Problem Set 2

Cameron Adams

September 15, 2017

```
##########
# Chunk 1: housekpeeing
##########

rm(list=ls())

setwd("/Users/CamAdams/repos/STAT243/ps2/") #wd
knitr::opts_chunk$set(cache=TRUE,background="#F7F7F7") #knitr global opts

#load packages
require(knitr)
require(curl)
require(XML)
require(testthat)
```

1. This problem explores file sizes and their relationship to the file format.

   (a) **Problem 1a SOLUTION**

   Flat text file tmp1.csv records the 1 million randomly generated letters as a vector (1 column matrix) of individual single character values. tmp2.csv collapsed the single character elements into one long string, and therefore is only coding one (long) element. Collapsing the single character elements removes newline characters, essentially, in this case removing half the file size. The tmp3.Rda file contains a numeric vector with one million values, stored in the R binary format. tmp4.csv contains the same values in ASCII format. tmp5.csv contains the same values (rounded to the nearest 0.01) also in ASCII format. Because tmp4.csv was saved in ASCII format, it has the largest filesize; whereas tmp5.csv has the smallest filesize because each of the numeric values contains fewer digits. This file is smaller than tmp3.Rda and tmp4.csv simply because there are many million fewer characters in tmp5.csv than the other two files.

```
##########
# Chunk 2: Question 1a code
#########

## save letters in text format
chars <- sample(letters, 1e6, replace = TRUE)
write.table(chars, file = 'tmp1.csv', row.names = FALSE, quote = FALSE,
col.names = FALSE)
system('ls -l tmp1.csv', intern = TRUE)
## [1] "-rw-r--r-- 1 paciorek scfstaff 2000000 Sep 11 07:38 tmp1.csv"
chars <- paste(chars, collapse = '')
write.table(chars, file = 'tmp2.csv', row.names = FALSE, quote = FALSE,
col.names = FALSE)
```

```
system('ls -l tmp2.csv', intern = TRUE)
## [1] "-rw-r--r-- 1 paciorek scfstaff 1000001 Sep 11 07:38 tmp2.csv"
## save in binary format
nums <- rnorm(1e6)
save(nums, file = 'tmp3.Rda')
system('ls -l tmp3.Rda', intern = TRUE)
## [1] "-rw-r--r-- 1 paciorek scfstaff 7678421 Sep 11 07:38 tmp3.Rda"
## save in text format
write.table(nums, file = 'tmp4.csv', row.names = FALSE, quote = FALSE,
col.names = FALSE, sep = ',')
system('ls -l tmp4.csv', intern = TRUE)
## [1] "-rw-r--r-- 1 paciorek scfstaff 18158912 Sep 11 07:38 tmp4.csv"
write.table(round(nums, 2), file = 'tmp5.csv', row.names = FALSE,
quote = FALSE, col.names = FALSE, sep = ',')
system('ls -l tmp5.csv', intern = TRUE)
## [1] "-rw-r--r-- 1 paciorek scfstaff 5379375 Sep 11 07:38 tmp5.csv"
```

(b) **Problem 1b SOLUTION**

In the below code, save() is being implemented with the default options, which writes data to R binary file format and compresses the data using gzip compression (level 6). Simply, tmp7.Rda is must smaller than tmp6.Rda because the tmp7.Rda data (one million a's) is much more compressable than the tmp6.Rda data which is comprised of one million alphabetic characters in random order. Compressing a uniform sequence only requies saving on instance of teh sequence and how many times its repeated, while compressing a random sequence requires much more information.

```
##########
# Chunk 3: Question 1b code
#########
chars <- sample(letters, 1e6, replace = TRUE)
chars <- paste(chars, collapse = '')
save(chars, file = 'tmp6.Rda')
system('ls -l tmp6.Rda', intern = TRUE)
## [1] "-rw-r--r-- 1 paciorek scfstaff 635237 Sep 11 07:38 tmp6.Rda"
chars <- rep('a', 1e6)
chars <- paste(chars, collapse = '')
save(chars, file = 'tmp7.Rda')
system('ls -l tmp7.Rda', intern = TRUE)
## [1] "-rw-r--r-- 1 paciorek scfstaff 1056 Sep 11 07:38 tmp7.Rda"
```

