Reading Fixed Record and Hierarchical Files in R

Len Greski
January 28, 2021

Level: basic / intermediate

ATL Users Group

Starting with the basics

```
textData <- "customer_id|gender|past_3_years_bike_related_purchases|DOB|job_industry_category|wealth_segment|owns_car|tenure|state
 1|Female| 93|19644|Health|Mass Customer|Yes|11|New South Wales
 2|Male| 81|29571|Financial Services|Mass Customer|Yes|16|New South Wales
 5|Female| 56|28258|n/a|Affluent Customer|Yes|8|New South Wales
 8|Male| 31|22735|n/a|Mass Customer| No|7|New South Wales
 9|Female| 97|26733|Argiculture|Affluent Customer|Yes| 8|New South Wales
12|Male| 58|34536|Manufacturing|Mass Customer| No| 8|QLD"

data <- read.csv(text = textData,
                 header = TRUE,
                 na.strings = c("n/a","na"),
                 sep="|")

data
```

```
> data
  customer_id gender past_3_years_bike_related_purchases   DOB job_industry_category    wealth_segment owns_car tenure           state
1           1 Female                                  93 19644                Health     Mass Customer       Yes     11 New South Wales
2           2   Male                                  81 29571    Financial Services     Mass Customer       Yes     16 New South Wales
3           5 Female                                  56 28258                  <NA> Affluent Customer       Yes      8 New South Wales
4           8   Male                                  31 22735                  <NA>     Mass Customer        No      7 New South Wales
5           9 Female                                  97 26733          Argiculture Affluent Customer       Yes      8 New South Wales
6          12   Male                                  58 34536        Manufacturing     Mass Customer        No      8             QLD
>
```

Source: [Remove NA values with tidyverse mutate]

ATL Users Group

What happens when the data looks like this?

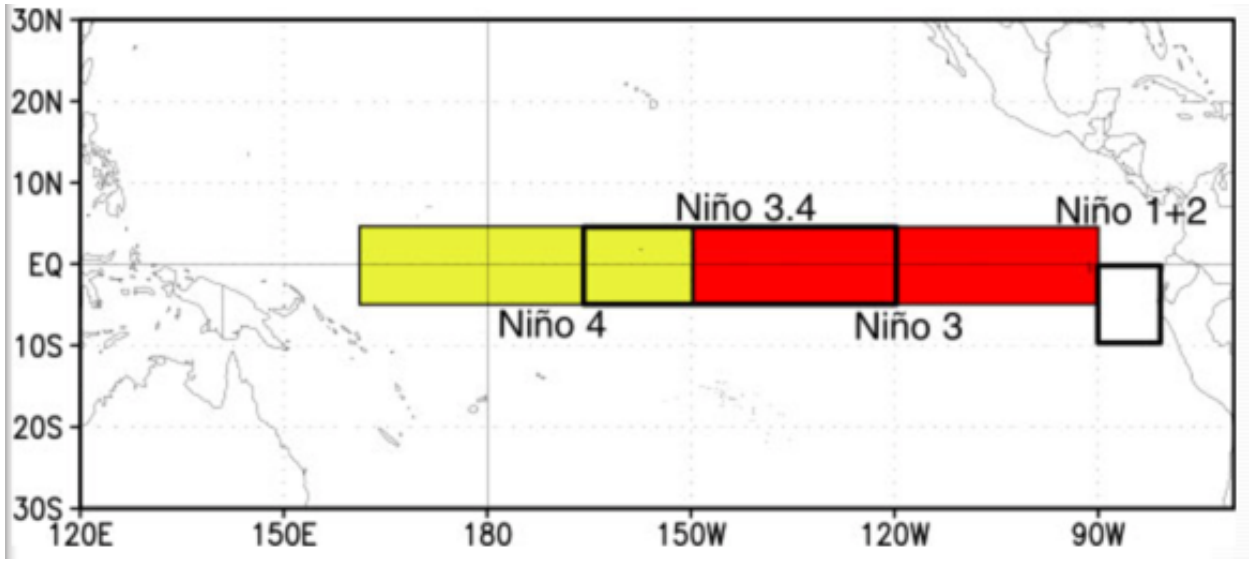
wksst8110.for

```
Weekly SST data starts week centered on 3Jan1990

                 Nino1+2       Nino3        Nino34        Nino4
Week           SST SSTA     SST SSTA     SST SSTA     SST SSTA
03JAN1990      23.4-0.4     25.1-0.3     26.6 0.0     28.6 0.3
10JAN1990      23.4-0.8     25.2-0.3     26.6 0.1     28.6 0.3
17JAN1990      24.2-0.3     25.3-0.3     26.5-0.1     28.6 0.3
24JAN1990      24.4-0.5     25.5-0.4     26.5-0.1     28.4 0.2
31JAN1990      25.1-0.2     25.8-0.2     26.7 0.1     28.4 0.2
07FEB1990      25.8 0.2     26.1-0.1     26.8 0.1     28.4 0.3
14FEB1990      25.9-0.1     26.4 0.0     26.9 0.2     28.5 0.4
21FEB1990      26.1-0.1     26.7 0.2     27.1 0.3     28.9 0.8
28FEB1990      26.1-0.2     26.7-0.1     27.2 0.3     29.0 0.8
07MAR1990      26.7 0.3     26.7-0.2     27.3 0.2     28.9 0.7
14MAR1990      26.1-0.4     26.9-0.2     27.3 0.1     28.6 0.4
21MAR1990      26.1-0.2     27.2 0.0     27.6 0.3     28.7 0.5
28MAR1990      25.7-0.4     27.5 0.2     27.8 0.3     28.8 0.5
04APR1990      25.6-0.3     27.6 0.3     27.9 0.4     28.8 0.4
```

- What does the .for file extension mean?
- How many columns?
- What separates the columns?
- How does one handle variable names for the data set?
- Why does government data have to be such a hassle to read?

