Political Economy of Development:

Introduction to R

Eduardo Montero

April 21, 2023

Outline

1. Introduction to R and RStudio:
   - Object-Oriented Programming
   - Data Editing and Data Management
   - Variables

2. R: Subsetting, Merging, and Appending Data

3. Regression models in R

4. R: Basic Visualization

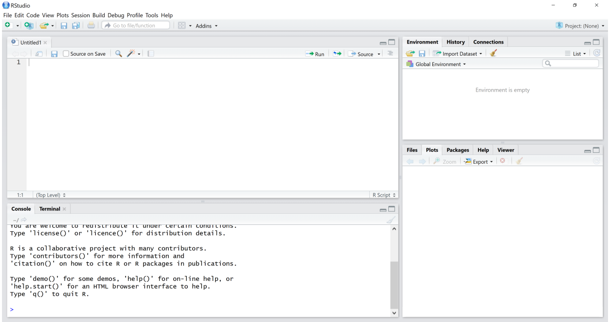
Quick Admin Comment

- To give you guys some flexibility (...especially this semester...), I will drop 3 lowest reading responses

Introduction to R

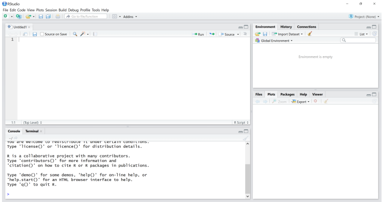
Installing R (and RStudio)

- **RStudio**: Hopefully, you all succeeded in downloading and installing R & RStudio. (Let me know if not!)
- RStudio with default settings should look something like this:



```
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

RStudio Layout



```
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

(I opened up a new R script so you can see the editor window.) Different from the Stata window, but many of the elements are the same:

- **Upper left**: Script editor - same as the .do file editor in Stata.
- **Bottom left**: Console, where you can type commands and see what happens, and it is also where you will see output from code you run in your script.
- **Bottom right**:Tabs for many things: any plots you make and help files you open.
- **Upper right**: list of current variables/data/objects are listed.

Getting Help in R

- Resources for learning R are abundant, and almost anything you would ever need to do has a solution somewhere.
- In addition, similar to Stata, all functions come with a help file.
- To do the same thing in R (say for the `table` command) you would type the following:

```r
# View Help File:
?table
```

- Or:

```r
# View Help File (Option 2):
help(table)
```

- Help file will appear on bottom right of RStudio

Object Oriented Coding

Before we get started, there is just one thing to cover

Object Oriented Coding

Before we get started, there is just one thing to cover Biggest

difference between Stata and R: R, like many other programming languages (e.g., Java, Python, C++), is object oriented. What

does this mean?

- If you use Stata, you used to thinking about an analysis in terms of actions.
  - You tabulate a categorical variable, you regress some dependent variable on independent variables, and you summarize continuous variables.

Object Oriented Coding

Before we get started, there is just one thing to cover Biggest

difference between Stata and R: R, like many other programming languages (e.g., Java, Python, C++), is object oriented. What

does this mean?

- If you use Stata, you used to thinking about an analysis in terms of actions.
    - You tabulate a categorical variable, you regress some dependent variable on independent variables, and you summarize continuous variables.
- In R, you need to think in terms of objects. For the same actions above, you create objects:
    - You create a table for a categorical variable, you create a regression model for some dependent variable and independent variables, and you create a summary for a continuous variable.

Object Oriented Coding - Example

- Example: in R, don't have to type "`display 2+2`", as you would need to do in Stata, in order to do the math
- In R, typing `2+2` into the console creates a numeric object,
- When an object is on a line by itself, R just spits back out what is contained in that object.
- But! You could assign $2+2$ to a named object, and it would store it for you to reference whenever you want.

Object Oriented Coding - Example

- Example: in R, don't have to type "`display 2+2`", as you would need to do in Stata, in order to do the math
- In R, typing 2+2 into the console creates a numeric object,
- When an object is on a line by itself, R just spits back out what is contained in that object.
- But! You could assign $2+2$ to a named object, and it would store it for you to reference whenever you want.
- To assign things to objects, we use the `<-` operator (the less-than symbol plus a hyphen). For example,

```
my_object <- 2+2
```

- And to display it's value, you can just run:

```
my_object

## [1] 4
```

Object Oriented Coding - Importance

- Most striking difference from Stata is that *your data is also an object*.
- This means is that you can have as many data sets open as your computer memory will allow, and they can all be stored in different objects with different names.
- This makes data management much easier in R (i.e. No more `preserve` and `restore` and opening and closing different files)

Data Management

- In R, there is a standard syntax for functions, where you have the function name, and then within parentheses, all inputs and options that go into that function, separated by commas.

Setting a working directory:

```
## Set Working Directory:
setwd("/Users/eduardomontero/Dropbox/")

## Error in setwd("/Users/eduardomontero/Dropbox/"):  cannot change
working directory
```

Getting a working directory:

```
## Current Working Directory:
getwd()

## [1] "/Users/emontero/Dropbox/Teaching/PEoDev_spring2023/R_Assignment/R_Intro
```

Reading in Data

There are a number of functions to read in data depending on the data type.

Usually, they have similar syntax.

For example, to read in an excel csv file:

```
my.data <- read.csv("./French_Cameroon_Outline.csv")
```

What is that “`my.data <-`” stuff at the beginning of the line?

