First R Experience

Paul E. Johnson[1] [2]

[1]Department of Political Science

[2]Center for Research Methods and Data Analysis, University of Kansas

2018

KU

Outline

KU

Outline

Assume you Installed R

- R(R Core Team, 2017) is freely downloadable from any CRAN MIRROR, such as http://rweb.crmda.ku.edu/cran
- CRMDA offers tips about installation and system maintenancehttp://crmda.ku.edu/setup
  - Includes Video "voiceover" to demonstrate installation on Win/Mac
- MAC special alert: see our guide about addon support libraries
  - XQuartz: http://xquartz.macosforge.org
  - And, if you can get it, also Xcode (App Store).
- R Core offers OS-specific FAQ
  - Win FAQ http://rweb.crmda.ku.edu/cran
  - Mac Faq: https://cran.r-project.org/bin/macosx/RMacOSX-FAQ.html
- Don't worry about "development environments" RStudio, Emacs, Notepad++ at the beginning.

KU

Launch R for some Interactive Fun

- Windows: Find the R icon and double click on it
- Mac: Use that Finder thing to launch R
- Linux: get a terminal, type R

Either Way, you should be in a "Console"

- "$>$" is called the "prompt"
- Type there to directly interact with the R system

Type this

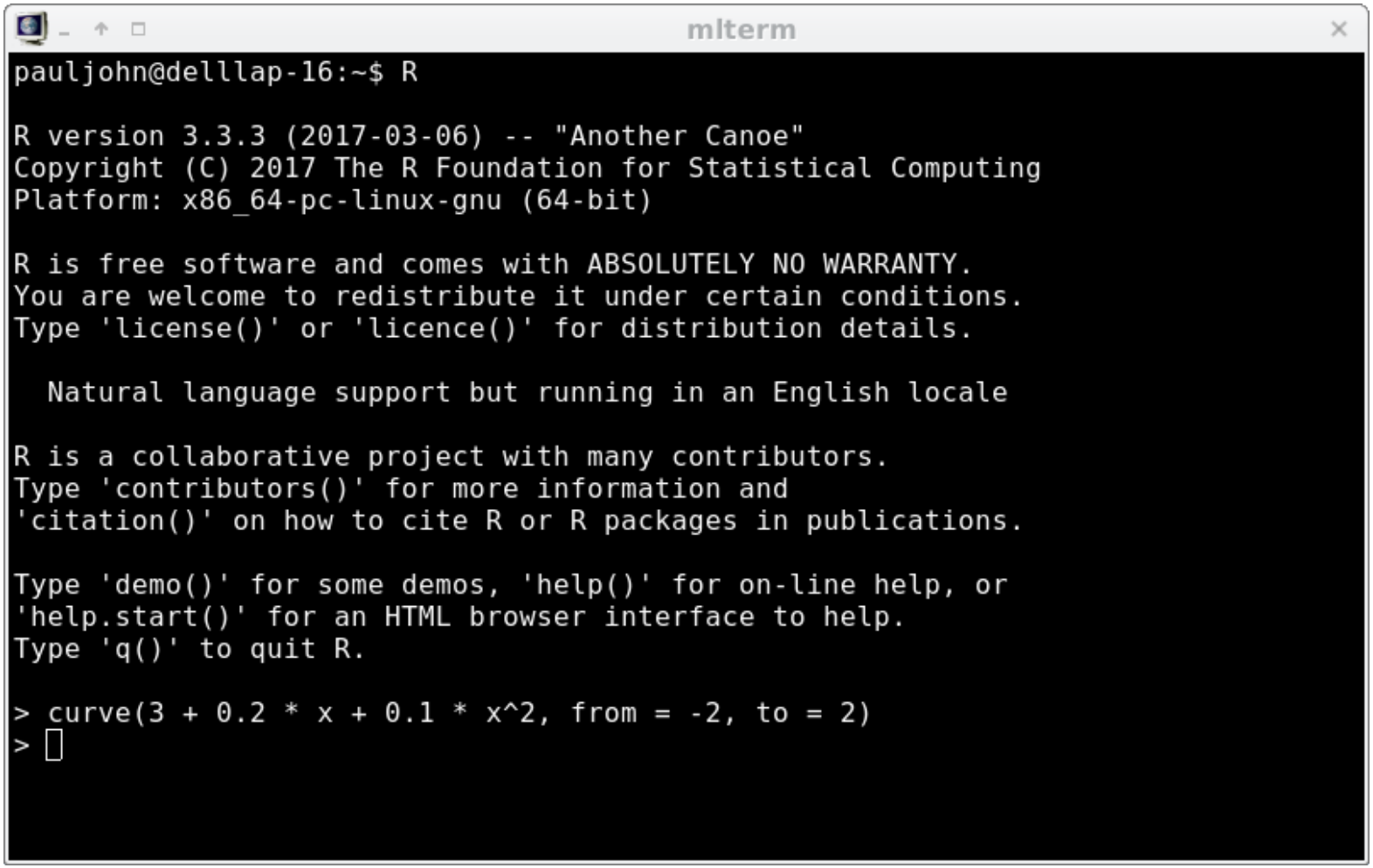
```
curve(3 + 0.2 * x + 0.1 * x^2, from = -2,
 to = 2)
```

curve is the name of a function.