Source: [NOAA Sea Surface Temperature Anomaly Readings, 1990 - present]

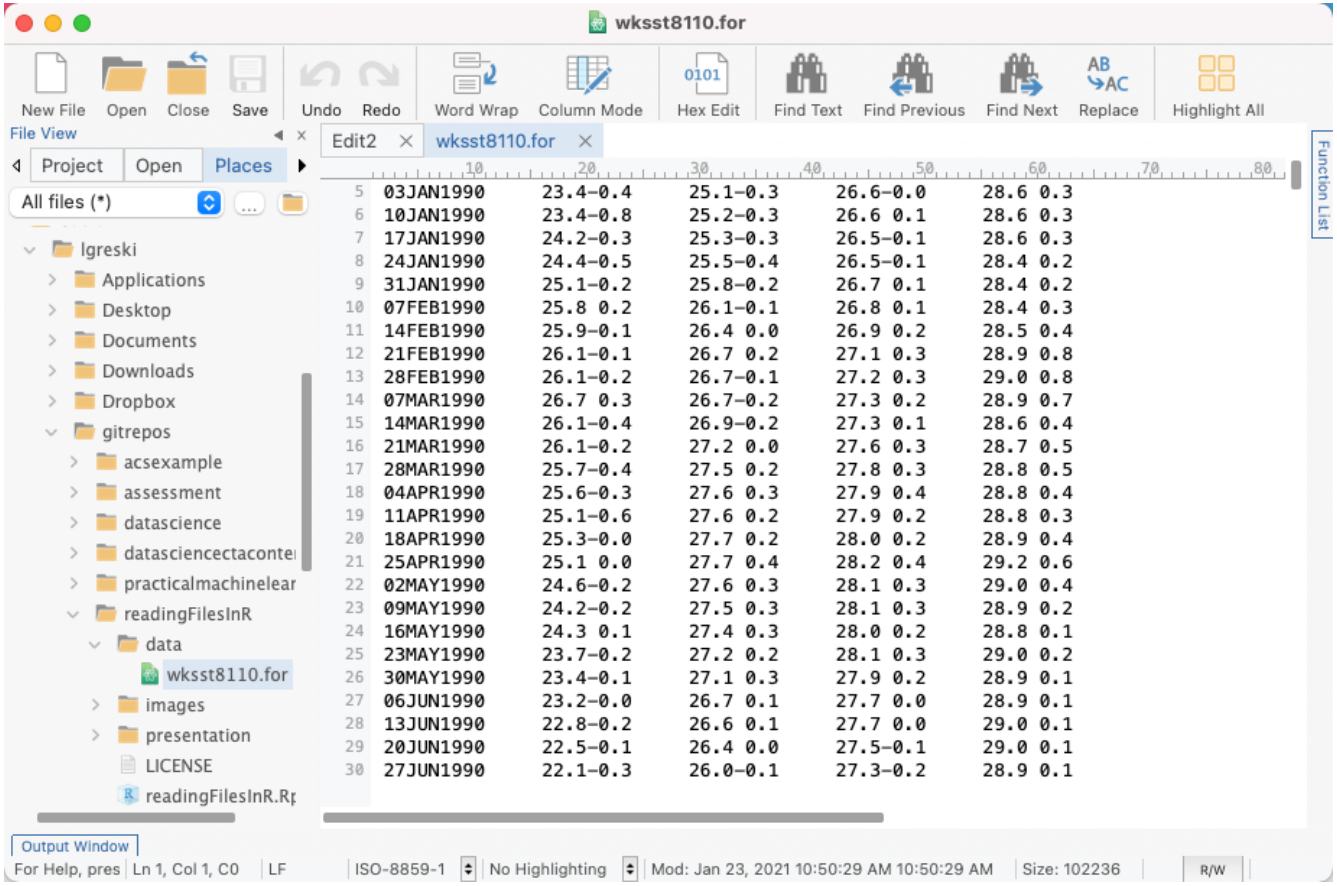
Background: El Niño Southern Oscillation



A combined atmospheric and ocean system consisting of four regions for which the NOAA collects data.