2. Go to https://scholar.google.com and enter the name ...

(a) **Problem 2a SOLUTION**

The function `getCitations()`, takes as input, one character string: input=`Geoffrey Hinton` and returns the Google scholar ID: `JicYpdAAAAAJ`, and that person's citation page HTML.

```
##########
# Chunk 4: Question 2a code
#########
#URLs used
#https://scholar.google.com/
#https://scholar.google.com/scholar?hl=en&q=Geoffrey+Hinton&as_sdt=1%2C5&as_sdtp=&oq=
#https://scholar.google.com/citations?view_op=search_authors&mauthors=Geoffrey+Hinton&hl=en&oi

getCitations <- function(name) {
```

```
  require(testthat)
  require(stringr)
  require(curl)
  require(XML)

  #check if valid input
  #is character?
  stopifnot(test_that("Input must be character string",
                      expect_equal(class(name), "character")))
  #is first last format?
  stopifnot(test_that("input must be name: e.g.'Albert Einstein'",
                      expect_true(str_detect(name,
                                             "^[[:Alpha:]]+ [[:Alpha:]]+$"))))

  #get names
  name1 = gsub("([A-Za-z]*)[[:space:]]*([A-Za-z]*)[[:space:]]*([A-Za-z]*)","\\1",name)
  name2 = gsub("([A-Za-z]*)[[:space:]]*([A-Za-z]*)","\\2",name)

  #search for scholar
  url_name   = paste0(name1,"+",name2) #name format for url search
  googleURL  = "https://scholar.google.com/citations?view_op=search_authors&mauthors="
  searchURL  = paste0(googleURL,url_name) #search name
  html       = readLines(searchURL,warn=F)
  links      = getHTMLLinks(html)

  #find Google Scholar ID
  citePage = grep("user=",links,value=T)[1] #get link for cite page
  id       = gsub(".*([A-Za-z0-9-]{12})&.*","\\1",citePage) #Gscholar id

  #check if google scholar ID is valid
  stopifnot(test_that("Invalid, or no google scholar ID for input.",
                      expect_true(str_detect(id,"^[A-Za-z0-9-]{12}$"))))

  #get citation page
  citePageURL  = paste0("https://scholar.google.com",citePage) #citation page
  citePageHTML = readLines(citePageURL,warn=F) #get HTML for citation page
  Sys.sleep(2) # pause to prevent capcha

  #return scholar it, cit page HTML, and the user input
  return(list(id           = id,
              citePageHTML = citePageHTML,
              input        = name)) #return google scholar ID and citation page HTML
}

#test function
  testName = "Geoffrey Hinton" #user input
  out1     = getCitations(testName) #run function
    attributes(out1) #attributes of getCitations output
```

```
## $names
## [1] "id"           "citePageHTML" "input"
```

```
    out1$id #scholar ID
```

```
## [1] "JicYPdAAAAAJ"
```

```
    out1$input #user input
```

```
## [1] "Geoffrey Hinton"
```

```
    substr(out1$citePageHTML,start=0,stop=40) #first html 40 characters
```

```
## [1] "<!doctype html><head><meta http-equiv=\"C"
```

```
  #error checking
  testError = c("Cameron Adams",
                "Geoffrey",
                "A1b3rt Einstein")
  outError1 = getCitations(testError[1]) #Error: person has no scholar ID
```

```
## Error:  Test failed:  'Invalid, or no google scholar ID for input.'
## * str_detect(id, "^[A-Za-z0-9-]{12}$") isn't true.
```

```
  outError2 = getCitations(testError[2]) #Error: only one name
```

```
## Error:  Test failed:  'input must be name:  e.g.'Albert Einstein''
## * str_detect(name, "^[[:Alpha:]]+ [[:Alpha:]]+$") isn't true.
```

```
  outError3 = getCitations(testError[3]) #Error: numeric characters
```

```
## Error:  Test failed:  'input must be name:  e.g.'Albert Einstein''
## * str_detect(name, "^[[:Alpha:]]+ [[:Alpha:]]+$") isn't true.
```