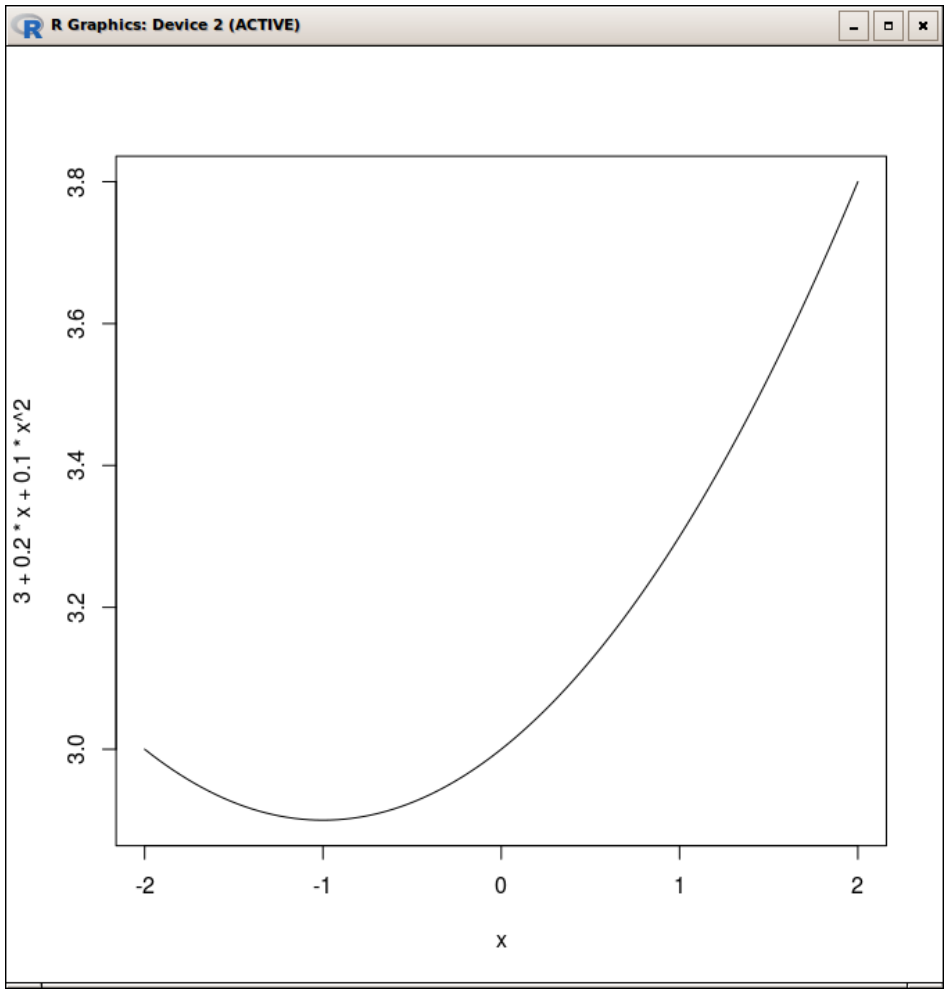
arguments are the comma-separated bits inside the parentheses

Linux R in a Terminal

On my Linux system, the "R console":