Reference: El Niño Southern Oscillation, NOAA Website

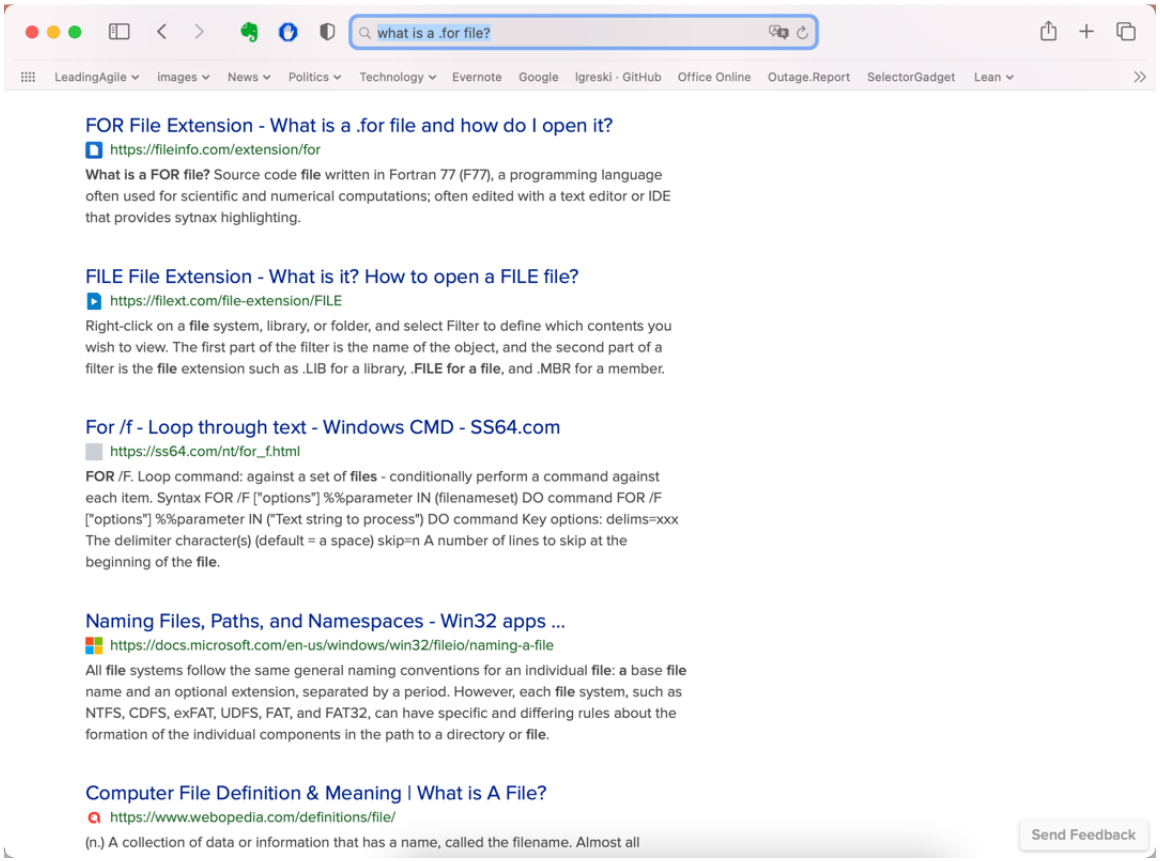
Identifying start and end columns




```
# 02 - 10 week as DDMMMYYYY
# 16 - 19 nino1and2sst
# 20 - 23 nino1and2ssta
# 29 - 32 nino3sst
# 33 - 36 nino3ssta
# 42 - 45 nino34sst
# 46 - 49 nino34ssta
# 55 - 58 nino4sst
# 59 - 62 nino4ssta
```

ATL R Users Group

What is a .for file?



ATL R Users Group

“Yes Virginia, there is a read.fortran()”

```
noaaSSTData <- "https://www.cpc.ncep.noaa.gov/data/indices/wksst8110.for"
download.file(noaaSSTData,"./data/wksst8110.for")
fileURL <- "./data/wksst8110.for"

# set addresses for fixed length fortran-style input file
theAddresses <- c("1X","A9","5X","2F4.0","5X","2F4.0","5X","2F4.0","5X","2F4.0")

# define column names
theColumns <- c("week","nino1and2sst","nino1and2ssta","nino3sst",
                "nino3ssta","nino34sst","nino34ssta",
                "nino4sst","nino4ssta")
df <- read.fortran(file=fileURL,theAddresses,skip = 4)
colnames(df) <- theColumns
head(df)
```

Purpose		*Edit Descriptors*	
Reading/writing **INTEGER**s		**I***w*	**I***w.m*
Reading/writing **REAL**s	Decimal form	**F***w.d*	
	Exponential form	**E***w.d*	**E***w.dEe*
	Scientific form	**ES***w.d*	**ES***w.dEe*
	Engineering form	**EN***w.d*	**EN***w.dEe*
Reading/writing **LOGICAL**s		**L***w*	
Reading/writing **CHARACTER**s		**A**	**A***w*
Positioning	Horizontal	***n*X**	
	Tabbing	**T***c*	**TL***c* and **TR***c*
	Vertical	**/**	
Others	Grouping	***r*(....)**	
	Format Scanning Control	**:**	
	Sign Control	**S**, **SP** and **SS**	
	Blank Control	**BN** and **BZ**	

Reference: [Michigan Technological University CS201 – Fortran Formats]

ATL R Users Group

…and the output

```
> head(df)
       week nino1and2sst nino1and2ssta nino3sst nino3ssta nino34sst nino34ssta
1 03JAN1990         23.4          -0.4     25.1      -0.3      26.6        0.0
2 10JAN1990         23.4          -0.8     25.2      -0.3      26.6        0.1
3 17JAN1990         24.2          -0.3     25.3      -0.3      26.5       -0.1
4 24JAN1990         24.4          -0.5     25.5      -0.4      26.5       -0.1
5 31JAN1990         25.1          -0.2     25.8      -0.2      26.7        0.1
6 07FEB1990         25.8           0.2     26.1      -0.1      26.8        0.1
  nino4sst nino4ssta
1     28.6       0.3
2     28.6       0.3
3     28.6       0.3
4     28.4       0.2
5     28.4       0.2
6     28.4       0.3
>
```

Alternatives: read.fwf() and read_fwf()

```
# read with base::read.fwf()
fwfCols <- c(-1,9,-5,4,4,-5,4,4,-5,4,4,-5,4,4)
df2 <- read.fwf(fileURL,widths=fwfCols,skip=4,
                    col.names=theColumns)

# read with readr::read_fwf()
library(readr)
df3 <- read_fwf(fileURL,fwf_cols(week=c(2,10),
                                 nino1and2sst=c(16,19),nino1and2ssta=c(20,23),
                                 nino3sst=c(29,32),nino3ssta=c(33,36),
                                 nino34sst=c(42,45),nino34ssta=c(46,49),
                                 nino4sst=c(55,58),nino4ssta=c(59,62)),
                skip=4)
```

