Introduction to Stata:

Computation & Regression Analysis

Kory Kantenga

University of Pennsylvania

Topics in Regression Analysis

1. Linear Regression Models
    - Least Squares Estimation
    - Inference
2. Panel Data Models
    - First Difference
    - Fixed Effects
3. Simultaneous-Equations Models
    - Instrumental Variables
4. Discrete Choice Models
    - Maximum Likelihood Estimation
5. Time Series Models

Today

1. What is Stata?
2. Computing in R
3. Graphics in R
4. More in R

Stata

Stata is a both high-level language for programming and a user interface

Stata Windows

- Do-file Editor: Commands to execute, make comments
- Data Editor: Browse and edit data
- Command Line: Type in command and hit `Return`
- Variables: Variable names and types
- Graphs: Plots (to export)
- Review: List of past commands entered
- Results

Stata Interface

Why Stata?

- Designed for statistics
- User Friendly
- Standard tools + developer tools

Getting Started

Download data from

Who am I? Where am I?

It is always a good idea to clear the environment before starting

```
rm(list=ls())
```

R needs to know where to look for things so we need to set the directory

```
getwd()
setwd("FolderPathGoesHere")
```

What are packages? Where to find them?

- Packages are bundles of commands, functions and data that you can use to do analysis

- R comes with a bunch of preloaded packages, but you may need more for the problem sets

Notable Examples:

- `ggplot2`: Famous package for making beautiful graphics
- `quantmod`: Tools for downloading financial data, plotting charts, and doing analysis
- `stargazer`: Package for creating beautiful regression tables

$\Rightarrow$ `https://cran.r-project.org`

How to install a package?

You can easily install them with *R* Studio, but you must load the package library each session so R knows you want to use it

Instructions:

1. Enter `install.packages("NameGoesHere")` into the Console.
2. Press the `Return` key
3. Add `library("NameGoesHere")` to the top of your R Script and run this command before starting each session

What is an R Script? Where to find them?

- An R Script is a text file (with the file extension `.R`) containing your R Code
- R Code is a series of comments, commands and functions that execute the tasks you require
- We can add comments using the # symbol

Where to find them?

You make your own :-)

The Magic of R Scripts

You can layer R Script together by telling R to use functions from another R Script.

Just add `Source("FileNameGoesHere.R")` at the top of your R Script and run it each session.

R Script

```
#########################################
#
# Econ 104 R Demo
#
##########################################

# Set Directory and Install packages here
setwd("/Users/korydkantenga/Dropbox/Teaching/Econometrics/

# Install
#install.packages("maps")

# Load Package
library("maps")

# Getting Help
?mean   # help file
??mean  # search
```

How to execute an R Script?

- Retype into Console and hit `Return` key
  line by line :-(
- Copy/Paste into Console and hit `Return` key
  line by line :-/
- Highlight a line and click `Run` :-)
- Click `Source` to run the entire script ;-)

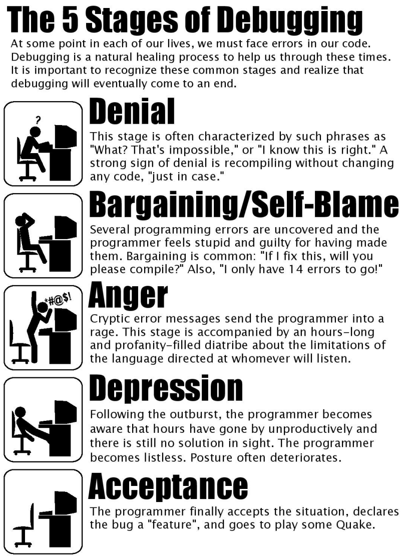
Debugging

More than likely you will run into errors. These errors are usually bugs.

Bugs are problems in YOUR CODE.
Almost all errors will be syntax errors.

Syntax errors are the coding equivalent of misspellings.

Debugging



The 5 Stages of Debugging

At some point in each of our lives, we must face errors in our code. Debugging is a natural healing process to help us through these times. It is important to recognize these common stages and realize that debugging will eventually come to an end.

Denial

This stage is often characterized by such phrases as "What? That's impossible," or "I know this is right." A strong sign of denial is recompiling without changing any code, "just in case."

Bargaining/Self-Blame

Several programming errors are uncovered and the programmer feels stupid and guilty for having made them. Bargaining is common: "If I fix this, will you please compile?" Also, "I only have 14 errors to go!"

Anger

Cryptic error messages send the programmer into a rage. This stage is accompanied by an hours–long and profanity–filled diatribe about the limitations of the language directed at whomever will listen.

Depression

Following the outburst, the programmer becomes aware that hours have gone by unproductively and there is still no solution in sight. The programmer becomes listless. Posture often deteriorates.

Acceptance

The programmer finally accepts the situation, declares the bug a "feature", and goes to play some Quake.