Reading in Data

read.csv() doesn't actually load the data into your environment; it just opens the file and gives back its contents. If it doesn't have

an object to give the contents to, it will just spit out the contents of the file right to the console:

```
read.csv("./French_Cameroon_Outline.csv")

##        X      long       lat order  hole piece id group
## 1      1  9.831374  2.299243     1 FALSE     1  1   1.1
## 2      2 10.001819  2.319322     2 FALSE     1  1   1.1
## 3      3  9.834263  2.312374     3 FALSE     1  1   1.1
## 4      4  9.820998  2.421728     4 FALSE     1  1   1.1
## 5      5  9.822069  2.505676     5 FALSE     1  1   1.1
## 6      6  9.824918  2.540874     6 FALSE     1  1   1.1
## 7      7  9.852112  2.679885     7 FALSE     1  1   1.1
## 8      8  9.862276  2.718100     8 FALSE     1  1   1.1
## 9      9  9.879542  2.754499     9 FALSE     1  1   1.1
## 10    10  9.888001  2.822381    10 FALSE     1  1   1.1
## 11    11  9.898876  2.920175    11 FALSE     1  1   1.1
## 12    12  9.928156  2.992959    12 FALSE     1  1   1.1
## 13    13  9.943402  3.153113    13 FALSE     1  1   1.1
## 14    14  9.911057  3.234903    14 FALSE     1  1   1.1
## 15    15  9.859841  3.340097    15 FALSE     1  1   1.1
## 16    16  9.779333  3.433227    16 FALSE     1  1   1.1
```

Installing Packages & Reading in Stata Data

R has a number of packages/libraries that contain very useful functions. For example, to read in Stata data, there is a package called "`readstata13`".

Installing Packages & Reading in Stata Data

R has a number of packages/libraries that contain very useful functions. For example, to read in Stata data, there is a package called "`readstata13`". To *install* a package and the functions + help files in that package, run the following code:

```
install.packages("readstata13", repos="https://cloud.r-project.org")

## Installing package into '/Users/emontero/Library/R/arm64/4.1/library'
## (as 'lib' is unspecified)


##
## The downloaded binary packages are in
##  /var/folders/rd/pn_807d13b7drmg3q97694pm0000gq/T//RtmpuVHnX6/downloaded_pac
```

Installing Packages + Reading in Stata Data

To *load* an installed package, you include the following code before calling any functions from the package (usually at the top of the R script you write):

```
library(readstata13)

## Warning:  package 'readstata13' was built under R version 4.1.1
```

Or:

```
require(readstata13)
```

Installing Packages + Reading in Stata Data

Now, once loaded, can run the following to read in stata code:

```
mexico.census.2010 <- read.dta13(file="./mexico_munic_2010PopCensus.dta")
```

Installing Packages + Reading in Stata Data

Now, once loaded, can run the following to read in stata code:

```
mexico.census.2010 <- read.dta13(file="./mexico_munic_2010PopCensus.dta")
```

Now, mexico.census.2010 is your object containing the data:

- The data is from the 2010 Mexico Population Census.
- It is at the municipality-level: each observation (row) represents data for a municipality.

How do we view it?

Viewing Data

R has a number of useful features for viewing data

To see all of the variable names in a data object, you could type `names(mexico.census.2010)`:

```
names(mexico.census.2010)


##  [1] "stateid"                "mun"
##  [3] "nom_mun"                "pobtot"
##  [5] "logpop"                 "pctpea"
##  [7] "pctpe_inac"             "pcteconomidactivefem"
##  [9] "pcteconomidactivemasc" "pctpocupada"
## [11] "pctpdesocup"            "pctp3ym_hli"
## [13] "pctp3hlinhe"            "pctphog_ind"
## [15] "avgyrseduc"             "avgyrseducmasc"
## [17] "avgyearseducfem"        "pctp3a5_noa"
## [19] "pctp6a11_noa"           "pctp12a14noa"
## [21] "pctp15a17a"             "pctp18a24a"
## [23] "pctp8a14illiterate"     "pctpilliterate"
## [25] "pctpnoschool"           "pctpsomeelementary"
## [27] "pctelementary"          "pctsomehighschool"
## [29] "pcthighschool"          "pctpcollege"
## [31] "pctpsinder"             "pctpder_ss"
## [33] "pctpder_iste"           "pctpder_istee"
## [35] "pctpder_segp"           "pctdirtfloor"
```

Viewing Data

Another very useful command is `head()`. This returns the first few rows of the data:

```
head(mexico.census.2010)

##   stateid mun               nom_mun  pobtot   logpop
## 1       9   2          Azcapotzalco  414711 12.93534
## 2       9   3              Coyoacan  620416 13.33815
## 3       9   4 Cuajimalpa de Morelos  186391 12.13560
## 4       9   5     Gustavo A. Madero 1185772 13.98590
## 5       9   6             Iztacalco  384326 12.85925
## 6       9   7            Iztapalapa 1815786 14.41203
##      pctpea pctpe_inac pcteconomidactivefem
## 1 0.5501565  0.4448176            0.2303411
## 2 0.5644862  0.4303903            0.2442410
## 3 0.5873938  0.4066882            0.2401707
## 4 0.5446249  0.4438270            0.2133334
## 5 0.5597216  0.4339091            0.2306973
## 6 0.5526652  0.4368862            0.2100045
##   pcteconomidactivemasc pctpocupada pctpdesocup pctp3ym_hli
## 1             0.3198154   0.5204677  0.02968876 0.007277357
## 2             0.3202452   0.5350439  0.02944232 0.013341049
## 3             0.3472231   0.5654005  0.02199327 0.011062765
## 4             0.3312915   0.5150440  0.02958087 0.012630590
## 5             0.3290243   0.5319392  0.02778239 0.009679283
## 6             0.3426607   0.5247431  0.02792215 0.016646234
```

Viewing Data

In RStudio, you can even view the data directly using `view`

Data Wrangling: Basics

Remember that in R, all data are stored in objects (like `mydata`), and you could in fact have multiple data sets in R at the same time.