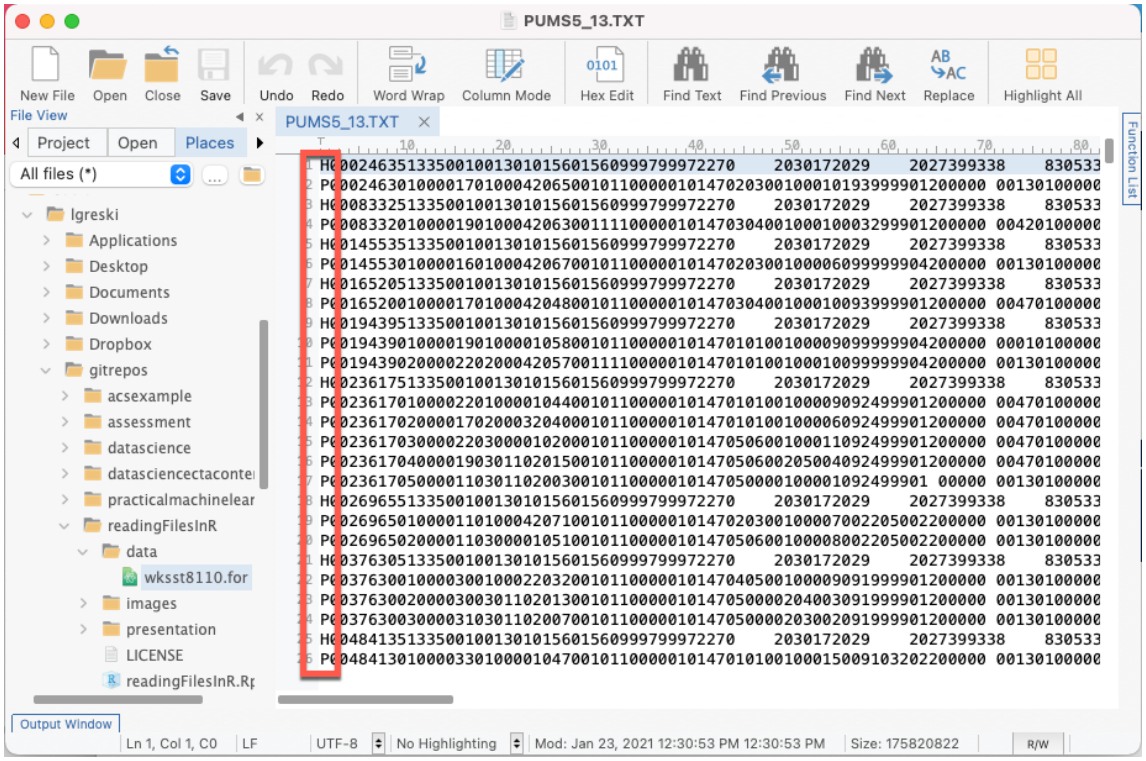
Checking our results…

```
library(testthat)
test_that("read.fortran equal to read.fwf",{
  expect_equal(nrow(df),nrow(df2))
  expect_equal(sum(df[["nino1and2sst"]]),sum(df2[["nino1and2sst"]]))
  expect_equal(sum(df[["nino1and2ssta"]]),sum(df2[["nino1and2ssta"]]))
  expect_equal(sum(df[["nino3sst"]]),sum(df2[["nino3sst"]]))
  expect_equal(sum(df[["nino3ssta"]]),sum(df2[["nino3ssta"]]))
  expect_equal(sum(df[["nino34sst"]]),sum(df2[["nino34sst"]]))
  expect_equal(sum(df[["nino34ssta"]]),sum(df2[["nino34ssta"]]))
  expect_equal(sum(df[["nino4sst"]]),sum(df2[["nino4sst"]]))
  expect_equal(sum(df[["nino4ssta"]]),sum(df2[["nino4ssta"]]))
})

test_that("read.fortran equal to read_fwf",{
  expect_equal(nrow(df),nrow(df3))
  expect_equal(sum(df[["nino1and2sst"]]),sum(df3[["nino1and2sst"]]))
  expect_equal(sum(df[["nino1and2ssta"]]),sum(df3[["nino1and2ssta"]]))
  expect_equal(sum(df[["nino3sst"]]),sum(df3[["nino3sst"]]))
  expect_equal(sum(df[["nino3ssta"]]),sum(df3[["nino3ssta"]]))
  expect_equal(sum(df[["nino34sst"]]),sum(df3[["nino34sst"]]))
  expect_equal(sum(df[["nino34ssta"]]),sum(df3[["nino34ssta"]]))
  expect_equal(sum(df[["nino4sst"]]),sum(df3[["nino4sst"]]))
  expect_equal(sum(df[["nino4ssta"]]),sum(df3[["nino4ssta"]]))
})
```

```
.../readingFilesInR/readSSTData.R
==> Testing R file using 'testthat'

== Testing readSSTData.R ================================
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 0 ]trying URL 'https://www.
cpc.ncep.noaa.gov/data/indices/wksst8110.for'
Content type 'text/plain; charset=UTF-8' length 102236 bytes
 (99 KB)
==========================================
downloaded 99 KB

[ FAIL 0 | WARN 0 | SKIP 0 | PASS 18 ] Done!

Test complete
```

ATL R Users Group

A hierarchical file: 2000 American Community Survey data



- Two types of records, Household and Person
- Data elements vary by record type
- Varying numbers of person records per household
- Data on each type of record is in fixed format

How does one go about reading and analyzing the person-level data?