Getting Help

1. Use `?x` to see the help file for a command with name `x`
2. Use `??x` to search for a command or package with name `x`
3. Enter `help.start()` into the Console and hit `Return` key
4. Try a tutorial series (`http://ditraglia.com/Econ103Public/`)
5. GIFY (G-It-For-Yourself)
6. Go to TA office hours
7. E-mail a TA

Today

1. What is Stata?
2. Computing in R
3. Graphics in R
4. More in R

Simple Computations

- Use `<-` and `=` to assign values (e.g. `x<-3.141593`)
- Standard arithmetic operators (e.g. `+,-,*,/`)
- Standard operators like
  - natural logarithm: `log(x)`
  - e^x: `exp(x)`
  - a^x: `a^x`

Creating New Variables

```r
# Suppose we want to create a list of numbers
# We use c() for combine.
x <- c(3,1,4,1,5,9,3)
y <- c(2,7,1,8,2,8,2)

# To create a column vector we combine rows.
x <- rbind(3,1,4,1,5,9,3)
print(x)
```

```
##      [,1]
## [1,]    3
## [2,]    1
## [3,]    4
## [4,]    1
## [5,]    5
## [6,]    9
## [7,]    3
```

```r
# To create a row vector we combine columns
y <- cbind(2,7,1,8,2,8,2)
print(y)
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,]    2    7    1    8    2    8    2
```

Creating a Data Frame

```r
# Transpose y into a column vector
y <- t(y)
# Combine in Data
data.new <- cbind(x,y)
# Turn into Data Frame
data.new <- as.data.frame(data.new)
print(head(data.new))
```

```
##   V1 V2
## 1  3  2
## 2  1  7
## 3  4  1
## 4  1  8
## 5  5  2
## 6  9  8
```

```r
# Extract new dataset from Data Frame
data.sub <- subset(data.new, select = c("V2"))
print(head(data.sub))
```

```
##   V2
## 1  2
## 2  7
## 3  1
## 4  8
## 5  2
## 6  8
```

Importing Data

```
# Importing Data
caschool <- read.csv("caschool.csv")
print(head(caschool))
```

```
##   Observation.Number dist_cod  county                        district
## 1                  1    75119 Alameda              Sunol Glen Unified
## 2                  2    61499   Butte            Manzanita Elementary
## 3                  3    61549   Butte     Thermalito Union Elementary
## 4                  4    61457   Butte Golden Feather Union Elementary
## 5                  5    61523   Butte        Palermo Union Elementary
## 6                  6    62042  Fresno         Burrel Union Elementary
##   gr_span enrl_tot teachers calw_pct meal_pct computer testscr   comp_stu
## 1   KK-08      195    10.90   0.5102   2.0408       67  690.80 0.3435898
## 2   KK-08      240    11.15  15.4167  47.9167      101  661.20 0.4208333
## 3   KK-08     1550    82.90  55.0323  76.3226      169  643.60 0.1090323
## 4   KK-08      243    14.00  36.4754  77.0492       85  647.70 0.3497942
## 5   KK-08     1335    71.50  33.1086  78.4270      171  640.85 0.1280899
## 6   KK-08      137     6.40  12.3188  86.9565       25  605.55 0.1824818
##   expn_stu      str    avginc    el_pct read_scr math_scr
## 1 6384.911 17.88991 22.690001  0.000000    691.6    690.0
```

Modify and Summarize Variables

```
# List Variables
ls(caschool)
```

```
##  [1] "avginc"             "calw_pct"           "comp_stu"
##  [4] "computer"           "county"             "dist_cod"
##  [7] "district"           "el_pct"             "enrl_tot"
## [10] "expn_stu"           "gr_span"            "math_scr"
## [13] "meal_pct"           "Observation.Number" "read_scr"
## [16] "str"                "teachers"           "testscr"
```

```
# Take logs of Average Income and
# add to data frame
caschool$logavginc <- log(caschool$avginc)

# Summary Statistics
mean(caschool$logavginc) #mean
```

```
## [1] 2.644841
```

```
sd(caschool$logavginc) #standard deviation
```

```
## [1] 0.392373
```

```
summary(caschool$logavginc) #quartiles
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.674   2.365   2.619   2.645   2.870   4.013
```

Hypothesis Testing

```
# Hypothesis Test on Mean
# H0: mu = 2.6
# H1: mu > 2.6
xbar = mean(caschool$logavginc)
serr = sd(caschool$logavginc)
N    = length(caschool$logavginc)
# test statistic (xbar-mu)/(sigma/sqrt(N))
t = (xbar-2.6)/(serr/sqrt(N))
# compute p-value from quantile function
print(round(1-pnorm(t),2))
```

```
## [1] 0.01
```