Many ways to reference variables and rows of a data in R.

Basic: in R, a data set is a matrix, where the columns have names. Observations are in the rows, variables are in the columns, and each column has a name.

In R, you can reference specific rows and columns using a standard programming notation: `mydata[1, 1]`.

What happens if we type this in R?

Data Wrangling: Basics

Remember that in R, all data are stored in objects (like `mydata`), and you could in fact have multiple data sets in R at the same time.

Many ways to reference variables and rows of a data in R.

Basic: in R, a data set is a matrix, where the columns have names. Observations are in the rows, variables are in the columns, and each column has a name.

In R, you can reference specific rows and columns using a standard programming notation: `mydata[1, 1]`.

What happens if we type this in R?

```
mexico.census.2010[1,1]

## [1] 9
```

Data Wrangling: Basics

In R, because data sets have column names, you could also have written the column name instead of the column number.

One can write the same code, but substitute the second 1 with ”stateid”, as shown below:

```
mexico.census.2010[1,"stateid"]

## [1] 9
```

In fact, you can reference the full column by leaving out the row number: `mexico.census.2010[,‘‘stateid’’]`

This would give you the full variable: the list of all of the state IDs.

Data Wrangling: Variables

In R, an equivalent syntax for referencing an individual variable is shown below, where you use the dollar symbol:

```
mexico.census.2010$stateid
```

```
##   [1]  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9  9 11 11
##  [19] 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
##  [37] 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
##  [55] 11 11 11 11 11 11 11 11 12 12 12 12 12 12 12 12 12 12
##  [73] 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12
##  [91] 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12
## [109] 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12
## [127] 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 13
## [145] 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
## [163] 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
## [181] 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
## [199] 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
## [217] 13 13 13 13 13 13 13 13 13 13 13 14 14 14 14 14 14 14
## [235] 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
## [253] 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
## [271] 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
## [289] 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
## [307] 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
## [325] 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
## [343] 14 14 14 14 14 14 14 14 14 14 15 15 15 15 15 15 15 15
```

Creating Variables – Really Easy in R!

You can use the operator we saw recently to create new variables in an object:

```
mexico.census.2010$numtelevisions <-mexico.census.2010$pobtot*mexico.census.201
```

Dummy Variables

Dummy variables: a variable equal to 1 if a condition is met and 0 otherwise.

To generate them in R:

```
mexico.census.2010$largepop <- mexico.census.2010$pobtot > 41634
```

Replacing Data

To replace a set of values in R, you can do the following

```r
mexico.census.2010$largepop[mexico.census.2010$largepop==1] <- 10
```

Here we changed all values where `largepop` equals 1 to be 10 as an example

Introduction to R: Subsetting, Merging, and Appending Data

Data Wrangling: Referencing Subsets

To reference a set of *rows* (in this case, rows = *municipalities*) that meet certain conditions, you reference the condition as follows:

```
mexico.census.2010[,mexico.census.2010$stateid==3]

## data frame with 0 columns and 1188 rows
```

Note: This is similar to `browse if == ...` in Stata.

Data Wrangling: Referencing Subsets of Variables

To reference a *variable value* where conditions are met, you type:

```
mexico.census.2010$stateid[mexico.census.2010$pobtot> 1000000]

## [1]  9  9 11 14 14 15 15 21
```

Data Wrangling: Referencing Subsets of Variables

To reference a *variable value* where conditions are met, you type:

```
mexico.census.2010$stateid[mexico.census.2010$pobtot> 1000000]

## [1]  9  9 11 14 14 15 15 21
```

⇒ Why do we only have one parameter in the brackets?

Data Wrangling: Referencing Subsets of Variables

To reference a *variable value* where conditions are met, you type:

```
mexico.census.2010$stateid[mexico.census.2010$pobtot> 1000000]

## [1]  9  9 11 14 14 15 15 21
```

⇒ Why do we only have one parameter in the brackets?

Later on, we will learn other ways to work with subsets of data (more advanced data wrangling).

Data Wrangling: Referencing Subsets with `subset`

We can also use the (basic library) `subset()` function.

Broadly, the subset function has the following syntax:

```
subset([data object], subset=[keeping conditions],
select=[variables to keep])
```

Data Wrangling: Referencing Subsets II

Keeping and Dropping Variables: similar to keep in stata, you can use the “select” argument in subset to keep some variables in the data

```
munic.data.subset <- subset(mexico.census.2010,select=stateid)
```

Data Wrangling: Referencing Subsets II

Keeping and Dropping Variables: To keep more than one variable, just extract the specific columns by name and either store them in a new object or overwrite the old one:

```
munic.data.subset2 <-
  subset(mexico.census.2010,select=c(pobtot,logpop,
                                     pcttelevision))
```

This is the first time we've seen the `c()` function, so let's talk about it.

