Using R for Introductory Statistics

John Verzani

CUNY/the College of Staten Island

January 6, 2009

http://www.math.csi.cuny.edu/verzani/R/AMS-MAA-Jan-09.pdf

Statistics and Computers

The field of statistics has been revolutionized by computers

- Access to large data sets
- Simulation studies
- Robust statistics, ...
- Computational statistics: iterative algorithms, bootstrap, MCMC, ...

Pedagogy could catch up, but for now *introductory* statistics:

- Focuses on summary statistics – now easy to compute
- Focuses on normal approximations to produce inference.
- Focuses on linear models (simple regression, ANOVA)
- Some focus on simulations

Some technology solutions for teaching statistics

As of now, no consolidation in statistics software

- Calculators – easy, reliable, students like; data sets, doesn't grow
- Excel – ubiquitious, familiar, great for data manipulation; not always right!, programming is tedious, ...
- GUI driven for students: Fathom, JMP, DataDesk; easy to use, don't grow so well
- GUI driven, commandline: Stata, SPSS, SAS, MINITAB; widely used, programming tedious.
- Command line: R (S-Plus); widely used, geared toward extending language.

Why use **R** in an Introductory class?

www.r-project.org

- Open source, multi-platform, statistical computing environment:
- Used by many academics, businesses worldwide
- A programming language (similar to S-Plus) geared toward statistical usage: pre-programmed functions for common things.
- Command line interface (CLI) encourages computational literacy (some GUIs exist)

Using **R** with introductory statistics

Things to consider

- Have clear learning goals for use: statistical understanding through examples to forced computational literacy – many target populations for *introductory statistics*
- GUIs make R easy to use to get at statistical questions (`http://www.amstat.org/publications/jse/v16n1/verzani.html`) but currently lack the polish of professional packages. (`Rcmdr`, `pmg`, `RKWard`)
- CLI is harder to teach, but R does not have a difficult syntax to learn. Introductory statistics is fairly finite.
- Never enough time, hard for all students to learn independently
- Success depends on students – students learn at different rates

Some Examples, load data

Load a data set

```
> x <- c(1.2, 2.1, 3.3)   # type in -- tedious for students
> ## built in data sets (from a package in this case)
> library(MASS)
> data(Cars93)
> ## output from excel as csv
> x <- read.csv("test.csv", header=FALSE)
> ## tab separated
> x <- read.table("test.txt", header=TRUE)
> ## or download from a website
> x <- source("http://wiener.math.csi.cuny.edu/st/R/Diet.R")
```