ATL R Users Group

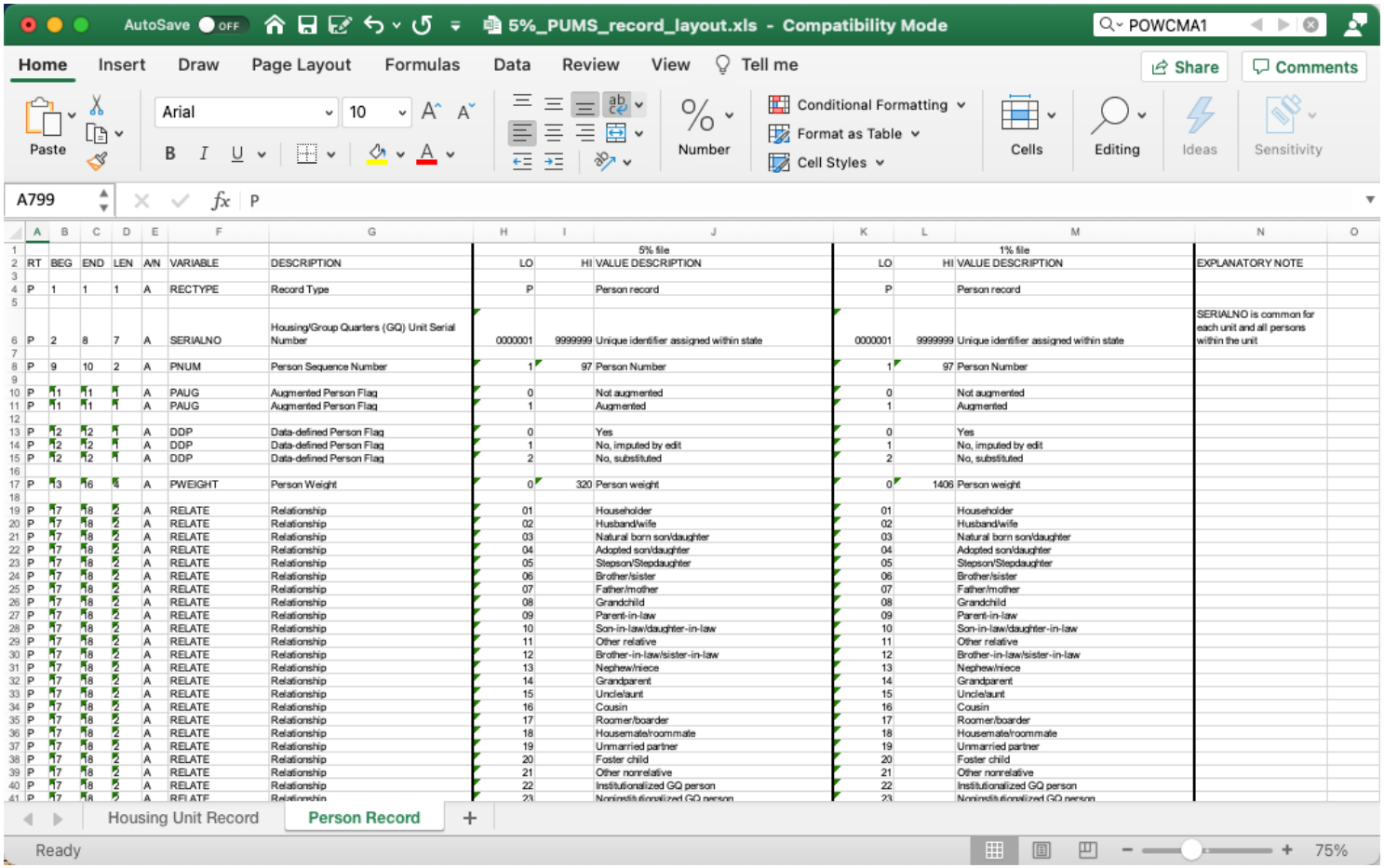
Background

Reference: [Public Use Microdata Sample, U. S. Census Bureau]

Considerations

- Lots of variables
- Large file size (Georgia file is 167Mb)
- Using the data dictionary to configure the data read function
- Separating 5% sample information from 1% sample information
- Eliminating value labels from the codebook

PUMS Codebook



AutoSave OFF — 5%_PUMS_record_layout.xls - Compatibility Mode — POWCMA1

Home Insert Draw Page Layout Formulas Data Review View Tell me — Share Comments

Paste — Arial — 10 — Number — Conditional Formatting — Format as Table — Cell Styles — Cells — Editing — Ideas — Sensitivity