Data Wrangling: Referencing Subsets II

In Stata, when you use a list of things as an argument in a command, you just list the items with spaces in between.

Lists are a specific type of *object* in R,

You can create them using `c()`, which stands for “combine.”

```
c("pobtot","logpop","pcttelevision")

## [1] "pobtot"        "logpop"        "pcttelevision"
```

We can see if the subset worked with this list:

```
head(munic.data.subset2)

##    pobtot   logpop pcttelevision
## 1  414711 12.93534     0.9861418
## 2  620416 13.33815     0.9824797
## 3  186391 12.13560     0.9805332
## 4 1185772 13.98590     0.9815794
## 5  384326 12.85925     0.9847666
## 6 1815786 14.41203     0.9796938
```

Data Wrangling: Referencing Subsets II

Keeping and Dropping Variables: To drop variables, you can type the following:

```
munic.data.subset3 <-
  subset(mexico.census.2010,select=-c(pobtot,logpop,pcttelevision))
```

(Again, you don't have to create a new data object each time and you could just overwrite the old one - I'm just doing creating new objects here for demonstration purposes.)

I just added the "-" symbol before the c() function. You can tell that it worked by looking at the variable names:

```
names(munic.data.subset3)

##  [1] "stateid"               "mun"
##  [3] "nom_mun"               "pctpea"
##  [5] "pctpe_inac"            "pcteconomidactivefem"
##  [7] "pcteconomidactivemasc" "pctpocupada"
##  [9] "pctpdesocup"           "pctp3ym_hli"
## [11] "pctp3hlinhe"           "pctphog_ind"
## [13] "avgyrseduc"            "avgyrseducmasc"
## [15] "avgyearseducfem"       "pctp3a5_noa"
```

Data Wrangling: Referencing Subsets III

Keeping and Dropping Observations: similar to `keep if` in stata, you can use the “subset” argument in `subset` to keep some observation in the data that fit certain criteria

```
munic.data.subset4 <- subset(mexico.census.2010,
                             subset= (mun==2 & stateid==9))
head(munic.data.subset4)

##   stateid mun      nom_mun pobtot   logpop    pctpea
## 1       9   2 Azcapotzalco 414711 12.93534 0.5501565
##   pctpe_inac pcteconomidactivefem pcteconomidactivemasc
## 1  0.4448176            0.2303411             0.3198154
##   pctpocupada pctpdesocup pctp3ym_hli pctp3hlinhe
## 1   0.5204677  0.02968876 0.007277357 2.65245e-05
##   pctphog_ind avgyrseduc avgyrseducmasc avgyearseducfem
## 1   0.0160859       10.8          11.12           10.53
##   pctp3a5_noa pctp6a11_noa pctp12a14noa pctp15a17a
## 1   0.2729374   0.01971134   0.03595081  0.8405031
##   pctp18a24a pctp8a14illiterate pctpilliterate pctpnoschool
## 1  0.4713126         0.01059999     0.01617128   0.02196638
##   pctpsomeelementary pctelementary pctsomehighschool
## 1         0.05012943     0.1186388        0.03695455
##   pcthighschool pctpcollege pctpsinder pctpder_ss
```

Data Wrangling: Referencing Subsets III

Keeping and Dropping Observations: to drop observations, we use `subset` as well but change the logical conditions:

```r
munic.data.subset4 <- subset(mexico.census.2010,
                             subset= !(mun==2 & stateid==9))
```

Here, we use the `!` logical operator (for "not").

Merging and Appending Data

Merging data is much easier in R because you can load multiple data objects at once

Because of the object oriented way of thinking, you read the two data sets into R first.

Let's create two fake separated data sets, using what we learned previously:

```r
munic.data.merge1 <- subset(mexico.census.2010,select= c("stateid", "mun"))
munic.data.merge2 <- subset(mexico.census.2010,
                            select=  c("stateid","mun", "nom_mun"))
```

Merging and Appending Data

The function for merging in R is `merge()`.

For R, it doesn't really matter if the merging is one-to-one, one-to-many, or many-to-many. It will figure it out.

(Importantly, the many-to-many functionality in R is more intuitive for R than it is for Stata: it actually just creates multiplicative matches.)

The simplest syntax in R for merging the two data sets we created

```
mergeddata <- merge(munic.data.merge1, munic.data.merge2)
names(mergeddata)

## [1] "stateid" "mun"     "nom_mun"
```

Merging and Appending Data

`merge()` has a number of other options.

You can specify the specific variables you want to merge on using the "by" option.

```
mergeddata <- merge(munic.data.merge1, munic.data.merge2, by=c("stateid","mun")
```

As a default, `merge` keeps only observation that are in both data sets by default (also known as "inner join").

You can also include options to do different types of joins, where adding `all.x=TRUE` keeps all the observations (even unmatched ones) from the first dataset listed, `all.y=TRUE` keeps all the observations in the second dataset, and `all=TRUE` keeps all observations from both data sets

Merging and Appending Data

Appending Data: `rbind()` is the function you would use to append observation ("row bind") as long as the variable names are the same.

Basic Regression Analysis in R

Linear Regressions in R

The function for a linear regression model is `lm()`, which stands for “linear model.”

Linear Regressions in R

The function for a linear regression model is `lm()`, which stands for “linear model.”

Say we want a model predicting pctpe_inac (unemployed rate) from pobtot and avgyrseduc? (Made up example.)

Linear Regressions in R

The function for a linear regression model is `lm()`, which stands for “linear model.”

