# Biostatistics workshop series: Introduction to Stata and R

Dr Cameron Hurst
cphurst@gmail.com

CEU, ACRO and DAMASAC, Khon Kaen University

June 24, 2013

# Objective

The role of today's session is to introduce software for the analysis of health and medical data. I will also mention software that SHOULD NOT be used for this purpose: We will talk about four packages.

1. **Stata:** An excellent biostatistics package whose particular strength is for 'off-the-shelve analysis'. Both a good front end (point and click) wonderfully succint succint syntax ('programming language').
2. **R:** An extremely diverse and versatile open source (free) statistical programming language. If a statistical method exists, it should be implemented in R. Problem: Very limited 'front-end'. Mostly syntax based (programming only)

# Objective

4. **SAS**: Until recently, considered the 'Rolls-Royce' of statistical packages. Like R, mainly syntax based, unlike R, very cumbersome (and out-dated) programing langague (based on PL/1 from the 1960s). Also VERY expensive (about \$5,000 USD $\approx$ 150,000 THB and about \$2,000 USD license fee every year).
5. **SPSS:** Acronym stands for Statistical Packages for Social Sciences, and that is where it should stay. It SHOULD NOT be used in biostatistics at all (Gives the illusion of being simple to use, which is true for simple analyses, but very quickly you see how poorly it performs in the area of biostatistics)

# What we will cover....

1. Introduction and case study
   - Introduction
   - Case study: The DMHT dataset
2. Stata basics and Data I/O
   - Stata: Quick view
   - Stata: Data I/O
3. R basics and Data I/O
   - R: Quick view
   - R: Data I/O
4. Data managment in Stata and R
5. Generating summary statistics Stata and R
   - Continuous variables
   - Categorical variables
6. Graphics in Stata and R

# Conventions

The conventions I will use:

**Note:.....**

Things to note will occur in a XXXXX box

**Pitfalls:.....**

Common mistakes and things to watch out for will occur in a red box

# Conventions: Continued

**R syntax:.....**

All R syntax will be in a purple box and in `courier` font.

**Stata Syntax:....**

All Stata syntax will be in a blue box and in `courier` font.

I hope this will help you find it easily when you have to refer back to these notes.

# DMHT: Study background

- Collaborative clinical study supported by the Thailand National Health Security Office (NHSO) and the Thailand Medical Research Network (MedResNet)
- Official title: *An Assessment on Quality of Care among Patients Diagnosed with Type 2 Diabetes and Hypertension Visiting Ministry of Public Health and Bangkok Metropolitan Administration Hospitals in Thailand* (Thailand DM/HT)
- In short, main research objective is to assess quality of care of (Type 2) Diabetic and Hypertensive patients in Thailand
- At present, about 150,000 patients from about 600 across Thailand, sampled from 2553-5
- Three main groups: (1) Patients with T2DM (alone), (2) Those with HT (alone), and (3) Those with both (DMHT)

# (Our) research questions

For the purpose of our excerises today, our research questions are:

1. Among diabetics, does the hypertension comorbidity represent an additional burden for achieving clinical (quality of care) goals?
2. What other (diabetic) patient characteristics (e.g. demographic/lifestyle) might influence achievement of the clinical goals?

## Data we will consider

To keep things simple:

- We will consider a random sample of 5000 patients (sampled in 2554), and only the diabetics [T2DMs and DMHTs]
- Of the hundreds of variables, we will only consider a subset:

| **Outcomes** | **Study effect** | **Other covariates** |
|---|---|---|
| a1cyn | ht | sex |
| bpyn | | age |
| ldlcyn | | religion |
| | | |
| all3yn | | duradm |
| any3yn | | smoke |
| | | bmigroup |

*Detailed description of variables next few slides...*

# Variable description: Outcomes

The outcomes in our 'subset' of the dataset are the "ABC" clincal goals often used to assess the quaility of diabetes care. These are:

