnet", "ggfortify"))

load_or_install(c("rpart", "MASS", "tidyr", "ggplot2", "seqinr", "Boruta", "kableExtra"))

load_or_install(c("LogicReg", "randomForest", "foreach", "caretEnsemble"))

load_or_install(c("import", "dplyr", "stringr", "stringi", "readr", "tinytex"))
```

Calculate the amino acid compositions (AAC) and Di-peptide compositions (DPC)

from .fasta formats, {Myoglobin, Non-Myoglobin}

Calculating the Amino Acid and Di-peptide composition of a protein string is a simple calculation requiring the total amino acid length of the peptide or poly-peptide of interest and a count of substrings. Initially, the command `seqinr::read.fasta` reads .fasta file formats and returns a list of proteins stripping away all other information. Secondly, the command `stringr::str_count()` produces an integer value of the number of substrings in a larger string, i.e. `peptide`.

For example, `aa_nums[j] = str_count(peptide, col_titles[j]) / total_aa`,

Where; `aa_nums[j]` is an array to saving values for later writing to file, `peptide` is the string to check, i.e. protein of interest, `col_titles[j]` is the substring which is either a single amino acid or di-peptide.

Input: .fasta Output: .csv

Libraries

```
Libraries = c("stringr", "knitr", "seqinr")

for (p in Libraries) {  # Install Libraries
    library(p, character.only = TRUE)
}

opts_chunk$set(cache = TRUE,
               warning = FALSE,
               message = FALSE,
               align = "center")
```

Import uniprot-myoglobin.fasta - Read peptide lines

```
read_fasta <- function(file) {
    listo_proteins <- read.fasta(file = file,
                                 seqtype = "AA",
                                 as.string = TRUE,
                                 seqonly = FALSE,
                                 strip.desc = TRUE)
    return(listo_proteins)
}

file = "./00-data/ORIGINAL_DATA/uniprot-myoglobin.fasta"
myoglobins <- read_fasta(file)
```

Column_titles

```
column_titles = function() {
    peptides = c("A", "C", "D", "E", "F",
                 "G", "H", "I", "K", "L",
                 "M", "N", "P", "Q", "R",
                 "S", "T", "V", "W", "Y")

    # Add DIPEPTIDES column titles
    di_titles = vector(mode = "character", length = 400)
    k = 1
    for (i in 1:20) {
        for (j in 1:20) {
            di_titles[k] <- paste(peptides[i], peptides[j], sep = "")
            k = k + 1
        }
    }
    aa_di_titles <- c("Class","TotalAA","PID", peptides, di_titles)
    return(aa_di_titles)
}

col_titles <- column_titles()
col_titles
```

Write empty .csv

```
write_empty_csv <- function(protein_class = "C") {
    col_titles <- column_titles()
    file_name <- paste(protein_class, "_aac_dpc.csv", sep = "")
    write.table(t(col_titles),
                file_name,
                sep = ",",
                col.names = FALSE,
                row.names = FALSE,
                eol = "\n")
    return(file_name)
}

file_name <- write_empty_csv()
```

Calculate AAC and DPC values function

```
calc_aac_dpc <- function(peptide, protein_class = "C", i, file_name) {
    aa_nums = matrix(0, ncol = 423)
    ###############################
    # First column is class
    aa_nums[1] = ifelse(protein_class == "C", 0, 1)
    # Second column is total number of amino acids
    total_aa = nchar(peptide)
    aa_nums[2] = total_aa
    # Third line is 'Protein ID', PID
    aa_nums[3] = paste(protein_class, i, sep = "")
    # Column 4:423 - Calculate AAC/DPC
    for (j in 4:423) {
        aa_nums[j] = str_count(peptide, col_titles[j]) / total_aa
        }
    write(t(aa_nums), file = file_name, append = TRUE, ncolumns = 423, sep = ",")
}
```

Run Myoglobin

```
# RUN Myoglobin
for (i in 1:1124) {
    peptide <- myoglobins[[i]][1]
    calc_aac_dpc(peptide, protein_class = "M", i, file_name)
}
```

Run Control / Human-NOT-myoglobin

- Import data - Read peptide lines

```
read_fasta <- function(file) {
    listo_proteins <- read.fasta(file = file,
                                seqtype = "AA",
                                as.string = TRUE,
                                seqonly = FALSE,
                                strip.desc = TRUE)
    return(listo_proteins)
}

file = "./00-data/ORIGINAL_DATA/uniprot-human+NOT+hemoglobin+NOT+myoglobin+random.fasta"
controls <- read_fasta(file)
```

Run Controls