Say we want a model predicting pctpe_inac (unemployed rate) from pobtot and avgyrseduc? (Made up example.)

```
lm(pctpe_inac ~ pobtot + avgyrseduc,data=mexico.census.2010)

##
## Call:
## lm(formula = pctpe_inac ~ pobtot + avgyrseduc, data = mexico.census.2010)
##
## Coefficients:
## (Intercept)       pobtot   avgyrseduc
##   6.948e-01   -2.736e-08   -2.511e-02
```

The syntax is similar to Stata’s, but with the tilde separating the dependent variable from the independent variables, and the independent variables are separated by plus signs.

Linear Regressions in R

But the `lm` function creates a “linear model object” that contains a lot of information.

The previous code gives you coefficients, but not much else.

You can use the summary command to get much more information:

Linear Regressions in R

But the `lm` function creates a “linear model object” that contains a lot of information.

The previous code gives you coefficients, but not much else.

You can use the summary command to get much more information:

```
model1 <- lm(pctpe_inac ~ pobtot + avgyrseduc,data=mexico.census.2010)
summary(model1)

##
## Call:
## lm(formula = pctpe_inac ~ pobtot + avgyrseduc, data = mexico.census.2010)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.251452 -0.022764 -0.000025  0.021102  0.212463
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.948e-01  6.207e-03  111.92  < 2e-16 ***
## pobtot      -2.736e-08  9.059e-09   -3.02  0.00258 **
## avgyrseduc  -2.511e-02  9.193e-04  -27.31  < 2e-16 ***
## ---
```

Regression Practice: Washington (2008) Results

- Ebonya Washington (2008) shows that U.S. congressman tend to vote more liberally the more daughters they have (controlling for the total number of children)

- We will replicate her main results today

Regression Practice: Washington (2008) Results

- Ebonya Washington (2008) shows that U.S. congressman tend to vote more liberally the more daughters they have (controlling for the total number of children)

- We will replicate her main results today

- We can load her main dataset: congress-member level data set with voting records linked to their family histor for 4 waves of congress

- In other words, each row = 1 congress-member. Columns include: voting record scores, number of daughters, number of total children, etc.

```
### Load Washington (2008) Dataset:
congress_voting <- read.dta13("basic.dta")
```

Regression Practice: Washington (2008) Results

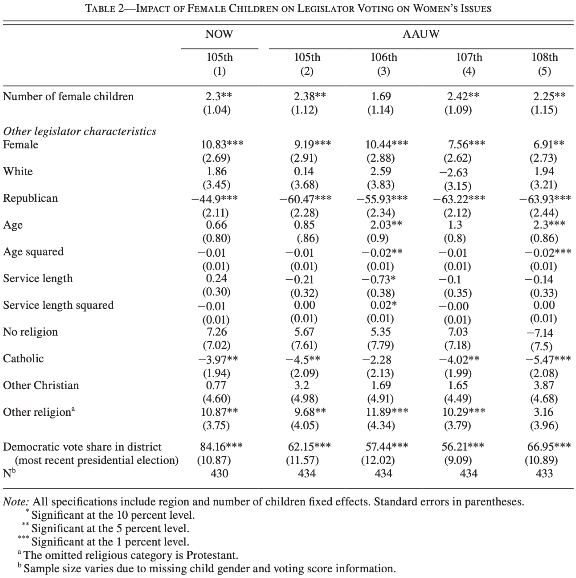


Table 2—Impact of Female Children on Legislator Voting on Women's Issues

	NOW	AAUW			
	105th	105th	106th	107th	108th
	(1)	(2)	(3)	(4)	(5)
Number of female children	2.3**	2.38**	1.69	2.42**	2.25**
	(1.04)	(1.12)	(1.14)	(1.09)	(1.15)
Other legislator characteristics					
Female	10.83***	9.19***	10.44***	7.56***	6.91**
	(2.69)	(2.91)	(2.88)	(2.62)	(2.73)
White	1.86	0.14	2.59	−2.63	1.94
	(3.45)	(3.68)	(3.83)	(3.15)	(3.21)
Republican	−44.9***	−60.47***	−55.93***	−63.22***	−63.93***
	(2.11)	(2.28)	(2.34)	(2.12)	(2.44)
Age	0.66	0.85	2.03**	1.3	2.3***
	(0.80)	(.86)	(0.9)	(0.8)	(0.86)
Age squared	−0.01	−0.01	−0.02**	−0.01	−0.02***
	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)
Service length	0.24	−0.21	−0.73*	−0.1	−0.14
	(0.30)	(0.32)	(0.38)	(0.35)	(0.33)
Service length squared	−0.01	0.00	0.02*	−0.00	0.00
	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)
No religion	7.26	5.67	5.35	7.03	−7.14
	(7.02)	(7.61)	(7.79)	(7.18)	(7.5)
Catholic	−3.97**	−4.5**	−2.28	−4.02**	−5.47***
	(1.94)	(2.09)	(2.13)	(1.99)	(2.08)
Other Christian	0.77	3.2	1.69	1.65	3.87
	(4.60)	(4.98)	(4.91)	(4.49)	(4.68)
Other religion[a]	10.87**	9.68**	11.89***	10.29***	3.16
	(3.75)	(4.05)	(4.34)	(3.79)	(3.96)
Democratic vote share in district	84.16***	62.15***	57.44***	56.21***	66.95***
(most recent presidential election)	(10.87)	(11.57)	(12.02)	(9.09)	(10.89)
N[b]	430	434	434	434	433

Note: All specifications include region and number of children fixed effects. Standard errors in parentheses.