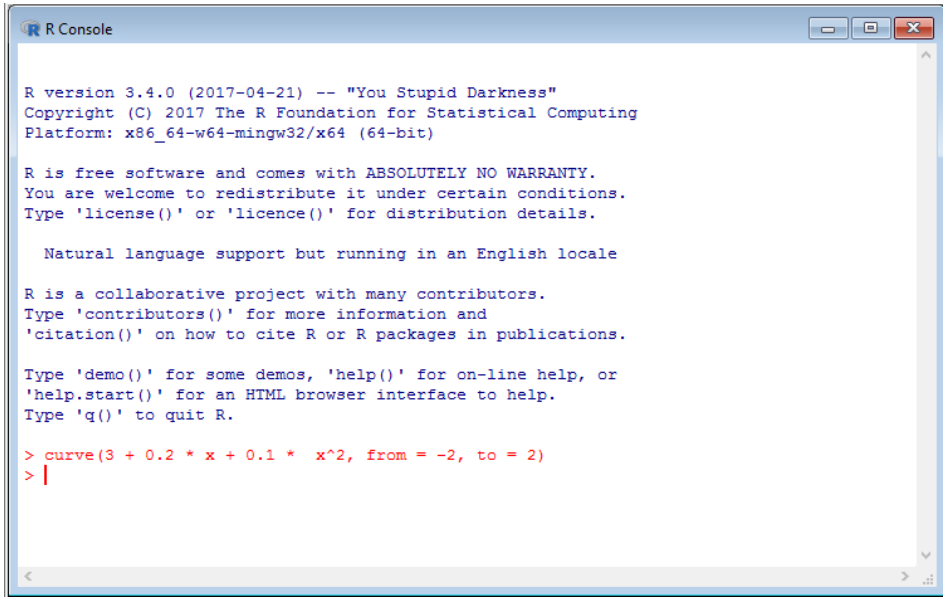


Graphic Device Window Pops up



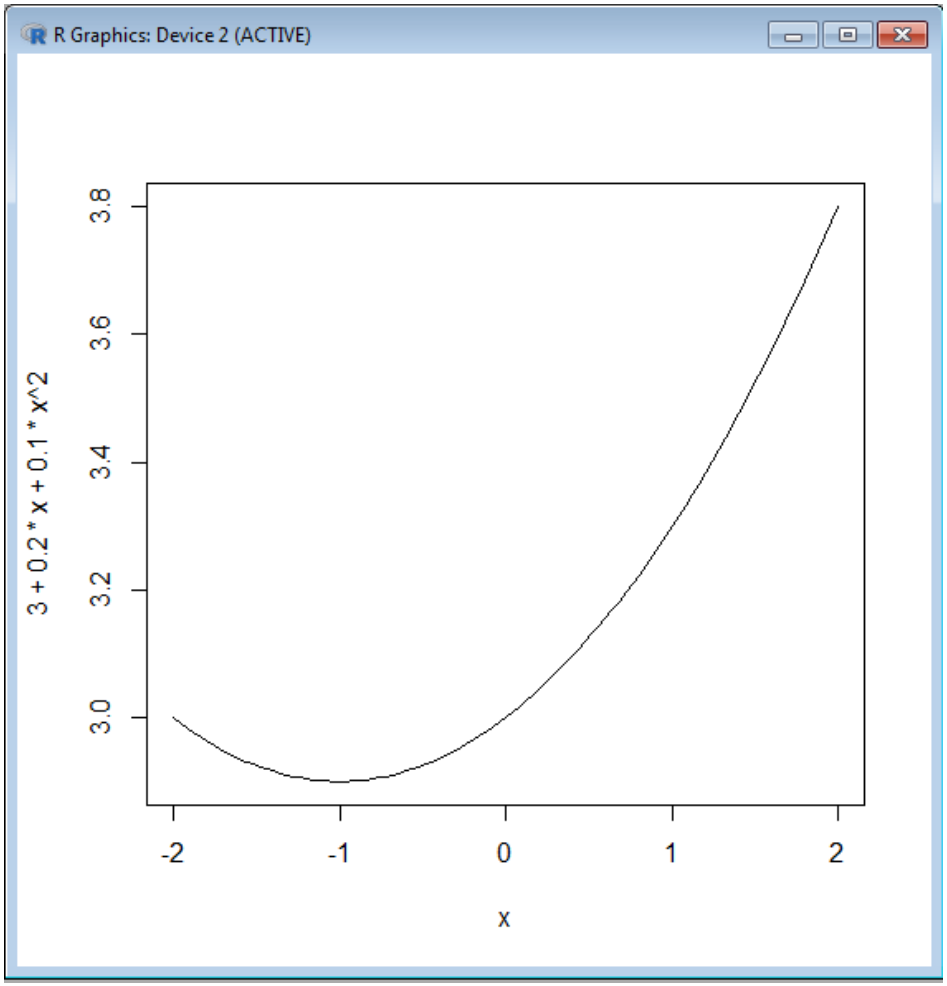
- Can have several "output devices" open at once.
- device: "just a place to draw".
- We will show you how to create "file devices" (PDF, png) as well.

In Windows, more GUI



- Similar to running R inside a "DOS Box"
- (run "cmd", then type "Rterm" and hit enter)

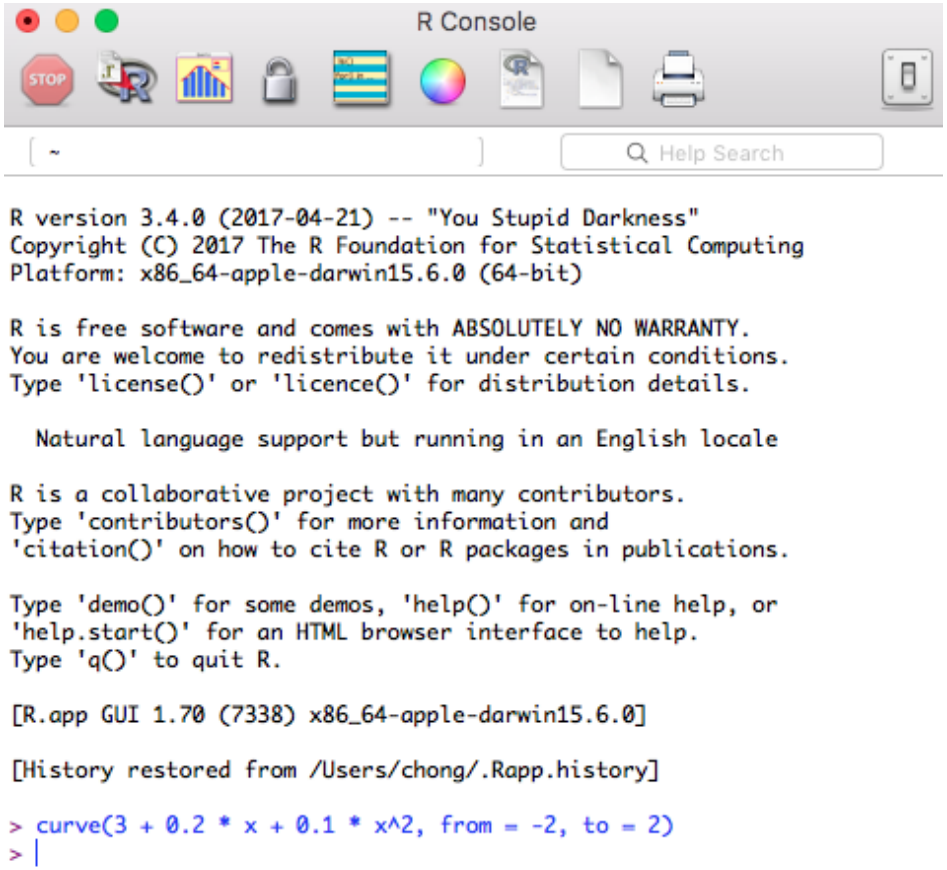
Graph pops up in a separate sub-window



- The R Windows install offers a choice of "one big window" or "many separate windows"