Numeric summaries

Summary statistics

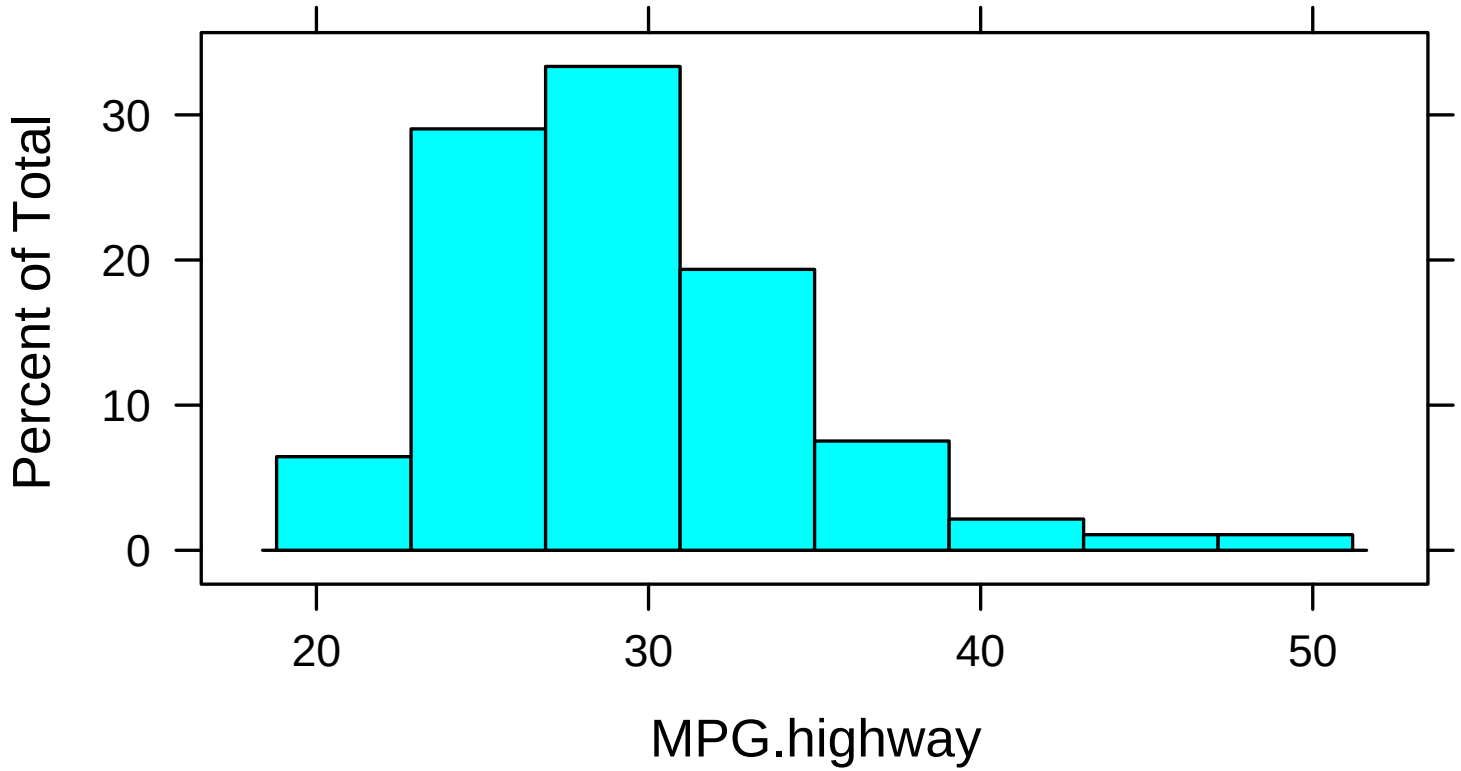
```
> mean(Cars93$MPG.highway)     # need variable name
[1] 29.08602
```

Tables

```
> xtabs( ~ Origin + Type, data=Cars93)
          Type
Origin     Compact Large Midsize Small Sporty Van
  USA            7    11      10     7      8   5
  non-USA        9     0      12    14      6   4
```

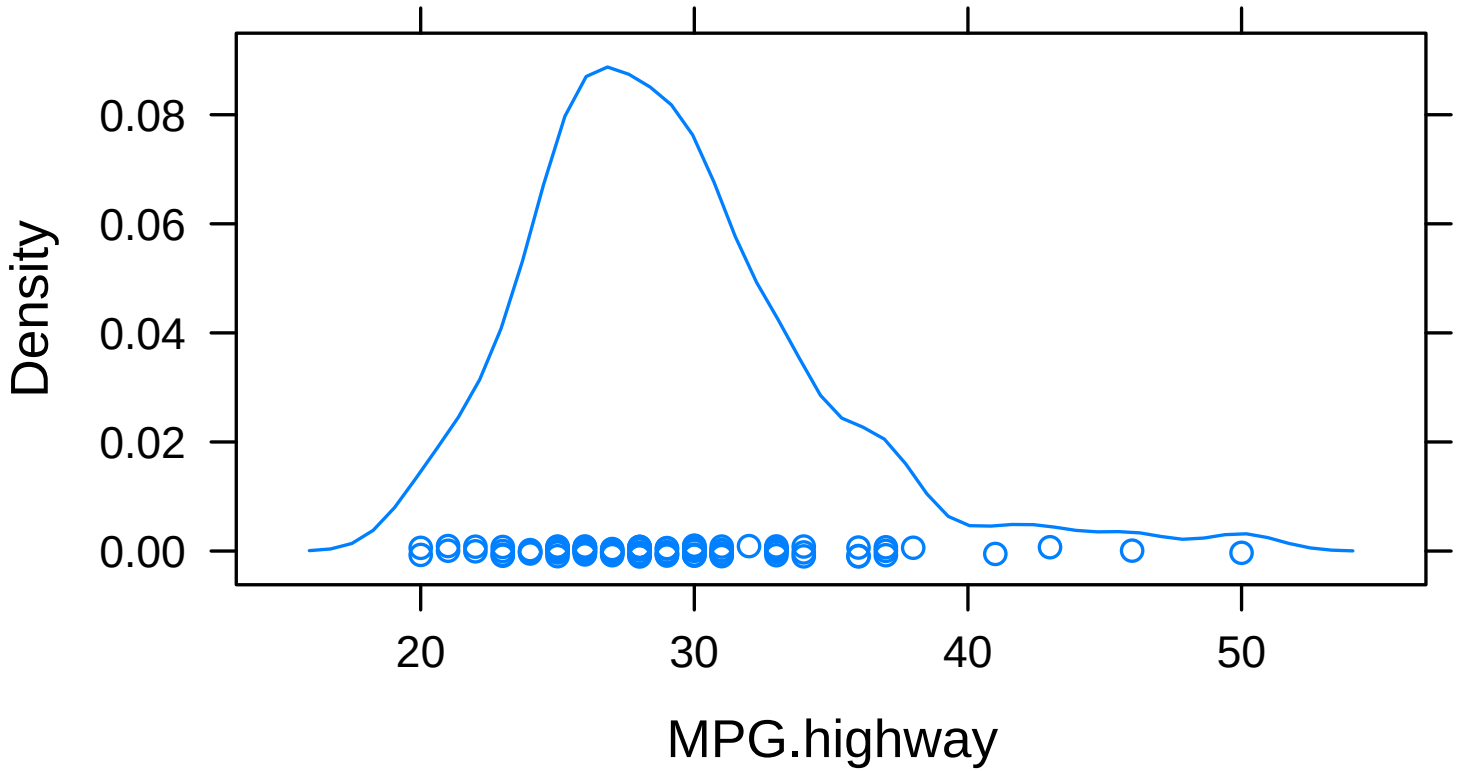
EDA: Histograms

```
> library(lattice)
> histogram(~ MPG.highway, data = Cars93)
```