* Significant at the 10 percent level.

** Significant at the 5 percent level.

*** Significant at the 1 percent level.

[a] The omitted religious category is Protestant.

[b] Sample size varies due to missing child gender and voting score information.

Regression Practice: Washington (2008) Replication

Exercise: Try running a Simple Regression, without additional controls, looking at the relationship betwee NOW scores (`nowtot`) and the number of daughers (`ngirls`)

What do you find?

What happens if you control for the total number of children? (`totchi`)

Why do we need to include controls for the number of children? (i.e. why does this simple regression suffer from `omitted variable bias`?)

Regression Practice: Washington (2008) Intuition

- "The identification strategy is predicated on the assumption that, conditional on number of children, the number of female children is a random variable."
- Logic of the "quasi-experiment":
    - A member of Congress has a child
    - Nature randomly assigns the child gender
    - Compare between two legislators, each with one additional child, but nature assigns the first a boy and nature assigns the second a girl
    - The difference in voting behavior between the two legislators would yield an estimate of the "daughter effect"

Regression Practice: Washington (2008) Intuition

- "The identification strategy is predicated on the assumption that, conditional on number of children, the number of female children is a random variable."
- Logic of the "quasi-experiment":
  - A member of Congress has a child
  - Nature randomly assigns the child gender
  - Compare between two legislators, each with one additional child, but nature assigns the first a boy and nature assigns the second a girl
  - The difference in voting behavior between the two legislators would yield an estimate of the "daughter effect"
- Important Variable (to address Omitted Variable Bias): controlling for the total number of children
  - Otherwise combines both the impact of parenting an additional daughter and the impact of parenting an additional child

Regression Practice: Washington (2008) Specification

$$Y_i = \alpha + \beta \times Girls_i + \gamma_{i,j} + \epsilon_i$$

- i indexes legislators
- $Y_i =$ legislator i's voting record score (or a dummy for an individual's liberal roll call vote)
- $Girls_i$ is the number of daughters that the individual legislator i parents
- γ_i is a set of fixed effects for total number of children j:
    - e.g. $\gamma_{i,1} = 1$ if legislator i has exactly 1 child; 0 otherwise.
    - $\gamma_{i,2} = 1$ if legislator i has exactly 2 children; 0 otherwise.
    - ...
- Also adds a number of controls for items that might affect voting records

Regression Practice: Washington (2008) Replication

Simple Regression, without additional controls:

```
daughter_effect_simple <- lm(nowtot ~ ngirls,
                             data= congress_voting,
                             subset = congress==105 & !is.na(totchi)) # Note: a
summary(daughter_effect_simple)

##
## Call:
## lm(formula = nowtot ~ ngirls, data = congress_voting, subset = congress ==
##     105 & !is.na(totchi))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -44.142 -34.850  -6.996  40.150  60.150
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   44.142      2.657  16.616   <2e-16 ***
## ngirls        -2.146      1.562  -1.374     0.17
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 36.48 on 428 degrees of freedom
```

Regression Practice: Washington (2008) Replication

Now, extend the simple regression, controlling for the number of total children, but without additional controls:

Regression Practice: Washington (2008) Replication

Now, extend the simple regression, controlling for the number of total children, but without additional controls:

```
daughter_effect_simple <- lm(nowtot ~ ngirls + (totchi),
                             data= congress_voting,
                             subset = congress==105 & !is.na(totchi))
summary(daughter_effect_simple)

##
## Call:
## lm(formula = nowtot ~ ngirls + (totchi), data = congress_voting,
##     subset = congress == 105 & !is.na(totchi))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.237 -32.809  -8.004  36.100  85.454
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   52.237      3.123  16.727  < 2e-16 ***
## ngirls         5.447      2.233   2.440   0.0151 *
## totchi        -7.115      1.528  -4.658 4.28e-06 ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Regression Practice: Washington (2008) Replication

Next, to match her model, we extend the model and control for "dummy variables" for each number of total children, but without additional controls:

Regression Practice: Washington (2008) Replication

Next, to match her model, we extend the model and control for "dummy variables" for each number of total children, but without additional controls:

```
daughter_effect_simple <- lm(nowtot ~ ngirls + factor(totchi),
                             data= congress_voting,
                             subset = congress==105 & !is.na(totchi)) # Note: f
summary(daughter_effect_simple)
```

```
##
## Call:
## lm(formula = nowtot ~ ngirls + factor(totchi), data = congress_voting,
##     subset = congress == 105 & !is.na(totchi))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -58.851 -33.494  -7.845  34.681  77.897
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        46.102      4.649   9.916  < 2e-16 ***
## ngirls              5.345      2.256   2.369  0.01828 *
## factor(totchi)1     7.404      7.500   0.987  0.32410
## factor(totchi)2    -6.128      6.014  -1.019  0.30884
## factor(totchi)3   -17.055      6.858  -2.487  0.01328 *
```

Regression Practice: Washington (2008) Replication

Note: `factor(.)` created the “dummy variables” (a.k.a “fixed effects”) for each number of total children. We can also define them ourselves