A799 — P

									5% file				1% file	
RT	BEG	END	LEN	A/N	VARIABLE	DESCRIPTION	LO	HI	VALUE DESCRIPTION	LO	HI	VALUE DESCRIPTION	EXPLANATORY NOTE	
P	1	1	1	A	RECTYPE	Record Type	P		Person record	P		Person record		
P	2	8	7	A	SERIALNO	Housing/Group Quarters (GQ) Unit Serial Number	0000001	9999999	Unique identifier assigned within state	0000001	9999999	Unique identifier assigned within state	SERIALNO is common for each unit and all persons within the unit	
P	9	10	2	A	PNUM	Person Sequence Number	1	97	Person Number	1	97	Person Number		
P	11	11	1	A	PAUG	Augmented Person Flag	0		Not augmented	0		Not augmented		
P	11	11	1	A	PAUG	Augmented Person Flag	1		Augmented	1		Augmented		
P	12	12	1	A	DDP	Data-defined Person Flag	0		Yes	0		Yes		
P	12	12	1	A	DDP	Data-defined Person Flag	1		No, imputed by edit	1		No, imputed by edit		
P	12	12	1	A	DDP	Data-defined Person Flag	2		No, substituted	2		No, substituted		
P	13	16	4	A	PWEIGHT	Person Weight	0	320	Person weight	0	1406	Person weight		
P	17	18	2	A	RELATE	Relationship	01		Householder	01		Householder		
P	17	18	2	A	RELATE	Relationship	02		Husband/wife	02		Husband/wife		
P	17	18	2	A	RELATE	Relationship	03		Natural born son/daughter	03		Natural born son/daughter		
P	17	18	2	A	RELATE	Relationship	04		Adopted son/daughter	04		Adopted son/daughter		
P	17	18	2	A	RELATE	Relationship	05		Stepson/Stepdaughter	05		Stepson/Stepdaughter		
P	17	18	2	A	RELATE	Relationship	06		Brother/sister	06		Brother/sister		
P	17	18	2	A	RELATE	Relationship	07		Father/mother	07		Father/mother		
P	17	18	2	A	RELATE	Relationship	08		Grandchild	08		Grandchild		
P	17	18	2	A	RELATE	Relationship	09		Parent-in-law	09		Parent-in-law		
P	17	18	2	A	RELATE	Relationship	10		Son-in-law/daughter-in-law	10		Son-in-law/daughter-in-law		
P	17	18	2	A	RELATE	Relationship	11		Other relative	11		Other relative		
P	17	18	2	A	RELATE	Relationship	12		Brother-in-law/sister-in-law	12		Brother-in-law/sister-in-law		
P	17	18	2	A	RELATE	Relationship	13		Nephew/niece	13		Nephew/niece		
P	17	18	2	A	RELATE	Relationship	14		Grandparent	14		Grandparent		
P	17	18	2	A	RELATE	Relationship	15		Uncle/aunt	15		Uncle/aunt		
P	17	18	2	A	RELATE	Relationship	16		Cousin	16		Cousin		
P	17	18	2	A	RELATE	Relationship	17		Roomer/boarder	17		Roomer/boarder		
P	17	18	2	A	RELATE	Relationship	18		Housemate/roommate	18		Housemate/roommate		
P	17	18	2	A	RELATE	Relationship	19		Unmarried partner	19		Unmarried partner		
P	17	18	2	A	RELATE	Relationship	20		Foster child	20		Foster child		
P	17	18	2	A	RELATE	Relationship	21		Other nonrelative	21		Other nonrelative		
P	17	18	2	A	RELATE	Relationship	22		Institutionalized GQ person	22		Institutionalized GQ person		
P	17	18	2	A	RELATE	Relationship	23		Noninstitutionalized GQ person	23		Noninstitutionalized GQ person		

Housing Unit Record | Person Record | + — Ready — 75%

ATL R Users Group

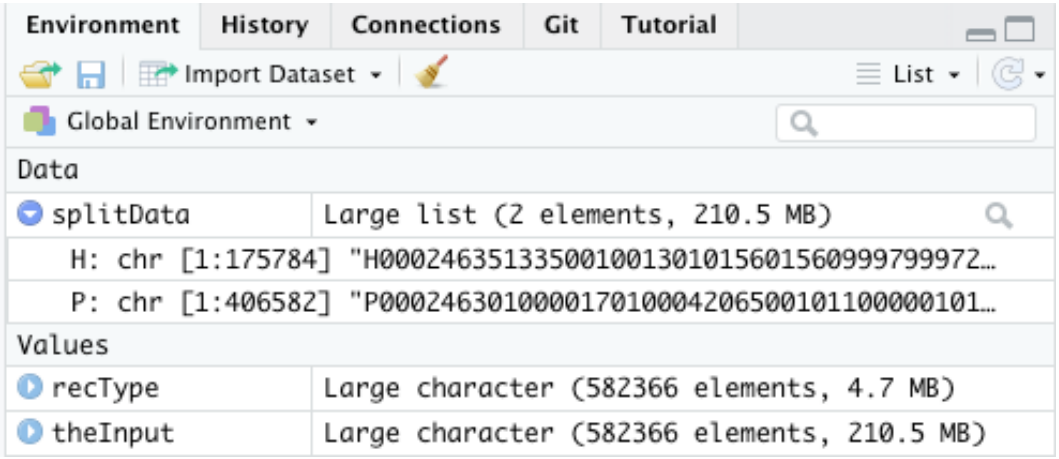
Obtaining the data

```
#
# read GA 2000 American Community Survey data

# create data directories if needed
if(!dir.exists("./data")) dir.create("./data")
if(!dir.exists("./data/Georgia")) dir.create("./data/Georgia")

# download & extract Georgia file if necessary
system.time(if(!file.exists("./data/Georgia/PUMS5_13.TXT")){
  download.file("https://www2.census.gov/census_2000/datasets/PUMS/FivePercent/Georgia/all_Georgia.zip",
                "./data/all_Georgia.zip",
                method="curl",
                mode="wb")
   unzip("./data/all_Georgia.zip",exdir="./data/Georgia")
})

# download record layout if necessary
if(!file.exists("./data/5%_PUMS_record_layout.xls")) {
  download.file("https://www2.census.gov/census_2000/datasets/PUMS/FivePercent/5%25_PUMS_record_layout.xls",
                "./data/5%_PUMS_record_layout.xls",
                method="curl",
                mode="wb")
}
```

Note: there is a revised PUMS file stored in the same location as the original, but the revised file is not in zip format so it takes about 10 minutes to download.