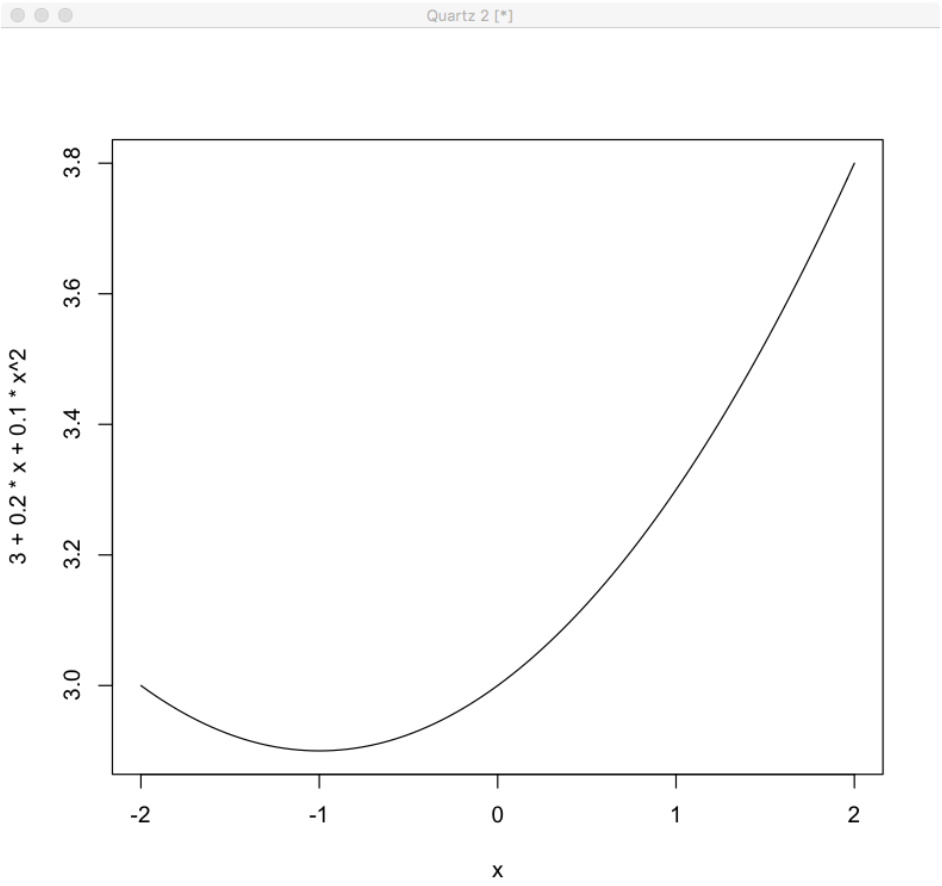
On A Mac, the Terminal is a Little Fancier

This is "**R.app**"



- This is "R.app"
- which has a Terminal Window wrapped inside an editor window
- ↙Note the handy "tooltip" down below

Mac Graphic!



- Quartz is one of the "graphic device drivers" available
- Mac users need to install XQuartz because many R packages assume the existence of the Quartz device.

Terminals have minimal Creature Comfort

- I use the terminal for quick chores, like calculating "`sqrt(534)`"
- "History" shortcut key
  - On Windows: "up arrow" cycles through previous commands

When you are ready to quit

- Run the function

```
q(save = "no")
```

- If you forget ' save = "no" ', and you just run ' q() ', R asks you

```
Save workspace image? [y/n/c]
```

- Almost never do I say "y". Here is why.
  - If you say "yes", R will save a snapshot of the current session in a file ".RData".
  - R will try to re-load that session next time you start R
- Why I always explicitly close down R sessions: If I kill the window without 'q()' then R will generally save a workspace.

Outline

Don't waste too much effort in the Console!

- Nothing you type there is "remembered" for future session
- Instead build a "script"! A file of R commands
- The Terminal is for quick testing of "what works," which you then put into a script file.

Your first Data Analysis!

- How old are Jane's children?

```
x <- c(3, 5, 7, 9, 11)
x
```

```
[1]  3  5  7  9 11
```

- x is a "vector" with 5 elements

```
mean(x)
```

```
[1] 7
```

```
var(x)
```

```
[1] 10
```

Note Well

1. Assignment symbol is "<- "

```
y <- 2 * x
```

Is the assignment symbol, we pronounce "y <- 2 * x" as "y is assigned as 2 times x".