(b) **Problem 2b ANSWER**

The function `processCitations()`, takes as input, the HTML for the citation page from `getCitations()` and returns a table of their first 20 citations on google scholar.

```
############
# Chunk 5: Question 2b code
############

processCitations <- function(html) {
  require(XML)
  #check ifinput is character string
  stopifnot(test_that("Input must be HTML code as character string",
                      expect_equal(class(html), "character"))) # is.character?

  #html=out1£citePageHTML #test

  #scrape citations using XML
  tmp  = htmlParse(html) #parse html string
  tmp2 = getNodeSet(tmp,"//table") #get table nodes
  tmp3 = xmlToList(tmp2[[2]]) #convert to list
  tbl  = tmp3$tbody #extract table

  #extract title, author, journal info
  title<-author<-journal<-year<-citations<-list() #empty lists for cite fields
  tbl_rows=length(tbl)-1 #num citations
  for (i in 1:tbl_rows) {
    title[[i]]      = tbl[[i]][[1]]$a$text
    author[[i]]     = tbl[[i]][[1]][[2]]$text
    journal[[i]]    = tbl[[i]][[1]][[3]][[1]]
    citations[[i]]  = tbl[[i]][[2]]$a$text
```

```
    year[[i]]       = tbl[[i]][[3]][[1]][[1]]
  }

  #combine into table
  output=data.frame(title     = unlist(title),
                    author    = unlist(author),
                    journal   = unlist(journal),
                    year      = unlist(year),
                    citations = unlist(citations))

  return(output)
  Sys.sleep(2) #pause to prevent google from adding capcha
}

#test output
  out1tbl=processCitations(out1$citePageHTML) #input citation page HTML from getCitations()
    dim(out1tbl) #table dimensions
```

```
## [1] 20  5
```

```
    head(out1tbl) #first few lines
```

```
##                                                              title
## 1             Learning representations by back-propagating errors
## 2          Learning internal representations by error-propagation
## 3          Learning internal representations by error propagation
## 4                                 Parallel distributed processing
## 5 Imagenet classification with deep convolutional neural networks
## 6                  A fast learning algorithm for deep belief nets
##                                           author
## 1            DE Rumelhart, GE Hinton, RJ Williams
## 2            DE Rumelhart, GE Hinton, RJ Williams
## 3            DE Rumelhart, GE Hinton, RJ Williams
## 4 DE Rumelhart, JL McClelland, PDP Research Group
## 5            A Krizhevsky, I Sutskever, GE Hinton
## 6                  GE Hinton, S Osindero, YW Teh
##                                                                    journal
## 1                                                          Nature 323, 533-536
## 2 Parallel Distributed Processing: Explorations in the Microstructure of ...
## 3                                CALIFORNIA UNIV SAN DIEGO LA JOLLA INST FOR
## 4                                                          MIT press 1, 184
## 5              Advances in neural information processing systems, 1097-1105
## 6                                      Neural computation 18 (7), 1527-1554
##   year citations
## 1 1986     34900
## 2 1986     27417
## 3 1985     23094
## 4 1987     18726
## 5 2012     15040
## 6 2006      6618
```

```
#test on different researcher
  #get citations
  out2=getCitations("Geoffrey Burnstock")
    out2$id #scholar id
```

```
## [1] "-T-OJAIAAAAJ"
```

```
    out2$input #user input
```

```
## [1] "Geoffrey Burnstock"
```

```
    substr(out2$citePageHTML,start=0,stop=40) #first html 40 characters
```

```
## [1] "<!doctype html><head><meta http-equiv=\"C"
```

```
  #get citation table
  out2tbl=processCitations(out2$citePageHTML)
    dim(out2tbl)
```

```
## [1] 20  5
```

```
    head(out2tbl)
```

```
##                                                                  title
## 1                            Receptors for purines and pyrimidines
## 2                                                    Purinergic nerves
## 3                Nomenclature and classification of purinoceptors.
## 4 Is there a basis for distinguishing two types of P2-purinoceptor?
## 5    Physiology and pathophysiology of purinergic neurotransmission
## 6       A basis for distinguishing two types of purinergic receptor
##                                                            author
## 1                                          V Ralevic, G Burnstock
## 2                                                      G Burnstock
## 3 BB Fredholm, MP Abbracchio, G Burnstock, JW Daly, TK Harden, ...
## 4                                          G Burnstock, C Kennedy
## 5                                                      G Burnstock
## 6                                                      G Burnstock
##                                                                journal
## 1                                  Pharmacological reviews 50 (3), 413-492
## 2                                  Pharmacological reviews 24 (3), 509-581
## 3                                  Pharmacological reviews 46 (2), 143-156
## 4                       General Pharmacology: The Vascular System 16 (5), 433-440
## 5                                    Physiological reviews 87 (2), 659-797
## 6 Cell membrane receptors for drugs and hormones: a multidisciplinary approach ...
##   year citations
## 1 1998      4243
## 2 1972      2418
## 3 1994      1748
## 4 1985      1332
## 5 2007      1267
## 6 1978      1196
```