Read & split the records by record type

```
# separate person records from household records
system.time(theInput <- readLines("./data/Georgia/PUMS5_13.TXT",n = -1))
recType <- sapply(theInput,substr,1,1)
names(recType) <- NULL
splitData <- split(theInput,recType)
```



Alternate approach: don't try this at home

```
# legacy approach: write records to files and read the person file
# warning: this requires about 10 minutes of runtime for GA file
inFile <- "./data/Georgia/PUMS5_13.TXT"
outputPersonFile <- "./data/Georgia/PUMS_person.txt"
outputHouseholdFile <- "./data/Georgia/PUMS_household.txt"

system.time(theInput <- readLines(inFile,n = -1))
system.time(theResult <- lapply(theInput,function(x) {
  if(substr(x,1,1)=="P") {cat(x,file=outputPersonFile,sep="\n",append=TRUE)}
  else {cat(x,file=outputHouseholdFile,sep="\n",append=TRUE)}
}))
```

Read & clean the codebook

```
# read the code book person record columns through 5% value description
library(readxl)
cellRange <- "A2:J1219"
codeBook <- read_xls("./data/5%_PUMS_record_layout.xls",
                     sheet=2,
                     range=cellRange)

# fix data error in spreadsheet: missing value for RT column
codeBook$RT <- "P"

# remove blank rows and columns specific to 1% sample,
# then drop LO and VALUE DESCRIPTION. If the 5% sample LO column is blank
# the row belongs to the 1% sample only
codeBook <- codeBook[!is.na(codeBook$VARIABLE) & !is.na(codeBook$LO),][,1:7]
## remove duplicate rows
codeBook <- unique(codeBook)

## remove NA rows by setting length to a numeric variable, and processing
## with !is.na
codeBook$LEN <- as.numeric(codeBook$LEN)
codeBook <- codeBook[!is.na(codeBook$LEN),]
```

Use codebook to configure read_fwf()

```
## set widths vector to LEN (length) column
colWidths <- codeBook$LEN

## sum of lengths should be <= 316, per codebook
sum(codeBook$LEN)

## set column names to the VARIABLE column in codebook
colNames <- codeBook$VARIABLE
```

```
> ## set widths vector to LEN (length) column
> colWidths <- codeBook$LEN
>
> ## sum of lengths should be <= 316, per codebook
> sum(codeBook$LEN)
[1] 314
>
> ## set column names to the VARIABLE column in codebook
> colNames <- codeBook$VARIABLE
>
```

ATL Users Group

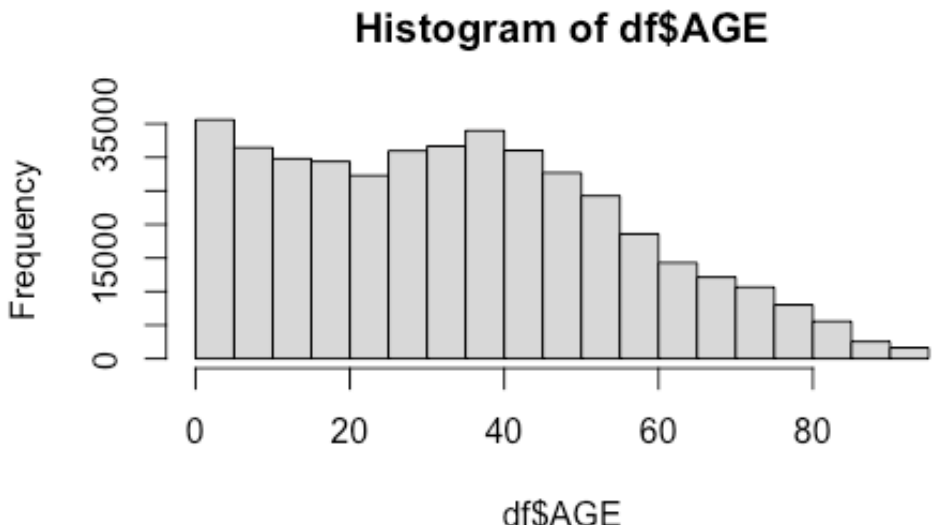
Read the file

```
library(readr)
system.time(df <- read_fwf(splitData[["P"]],
                fwf_widths(colWidths,col_names = colNames)))
```

```
> library(readr)
> system.time(df <- read_fwf(splitData[["P"]],
+                 fwf_widths(colWidths,col_names = colNames)))
|======================================================================| 100% 122 MB
   user  system elapsed
  7.858   0.438   8.174
> |
```