2. All function use requires parentheses. Even simple things no arguments require parens. ls() displays an object list for the current session:

```
ls()
```

```
[1] "opts.orig" "par.orig"  "pjmar"     "tdir"      "x"
```

3. print(x) is the same as x, because R assumes that, if you type the name of something, then your must want to print it.

```
print(x)
```

```
[1]  3  5  7  9 11
```

Note Well ...

```
x
```

```
[1]  3  5  7  9 11
```

The assignment symbol <- is preferred to =

1. Historically, "=" caused an error
2. Around 2002, the R command interpreter was redesigned to treat "=" as "<-" most of the time
3. Some books are published which use the "=" sign, forgetting altogether about "<-"
4. R FAQ says "<-" still preferred because "=" leads to some hard-to-fix bugs.

Your first statistics in the R Terminal

- How much do those children weigh?

```
y <- c(21, 41, 44, 115, 105)
mean(y)
```

```
[1] 65.2
```

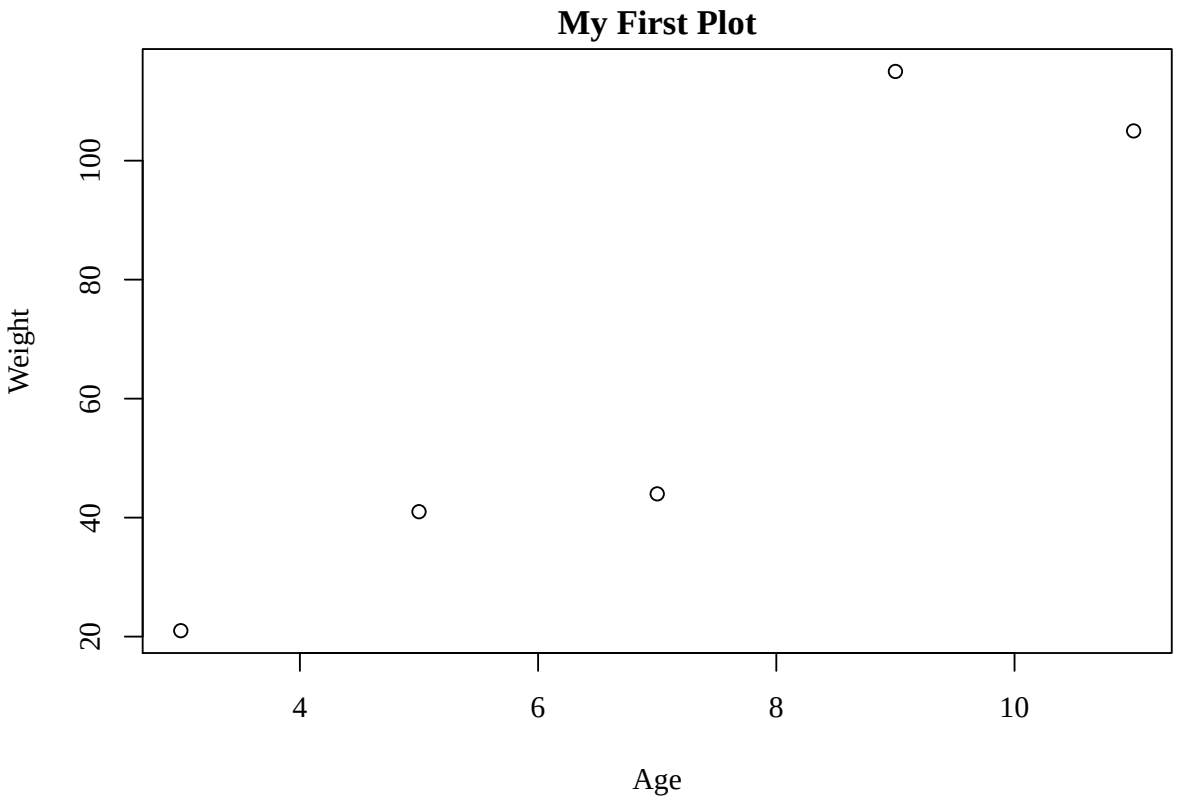
```
var(y)
```

```
[1] 1763.2
```

Your first plot, launched from the R Terminal

- This should launch a scatterplot in a separate display window

```
plot(y ~ x, xlab = "Age", ylab = "Weight", main =
    "My First Plot")
```



That's an unrealistic example

- We will import from text, or Excel, or SPSS, or ...
- But I often do create little "fake" data examples for testing things. Very often, I simulate draws from a random variable, such as $N(\mu, \sigma^2)$, with μ ("mu") and σ ("sigma") as arguments.

```
rnorm(n = 5, mean = 100, sd = 10)
```

```
[1] 105.85529 107.09466  98.90697  95.46503 106.05887
```

Draws 5 "pseudo random" values from $N(100, 10^2)$.