Density plots

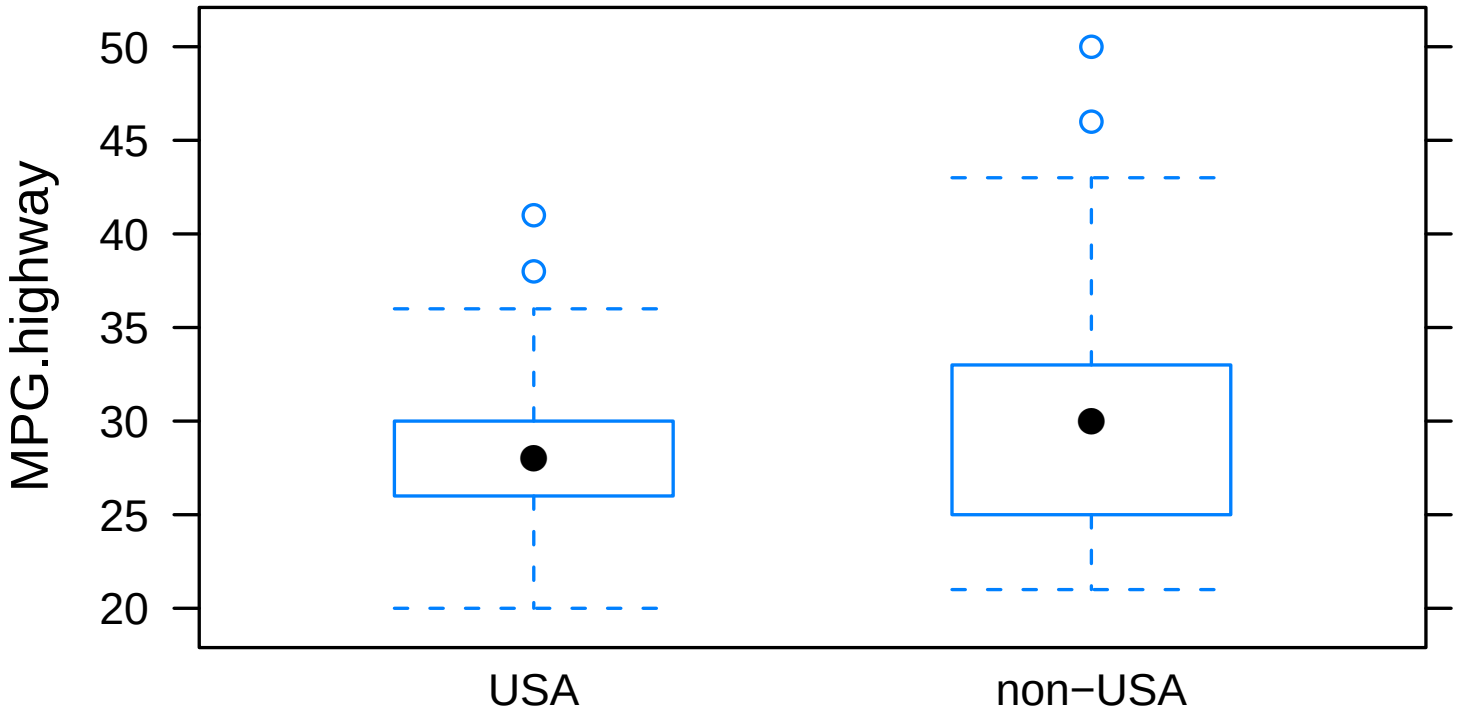
```
> densityplot(~ MPG.highway, data = Cars93)
```