```
for (i in 1:1216) {
    peptide <- controls[[i]][1]
    calc_aac_dpc(peptide, protein_class = "C", i, file_name)
}
```

KEEP AAC ONLY FOR RAW DATA

```
file = "./00-data/aac_dpc_values/C+M_aac_dpc.csv"
C+M_aac_dpc <- read.csv(file,
                        stringsAsFactors=FALSE)
# View(`C+M_aac_dpc`)

# Select 1st thru 23rd variables
c_m_RAW_AAC <- C+M_aac_dpc[c(1:23)]
```

- To A Comma Delimited Text File

```
setwd("../00-data/02-aac_dpc_values/")

write.table(c_m_RAW_AAC,
            file = "./00-data/02-aac_dpc_values/c_m_RAW_AAC.csv",
            sep = ",",
            row.names = F)
```

Transform {C, F, I} from c_m_RAW_AAC

```
library(readr)

file = "../00-data/02-aac_dpc_values/c_m_RAW_AAC.csv"
c_m_RAW_AAC <- read_csv(file,
                        col_types = cols(Class = col_factor(levels = c("0","1"))))
c_m_TRANSFORMED_AAC <- c_m_RAW_AAC
```

1. Transfrom C,F,I using sqrt(x)
2. Columns: C=5, F=8, I=11

```
c_m_TRANSFORMED_AAC[, 5] <- sqrt(c_m_TRANSFORMED_AAC[, 5]) # C
c_m_TRANSFORMED_AAC[, 8] <- sqrt(c_m_TRANSFORMED_AAC[, 8]) # F
c_m_TRANSFORMED_AAC[,11] <- sqrt(c_m_TRANSFORMED_AAC[,11]) # I

file = "./00-data/02-aac_dpc_values/c_m_TRANSFORMED.csv"
write_csv(c_m_TRANSFORMED_AAC,
          file = file,
          col_names = T)
```

Where To Find Help

1. Cheat Sheets
2. https://community.rstudio.com
3. https://www.reddit.com/r/RStudio/
4. https://R-bloggers.com/

5. https://resources.rstudio.com/
6. Rpubs.com

- **Rpubs.com** contains R/RStudio notebooks and Markdown pages, VERY HELPFUL work from other peoples online R documents. It is a way to learn from others and share your work. - Sign up, it is FREE! then press: *Get Started*

NOTE: If you are interested in seeing what others have published search Google, Rpubs.com does not have its own search function. In Google, Search: `site:rpubs.com eda`

Other sites:

1. Coursera
2. Stack Overflow
3. Quora
4. Roger Peng's EDA
5. Bookdown - **terrible** yet necessary resource

The Lean Publishing (https://leanpub.com) company contains a library in the form of FREE down-loadable books/pdfs. I recommend;

1. How to be a modern scientist[1] by Jeffrey Leek[2]
2. R Programming for Data Science[3] by Roger Peng[4]
3. Exploratory Data Analysis with R[5] by Roger Peng
4. Data Analysis for the Life Sciences[6] by Rafael Irizarry & Michael Love

Machine Setting & Session Info

```
Sys.info()[c(1:3,5)]
```

```
##                                                 sysname
##                                                 "Linux"
##                                                 release
##                                     "5.3.0-40-generic"
##                                                 version
## "#32~18.04.1-Ubuntu SMP Mon Feb 3 14:05:59 UTC 2020"
##                                                 machine
##                                                "x86_64"
```

```
sessionInfo()
```

```
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 18.04.4 LTS
##
```

[1] https://leanpub.com/modernscientist
[2] http://jtleek.com
[3] https://leanpub.com/rprogramming
[4] https://simplystatistics.org
[5] https://leanpub.com/exdata
[6] https://leanpub.com/dataanalysisforthelifesciences

```
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.7.1
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.7.1
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] knitr_1.28
##
## loaded via a namespace (and not attached):
##  [1] compiler_3.6.0   magrittr_1.5     tools_3.6.0      htmltools_0.4.0
##  [5] yaml_2.2.1       Rcpp_1.0.3       codetools_0.2-16 stringi_1.4.6
##  [9] rmarkdown_2.1    stringr_1.4.0    xfun_0.12        digest_0.6.25
## [13] rlang_0.4.5      evaluate_0.14
```