Outline

Manufacture example data

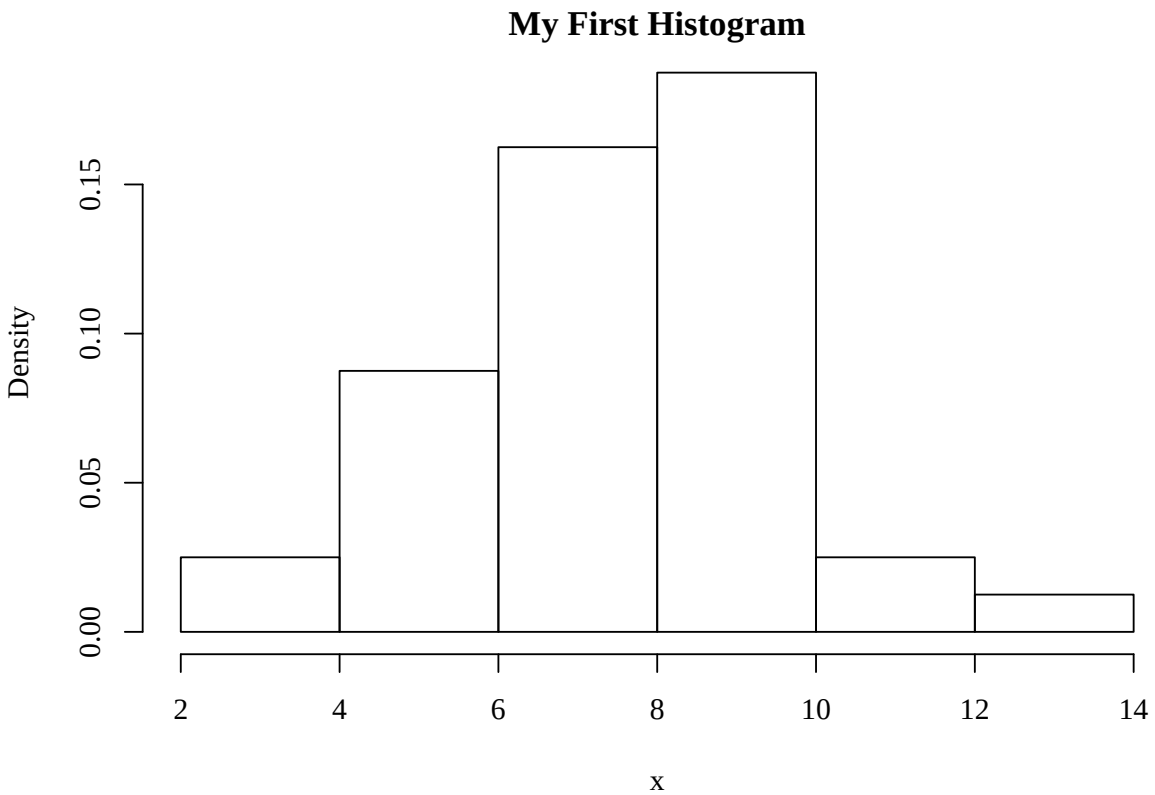
- Lets use some random normal draws

```
set.seed(66565) #put in any integer you like
x <- rnorm(n = 40, mean = 7, sd = 2)
y <- rnorm(n = 40, mean = 35, sd = 10)
```

- Two data vectors, x and y, with 40 observations in each one
- Example demonstrates "*named* arguments," `n`, `mean`, and `sd`.

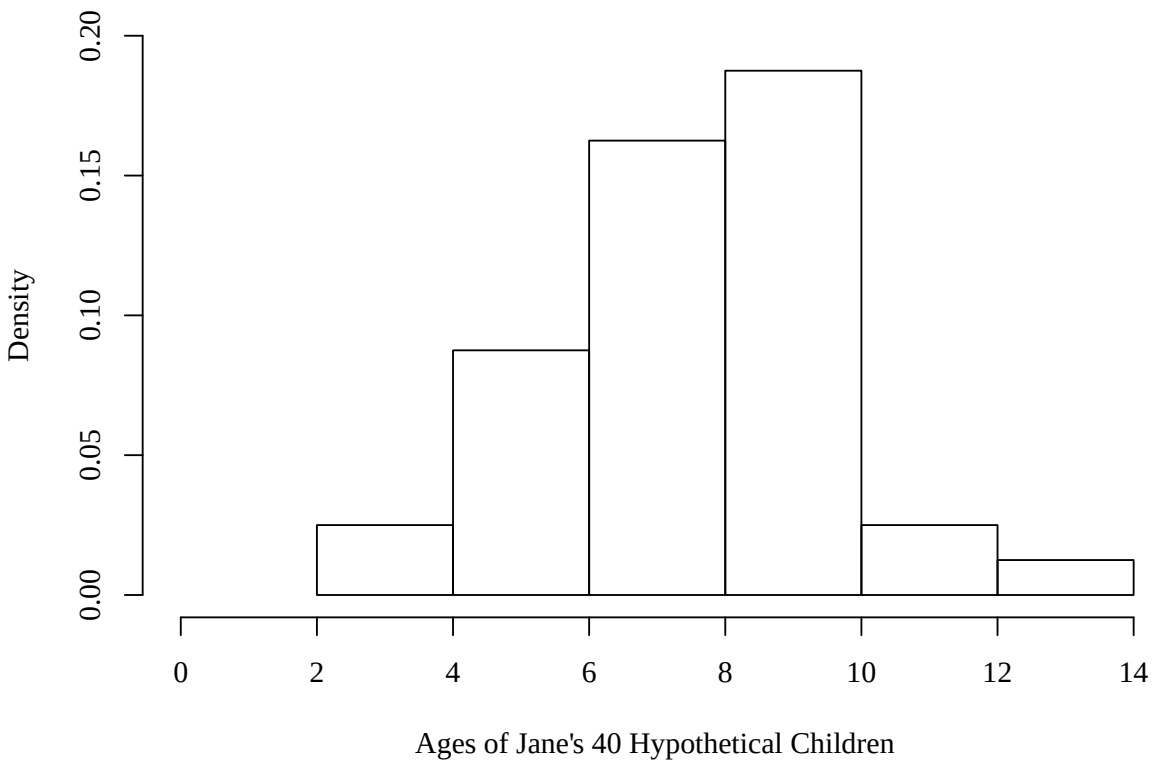
A histogram is a basic type of R Graphic

```
hist(x, prob = TRUE, main = "My First Histogram")
```