Run a simple analysis

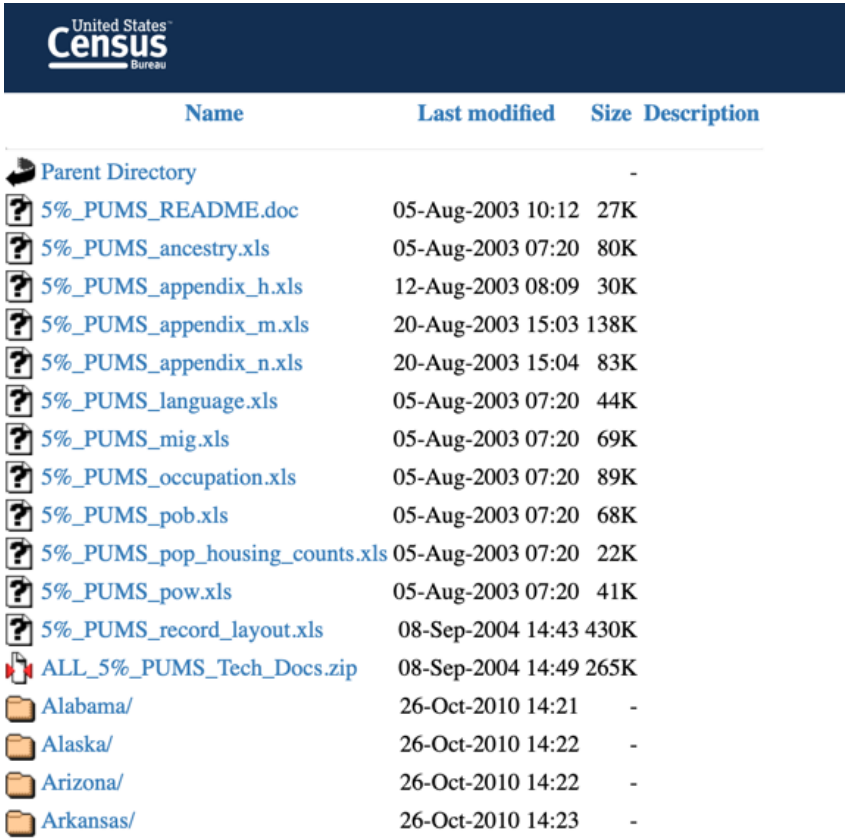
```
df$AGE <- as.numeric(df$AGE)
hist(df$AGE)
summary(df$AGE)
```




```
> summary(df$AGE)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.00   16.00   34.00   34.62   50.00   93.00
> 
```

ATL Users Group

Generalizing the solution


United States Census Bureau

Name	Last modified	Size	Description
Parent Directory		-	
5%_PUMS_README.doc	05-Aug-2003 10:12	27K	
5%_PUMS_ancestry.xls	05-Aug-2003 07:20	80K	
5%_PUMS_appendix_h.xls	12-Aug-2003 08:09	30K	
5%_PUMS_appendix_m.xls	20-Aug-2003 15:03	138K	
5%_PUMS_appendix_n.xls	20-Aug-2003 15:04	83K	
5%_PUMS_language.xls	05-Aug-2003 07:20	44K	
5%_PUMS_mig.xls	05-Aug-2003 07:20	69K	
5%_PUMS_occupation.xls	05-Aug-2003 07:20	89K	
5%_PUMS_pob.xls	05-Aug-2003 07:20	68K	
5%_PUMS_pop_housing_counts.xls	05-Aug-2003 07:20	22K	
5%_PUMS_pow.xls	05-Aug-2003 07:20	41K	
5%_PUMS_record_layout.xls	08-Sep-2004 14:43	430K	
ALL_5%_PUMS_Tech_Docs.zip	08-Sep-2004 14:49	265K	
Alabama/	26-Oct-2010 14:21	-	
Alaska/	26-Oct-2010 14:22	-	
Arizona/	26-Oct-2010 14:22	-	
Arkansas/	26-Oct-2010 14:23	-	

We can use the state names to drive an apply() function to download and process the data for multiple states.

Reference: https://www2.census.gov/census_2000/datasets/PUMS/FivePercent/

First, the setup

```
#
# download and read multiple states' PUMS data

theStates <- c("Alabama","Alaska","Arizona")
library(readr)
library(readxl)
# read and clean codebook
source("./readAndCleanCodebookPersonFile.R")

# create data directory if needed
if(!dir.exists("./data")) dir.create("./data")
```

Next, the load process driven by lapply()

```
dfList <- lapply(theStates,function(x){
  # create data directory if needed
  theDirectory <- paste0("./data/",x)
  if(!dir.exists(theDirectory)) dir.create(theDirectory)
  if(!file.exists(paste0("./data/all_",x,".zip"))) {
       download.file(paste0("https://www2.census.gov/census_2000/datasets/PUMS/FivePercent/",x,
                            "/all_",x,".zip"), paste0("./data/all_",x,".zip"),
                     method="curl",
                     mode="wb")
       unzip(paste0("./data/all_",x,".zip"),exdir=paste0("./data/",x))
  }
  # find correct file
  theFile <- list.files(path=theDirectory,pattern="^PUMS",full.names=TRUE)
  # separate person records from household records
  system.time(theInput <- readLines(theFile,n = -1))
  recType <- sapply(theInput,substr,1,1)
  names(recType) <- NULL
  splitData <- split(theInput,recType)
  df <- read_fwf(splitData[["P"]],
                              fwf_widths(colWidths,col_names = colNames))
  df$STATE <- x
  # write out data frame as RDS file, using state name
  saveRDS(df,paste0("./data/",x,"_person.RDS"))
  df # return data frame to parent environment
})
names(dfList) <- theStates
```

ATL Users Group

Q & A

ATL  Users Group

About Len

Len Greski currently serves as Principal Consultant at LeadingAgile, the leader in helping large companies generate economic value through agile transformation. Len started his career at Information Resources Inc., developing statistical and AI models to predict consumer behavior. He learned R in 2015 when he needed a way to analyze the value of a software portfolio without spending $9,000 on a copy of SAS. Len has mentored hundreds of thousands of students in the Johns Hopkins University Data Science Specialization on Coursera, having served as a Community Mentor since 2015. Len has a top 5% ranking on Stackoverflow.com, where he primarily answers questions about R.

len@greskilabs.com

lgreski

lgreski/readingFilesInR – repository where tonight's code and presentation are stored

lgreski

Data Science Depot blog: https://lgreski.github.io/datasciencedepot/

Len Greski
8,102
●2 ●13 ●27