```
congress_voting$chid1 <- ifelse(congress_voting$totchi == 0,1,0) # No Child
congress_voting$chid2 <- ifelse(congress_voting$totchi == 1,1,0) # 1 child
congress_voting$chid3 <- ifelse(congress_voting$totchi == 2,1,0) # 2 children
congress_voting$chid4 <- ifelse(congress_voting$totchi == 3,1,0)
congress_voting$chid5 <- ifelse(congress_voting$totchi == 4,1,0)
congress_voting$chid6 <- ifelse(congress_voting$totchi == 5,1,0)
congress_voting$chid7 <- ifelse(congress_voting$totchi == 6,1,0)
congress_voting$chid8 <- ifelse(congress_voting$totchi == 7,1,0)
congress_voting$chid9 <- ifelse(congress_voting$totchi == 8,1,0)
congress_voting$chid10 <- ifelse(congress_voting$totchi == 9,1,0)
congress_voting$chid11 <- ifelse(congress_voting$totchi == 10,1,0)
```

Regression Practice: Washington (2008) Replication

Estimate the regression with the "dummy variables" we created:

```
daughter_effect_simple <- lm(nowtot ~ ngirls + chid1 + chid2 +
                               chid3 + chid4 + chid5 + chid6 +
                               chid7 + chid8 +  chid9 + chid10 +
                               chid11,
                             data= congress_voting, subset = congress==105 & !i
summary(daughter_effect_simple)

##
## Call:
## lm(formula = nowtot ~ ngirls + chid1 + chid2 + chid3 + chid4 +
##     chid5 + chid6 + chid7 + chid8 + chid9 + chid10 + chid11,
##     data = congress_voting, subset = congress == 105 & !is.na(totchi))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -58.851 -33.494  -7.845  34.681  77.897
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -21.380     36.835  -0.580   0.5619
## ngirls         5.345      2.256   2.369   0.0183 *
## chid1         67.481     37.127   1.818   0.0698 .
## chid2         74.885     36.965   2.026   0.0434 *
## chid3         61.353     36.466   1.682   0.0932 .
```

Regression Practice: Washington (2008) Replication

Main specification in Table 2 (Column (1)) includes a number of additional controls.

To add her controls, we first create some additional variables:

```
# Age Squared:
congress_voting$agesq <- congress_voting$age * congress_voting$age
# Service Length Squared:
congress_voting$srvlngsq <- congress_voting$srvlng * congress_voting$srvlng
# Religious Groups:
congress_voting$reld1 <- ifelse(congress_voting$rgroup == 1,1,0) # No Rel
congress_voting$reld3 <- ifelse(congress_voting$rgroup == 3,1,0) # Cath
congress_voting$reld4 <- ifelse(congress_voting$rgroup == 4,1,0) # Other Christ
congress_voting$reld5 <- ifelse(congress_voting$rgroup == 5,1,0) # Other
```

Regression Practice: Washington (2008) Replication

Then add the controls into our regression formula in our `lm` code:

```
daughter_effect_full <- lm(nowtot ~ ngirls +
                               factor(totchi) +
                               white + female + repub + age + agesq +
                               srvlng + srvlngsq +
                               reld1 + reld3 +reld4 + reld5 +
                               +factor(region) + demvote,
                             data= congress_voting,
                             subset = congress==105 & !is.na(totchi))
```

Regression Practice: Washington (2008) Replication

Now, print a summary of the model:

```
summary(daughter_effect_full)