(c) Incorporated into answers for 2(a) and 2(b).

(d) **Problem 2b ANSWER**

EXTRA CREDIT. When you click *SHOW MORE*, you can get the user profile citation page to show 100 citations per page: 0-100, 100-200, 200-300, etc: `https://scholar.google.com/citations?user=JicYPdAAAAAJ&hl=en&cstart=0&pagesize=100`. So we must specificy an intial start and pagesize values (0,100) and then iterate (start_i=start_i+100) until we reach the end of the available citations and google scholar no longer produces an html table.

```
############
# Chunk 6: Question 2d code
############

processCitationsALL = function(id) {

  library(XML)

  # pull html from first page of google scholar id
  googleURL  = "https://scholar.google.com/citations?user="
  start      = 0
  pagesize   = 100
  id         = id #google scholar id
  searchURL  = paste0(googleURL,
                      id,
                      "&hl=en&cstart=",
                      start,
                      "&pagesize=",
                      pagesize)
  html=readLines(searchURL,warn=F)


  #create while loop to iterate scraping of citations until no more citations
  htmlTable = NULL #start with empty html table data
  tbl = data.frame()  #start with empty table dataframe
  while(is.null(htmlTable)) {

    #get table and append to existing table data
    tmp   = processCitations(html) # get table
    tbl   = rbind(tbl, tmp) # append to existing table data

    # pull html for next page of 100 papers
    start = start+100  #update start position
    searchURL = paste0(googleURL,
                      id,
                      "&hl=en&cstart=",
                      start,
                      "&pagesize=",
                      pagesize)
    html=readLines(searchURL,warn=F)
    htmlTable = xmlToList(getNodeSet(htmlParse(html),
                                     "//table")[[2]])$tbody[[1]]$td$text
    Sys.sleep(2) #pause to prevent google from adding capcha
  }
  return(tbl) # return table of all papers on google scholar page
}

#test with Geoffrey Burnstock data.
tblAll=processCitationsALL(out2$id)
  dim(tblAll)

## [1] 2032    5

  head(tblAll)
```

```
##                                                                    title
## 1                              Receptors for purines and pyrimidines
## 2                                                        Purinergic nerves
## 3                  Nomenclature and classification of purinoceptors.
## 4 Is there a basis for distinguishing two types of P2-purinoceptor?
## 5    Physiology and pathophysiology of purinergic neurotransmission
## 6       A basis for distinguishing two types of purinergic receptor
##                                                                  author
## 1                                              V Ralevic, G Burnstock
## 2                                                        G Burnstock
## 3 BB Fredholm, MP Abbracchio, G Burnstock, JW Daly, TK Harden, ...
## 4                                              G Burnstock, C Kennedy
## 5                                                        G Burnstock
## 6                                                        G Burnstock
##                                                                                journal
## 1                                            Pharmacological reviews 50 (3), 413-492
## 2                                            Pharmacological reviews 24 (3), 509-581
## 3                                            Pharmacological reviews 46 (2), 143-156
## 4                          General Pharmacology: The Vascular System 16 (5), 433-440
## 5                                              Physiological reviews 87 (2), 659-797
## 6 Cell membrane receptors for drugs and hormones: a multidisciplinary approach ...
##   year citations
## 1 1998      4243
## 2 1972      2418
## 3 1994      1748
## 4 1985      1332
## 5 2007      1267
## 6 1978      1196
```