Better labels

```
hist(x, prob = TRUE, ylim = c(0, 0.20), xlim =
    c(0, 14), main = "", xlab = "Ages of Jane's
    40 Hypothetical Children")
```



Want more detail on histograms?

- Read the R documentation

```
?hist
```

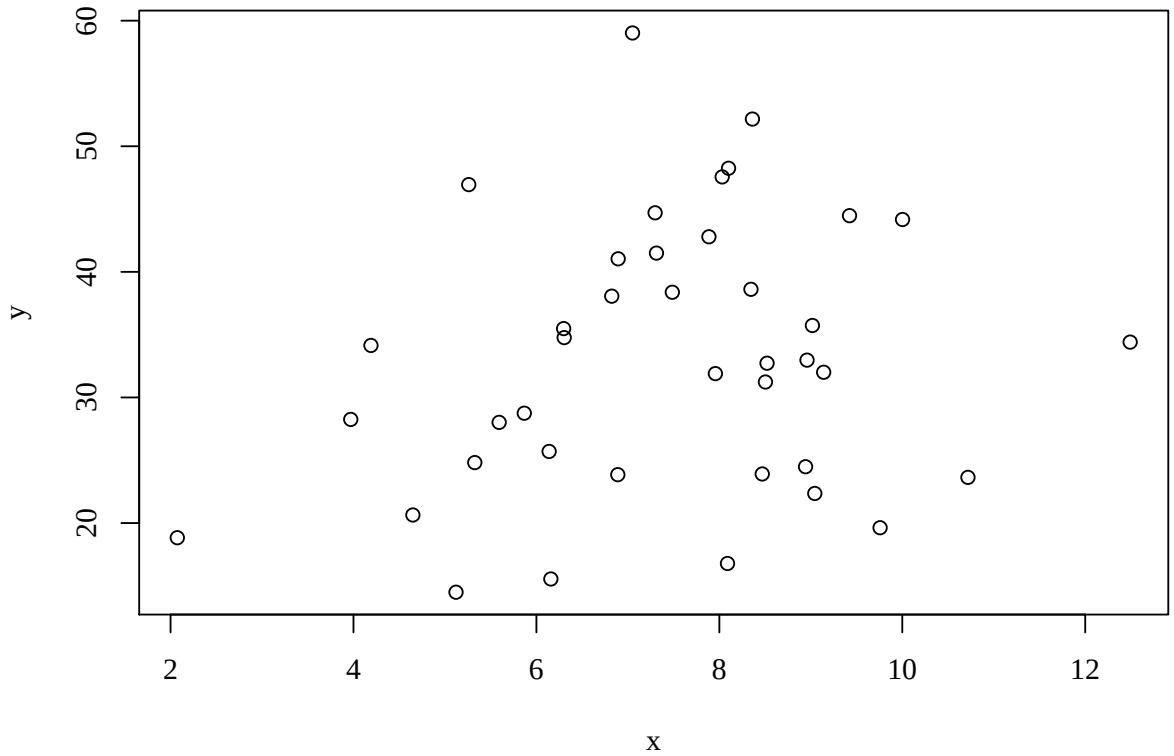
- Run their examples

```
example(hist)
```

Lets check the scatterplot

```
plot(y ~ x)
```

- "~" is "formula notation"
- Because x and y are numeric variables, R guesses we want scatterplot



Re-group Variables into Data Frame

- I left `x` and `y` floating loose, all by themselves in my workspace.
- Usually, I'd gather them together into a data frame, by doing

```
mydf <- data.frame(x = x, y = y)
```

- R functions will generally work better with data in a data.frame structure, rather than stray vectors floating about.
- After putting copies in the data frame mydf, "clean up" by removing those stray vectors

```
rm(x, y)
```

Re-group Variables into Data Frame ...