##
## Call:
## lm(formula = nowtot ~ ngirls + factor(totchi) + white + female +
##     repub + age + agesq + srvlng + srvlngsq + reld1 + reld3 +
##     reld4 + reld5 + +factor(region) + demvote, data = congress_voting,
##     subset = congress == 105 & !is.na(totchi))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -52.117  -9.692  -0.437   9.207  61.997
##
## Coefficients: (1 not defined because of singularities)
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       12.702228  21.658879   0.586 0.557893
## ngirls             2.334994   1.039919   2.245 0.025292 *
## factor(totchi)1   -3.323180   3.503809  -0.948 0.343476
## factor(totchi)2   -5.678595   2.824776  -2.010 0.045074 *
## factor(totchi)3   -7.677038   3.273487  -2.345 0.019505 *
## factor(totchi)4  -11.412828   3.778003  -3.021 0.002683 **
## factor(totchi)5  -10.249032   4.728224  -2.168 0.030778 *
## factor(totchi)6  -13.141908   8.283631  -1.586 0.113420
## factor(totchi)7  -11.062139   9.598304  -1.153 0.249802
```

Regression Practice: Washington (2008) Results

Matched Column (1)!

The NOW score increases by about two points with each additional daughter parented.

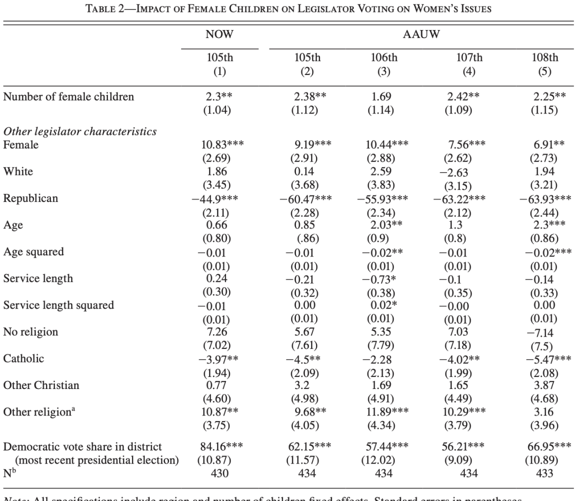


TABLE 2—IMPACT OF FEMALE CHILDREN ON LEGISLATOR VOTING ON WOMEN'S ISSUES

	NOW	AAUW			
	105th	105th	106th	107th	108th
	(1)	(2)	(3)	(4)	(5)
Number of female children	2.3**	2.38**	1.69	2.42**	2.25**
	(1.04)	(1.12)	(1.14)	(1.09)	(1.15)
Other legislator characteristics					
Female	10.83***	9.19***	10.44***	7.56***	6.91**
	(2.69)	(2.91)	(2.88)	(2.62)	(2.73)
White	1.86	0.14	2.59	−2.63	1.94
	(3.45)	(3.68)	(3.83)	(3.15)	(3.21)
Republican	−44.9***	−60.47***	−55.93***	−63.22***	−63.93***
	(2.11)	(2.28)	(2.34)	(2.12)	(2.44)
Age	0.66	0.85	2.03**	1.3	2.3***
	(0.80)	(.86)	(0.9)	(0.8)	(0.86)
Age squared	−0.01	−0.01	−0.02**	−0.01	−0.02***
	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)
Service length	0.24	−0.21	−0.73*	−0.1	−0.14
	(0.30)	(0.32)	(0.38)	(0.35)	(0.33)
Service length squared	−0.01	0.00	0.02*	−0.00	0.00
	(0.01)	(0.01)	(0.01)	(0.01)	(0.01)
No religion	7.26	5.67	5.35	7.03	−7.14
	(7.02)	(7.61)	(7.79)	(7.18)	(7.5)
Catholic	−3.97**	−4.5**	−2.28	−4.02**	−5.47***
	(1.94)	(2.09)	(2.13)	(1.99)	(2.08)
Other Christian	0.77	3.2	1.69	1.65	3.87
	(4.60)	(4.98)	(4.91)	(4.49)	(4.68)
Other religion[a]	10.87**	9.68**	11.89***	10.29***	3.16
	(3.75)	(4.05)	(4.34)	(3.79)	(3.96)
Democratic vote share in district	84.16***	62.15***	57.44***	56.21***	66.95***
(most recent presidential election)	(10.87)	(11.57)	(12.02)	(9.09)	(10.89)
N[b]	430	434	434	434	433

Note: All specifications include region and number of children fixed effects. Standard errors in parentheses.

Commenting Code

Adding in comments to code:

When writing a script in R, it is important to add comments to explain what you are doing.

This is helpful not just to you - in case you come back to a script - but also for sharing it with others.

To add comments in R, use:

```r
# This is a comment. The following code does...
```

Make sure to comment code for your scripts

Basic Data Visualization in R

Basic Data Visualization in R

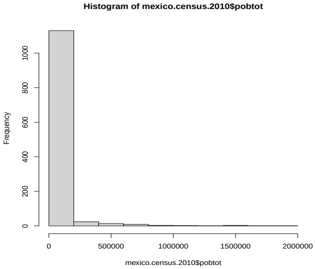
Arguably one of the biggest selling points for R is its graphing capabilities.

These capabilities have largely been advanced by the `ggplot2` package, and we will cover this in the future.

However, R does come with some basic graphing functions with its base installation, so I will cover that here

Histogram

```r
hist(mexico.census.2010$pobtot)
```



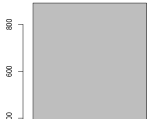
Bar Graph

The basic bar plot function is a little strange, in that you need to pass it the frequencies of the categories.

This is the first time we will see some neat properties of R, so let's take a second.

We can create frequencies using the table command:

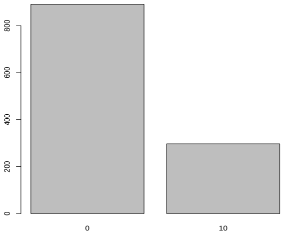
```
freq<- table(mexico.census.2010$largepop)
table(mexico.census.2010$largepop)

##
##   0  10
## 891 297

barplot(freq)
```



Bar Graph

But, you can put this all in one line of code:

```
barplot(table(mexico.census.2010$largepop))
```

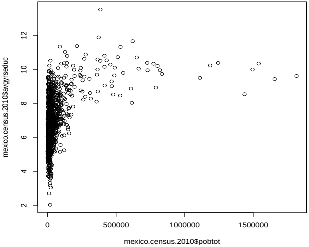


R is nice because we can call multiple functions on a single line. They are evaluated inside-out.

Scatterplot

List variables with x axis first:

```
plot(mexico.census.2010$pobtot,mexico.census.2010$avgyrseduc)
```

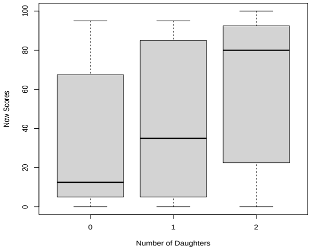


Boxplots

Using Washington (2008), we can produce a figure similar to her figure 1 using base graphics.

(To fully replicate it, we will need other libraries: much more to learn!)

```
## Keep only congress members with 2 total children:
subset_2children <- subset(congress_voting, totchi == 2)

## Box plot: NOW Scores vs. Number of Daughts,
  # Conditional on having 2 total children:
boxplot(subset_2children$nowtot~ subset_2children$ngirls,
        xlab="Number of Daughters",ylab="Now Scores") # Labels
```



Next Class

Next class

Next class:

- Importance of Institutions for Development, Part I
- R: More on Regressions in R