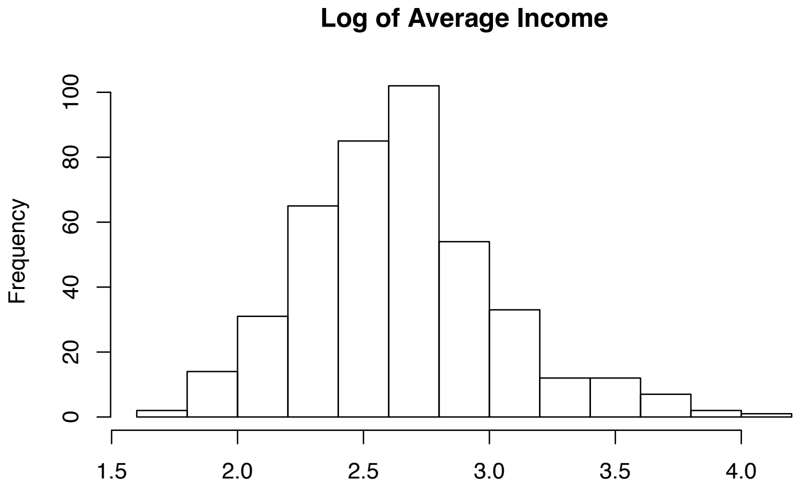
```
# Hence, we fail to reject
# the null at 0.5% signifiance level
# but may reject at the 1% level.
```

Today

1. What is Stata?
2. Computing in R
3. Graphics in R
4. More in R

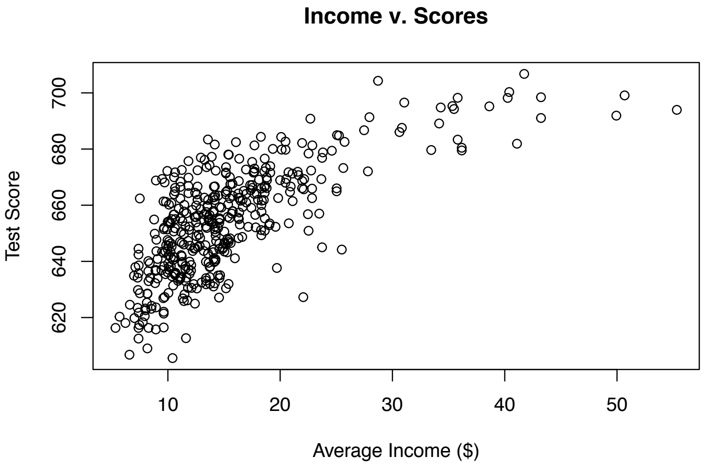
Histogram

```
# Is log average income normal?
hist(caschool$logavginc,xlab="",main="Log of Average Income")
```



Scatter Plot

```
# Is there a relationship between income and test scores?
plot(caschool$avginc,caschool$testscr, xlab="Average Income ($)",ylab="Test Score"
     , main="Income v. Scores")
```



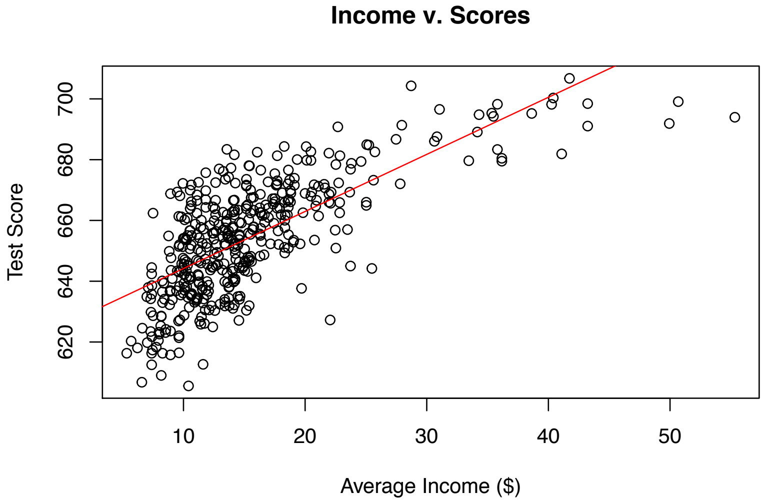
Simple Linear Regression

```r
# Does income predict test scores?
# Simple Linear Regression
lm.ols <- lm(testscr ~ avginc, data = caschool)
summary(lm.ols)
```

```
##
## Call:
## lm(formula = testscr ~ avginc, data = caschool)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.574  -8.803   0.603   9.032  32.530
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 625.3836     1.5324  408.11   <2e-16 ***
## avginc        1.8785     0.0905   20.76   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.39 on 418 degrees of freedom
## Multiple R-squared:  0.5076, Adjusted R-squared:  0.5064
## F-statistic: 430.8 on 1 and 418 DF,  p-value: < 2.2e-16
```

Fitted Line

```
plot(caschool$avginc,caschool$testscr, xlab="Average Income ($)",ylab="Test Score"
      , main="Income v. Scores")
abline(lm(testscr ~ avginc, data = caschool),col="red")
```



Today

1. What is Stata?
2. Computing in R
3. Graphics in R
4. More in R

Advanced Topics (beyond our scope)

- Functions
- Loops
- Low-level Coding
- Astral Projection and Telekinesis

For all of these and more, try some *R* tutorials and GIFY.

References

- Xie, Yihui (2014). *An Introduction to R*. UPenn, The Warren Center. Nov 21. `http://ditraglia.com/pdf/yihui1.pdf`
- The R Project for Statistical Computing. `https://www.r-project.org`
- R-Bloggers. `https://www.r-bloggers.com`