- After that, verify that x, y, are no longer objects

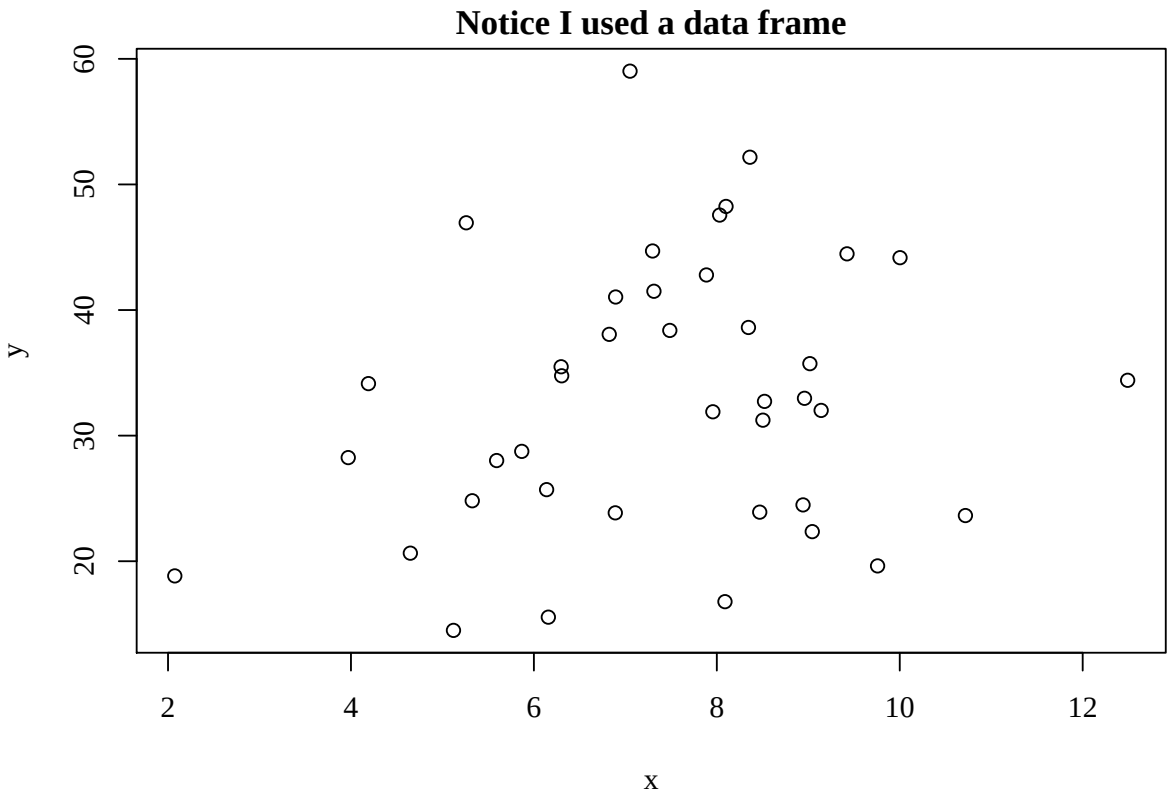
```
ls()
```

```
[1] "mydf"      "opts.orig" "par.orig"  "pjmar"     "tdir"
```

- The data.frame `mydf` exists, we adjust the code in `plot` to tell it to look into mydf for the variables "`data = mydf`":

```
plot(y ~ x, data = mydf, main = "Notice I used a
    data frame")
```

Re-group Variables into Data Frame ...



Outline

My Previous Example used Named arguments

```
set.seed(1234)
x <- rnorm(n = 40, mean = 7, sd = 2)
```

- **NEWS FLASH**: Names are not required.

```
set.seed(1234)
x <- rnorm(40, 7, 2)
```

As long as the first 3 legal arguments are the ones you want, this is a safe gamble.

- Can jumble the order of *named* arguments (must use names!)

```
set.seed(1234)
x <- rnorm(sd = 2, mean = 7, n = 40)
```

- Also possible: Partial matching of partially spelled names

```
set.seed(1234)
x <- rnorm(40, m = 7, s = 2)
```

KU

Outline

When you ask for help

- People who want help forget to tell us
  1. What *exactly* they did, or
  2. What *exactly* went wrong
- We waste time writing back asking
  1. What did you do?
  2. What kind of computer do you have?
  3. What were the error messages, exactly?
  4. Show us your code! We require a "minimal reproducible example" (MRE) of the problem.

KU

Give sessionInfo()

If you are writing to ask anybody for help about R, they need these bits of information.

1. ALWAYS provide the output of sessionInfo().

```
sessionInfo()
```

```
R version 3.6.0 (2019-04-26)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 19.04

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
     LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8
     LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8
    LC_IDENTIFICATION=C
```

Give sessionInfo() ...

```
attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

loaded via a namespace (and not attached):
[1] compiler_3.6.0 tools_3.6.0
```

sessionInfo() from a Mac

```
> sessionInfo()
R version 3.3.0 (2016-05-03)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.11.4 (El Capitan)

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base
>
```

- Observe the uniformity across operating systems (for which the R team has fought and struggled for a decade).

Outline

Editors and IDEs

- GOOD PROGRAM EDITOR.
- R's editor for Windows is not adequate
- Try
  - Notepad++
  - Emacs
  - RStudio
- Mac's R.app is OK, but possible now to use
  - Emacs
  - RStudio
- Automatic Indentation
- Highlight Matching Parentheses
- Color-coded text, program "tips"

KU

Outline

PDF, R, Rnw

- Look in the summeR-1-1 folder
- I'm preparing this file with LaTeX using
  - " Beamer " class for slide shows and
  - " Sweave " to put together R code and R code output into the PDF
- By-product from manufacturing PDF
  - " summeR-1-1-hello.R " has the R code "chunks" that are used in producing this PDF
  - You can open that R file and run the commands, assuming you have correctly specified the working directory (to be explained later)

References

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Session

```
sessionInfo()
```

```
R version 3.6.0 (2019-04-26)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 19.04

Matrix products: default
BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
     LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8
     LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8
    LC_IDENTIFICATION=C

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

loaded via a namespace (and not attached):
[1] compiler_3.6.0 tools_3.6.0
```