Multivariate

Multivariate graphics

```
> bwplot(MPG.highway ~ Origin , data = Cars93)
```



T-tests. (Ignore lack of random sampling!)

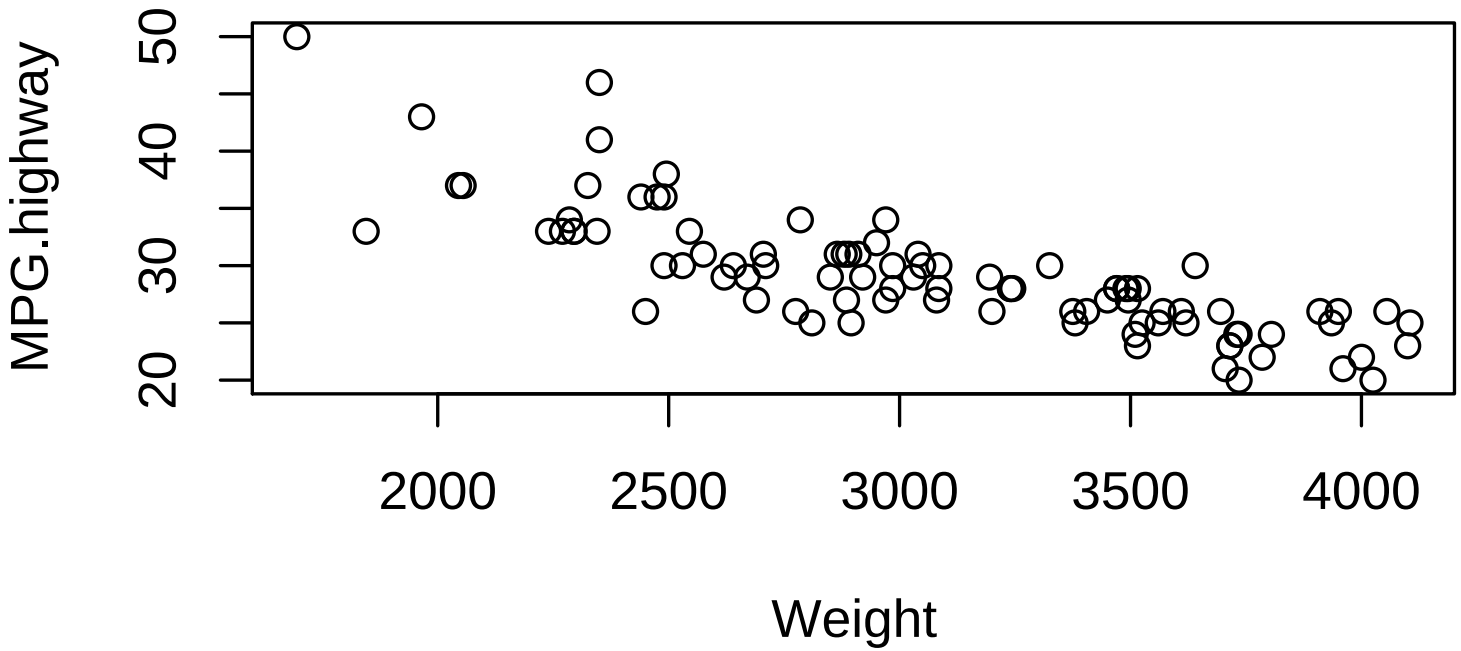
```
> t.test(MPG.highway ~ Origin, data = Cars93)

Welch Two Sample t-test

data:  MPG.highway by Origin
t = -1.7545, df = 75.802, p-value = 0.08339
alternative hypothesis: true difference in means is not equal
95 percent confidence interval:
 -4.1489029  0.2627918
sample estimates:
    mean in group USA mean in group non-USA
             28.14583                 30.08889
```

Permutation tests

```
> library(coin) ## external package -- *lots* of them
> wilcox_test(MPG.highway ~ Origin, data = Cars93)

Asymptotic Wilcoxon Mann-Whitney Rank Sum Test

data:  MPG.highway by Origin (USA, non-USA)
Z = -1.3109, p-value = 0.1899
alternative hypothesis: true mu is not equal to 0
```

Scatterplots

```
> plot(MPG.highway ~ Weight, data=Cars93)
```

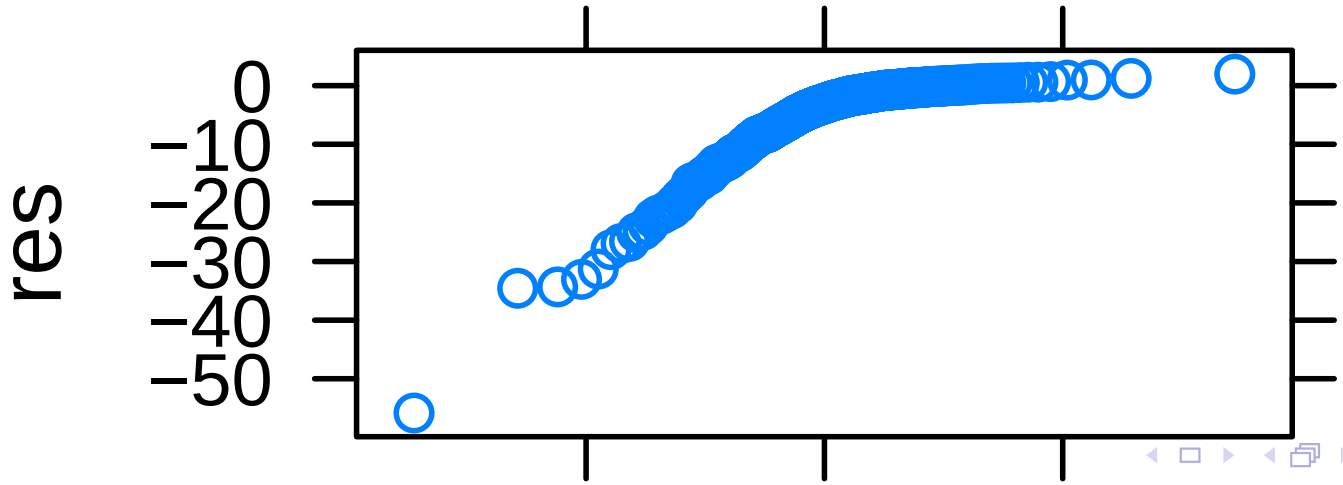


Regression

```
> res <- lm(MPG.highway ~ Weight, data = Cars93)
> summary(res)

Call:
lm(formula = MPG.highway ~ Weight, data = Cars93)

Residuals:
     Min       1Q   Median       3Q      Max
-7.65007 -1.83591 -0.07741  1.82353 11.61722

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 51.6013654  1.7355498   29.73   <2e-16 ***
Weight      -0.0073271  0.0005548  -13.21   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Simulations: for loops are one approach

```
> res <- c(); n <- 5  ## initialize
> for(i in 1:1000) {
+   x <- rexp(n) - 1
+   res[i] <- (mean(x) - 1)/(sd(x)/sqrt(n))
+ }
> qqmath(~ res, distribution = function(p) qt(p, df = n-1))
```



Materials

Students need materials to follow, often these are written by the instructor

- Examples should be driven by material – not computer use. Examples should tell a story.
- Students learn at different rates – materials should keep them all busy
- Use large, real data sets – or do problems that can't otherwise be done
- Do share – on the internet – what you create
- Avoid complication at the expense of simplicity (at times):
  - Focus on easy before hard: for loops before vectorized approaches (in **R** apply functions, sage list comprehensions)
  - Computer languages are **hard** to learn, but that part is easy to forget.
  - Functional programming easier to get across than OOP
  - Classes are needed by the developers more so than the users

Be aware that...

- CLI + syntax has no common metaphor (email, text message, ...);
- Source files, editors have no common metaphor
- Colleagues learn slower than students
- Adoption within dept. hard (standarization can be hard)
- It can be hard for students to conceptually relate computational result with question
- Many students aren't motivated by cost/availability of software