(a) `a1cyn`→Hemoglobin **A**1C: yes: $< 7\%$; no: $\geq 7\%$

(b) `bpyn`→**B**lood pressure: yes: $< \frac{130}{80}$ mmHg; no: $\geq \frac{130}{80}$ mmHg;

(c) `ldlcyn`→Low density lipid-**C**holesterol: yes: LDL-C $< 100$ mg/dL; no: LDL-C$\geq$ 100 mg/dL

We will also consider the collective quality performance outcomes:

- `all3yn`: **All** three (ABC) clinical goals are met: yes; no
- `any3yn`: **Any** of the three (ABC) goals are met: yes; no

**Question**: What type of measurement scale do all of these 5 variables have?

# Study effect

We are interested in whether there is any difference in the achievement of treatment goals between those with diabetes alone (T2DM) and those diabetics who ALSO have hypertension; Does a hypertention comorbidity represent an additonal burden specifically in terms of diabetes care.

We will use the variable, `ht` to measure this: no (T2DM alone); yes (DM+HT)

## Note: Study Effects

Remember a **study effect** is the explanatory variable (X) that is of **primary interest** (in terms of our research hypothesis)

# Covariates

In observational studies (such as the DMHT study), many other explanatory (X) variables need to be considered. I will make a dinstiction between two different types of covariates here:

1. Independant risk factors: Other important **independant** explanatory variables of the outcome
2. Confounders: Variables that are associated with BOTH the outcome AND the study effect which, if ignored, can misleadingly enhance/diminish (bias) the true relationship between the outcome (Y) and the study effect (X)

## Example of a confounder: RCT

What would happen if randomiation (in an RCT) failed and we put the more sick people in the control group, and less sick people in the treatment group?

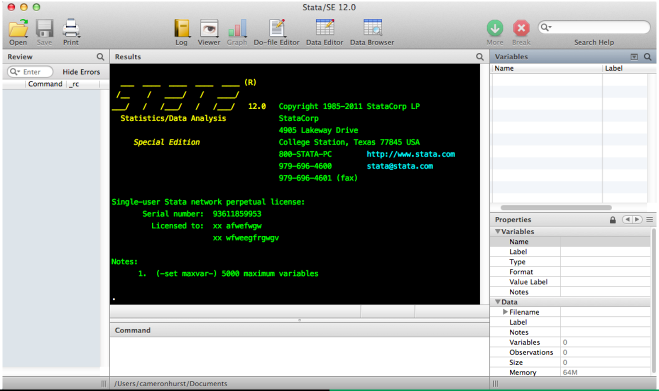
## Covariates

In our case study, I have included 6 covariates which can be split into two different groups:

- Demographic variables
    - `sex`: Patient gender (binary)
    - `age`: Age in years (continuous)
    - `religion`: Buddhist/Non-buddhist (binary)
- Lifestyle/patient history variables:
    - `duradm`: Duration of T2DM; How long (years) since patient was diagnosed with T2DM (continuous)
    - `smoke`: Smoking history: Current, Previous, Never, Unknown (Nominal/Ordinal)
    - `bmigroup`: Underweight, Normal, Overweight, Obese (Ordinal)

# Our first Stata session

To start, let's just fire up Stata and see what it looks like (Note: I use Stata for Mac version 12, looks a little different, but works exactly the same)



```
  ___  ____  ____  ____  ____ (R)
 /__    /   ____/   /   ____/
___/   /   /___/   /   /___/   12.0   Copyright 1985-2011 StataCorp LP
  Statistics/Data Analysis            StataCorp
                                      4905 Lakeway Drive
     Special Edition                  College Station, Texas 77845 USA
                                      800-STATA-PC        http://www.stata.com
                                      979-696-4600        stata@stata.com
                                      979-696-4601 (fax)

Single-user Stata network perpetual license:
       Serial number:  93611859953
         Licensed to:  xx afwefwgw
                       xx wfweegfrgwgv

Notes:
      1.  (-set maxvar-) 5000 maximum variables

.
```

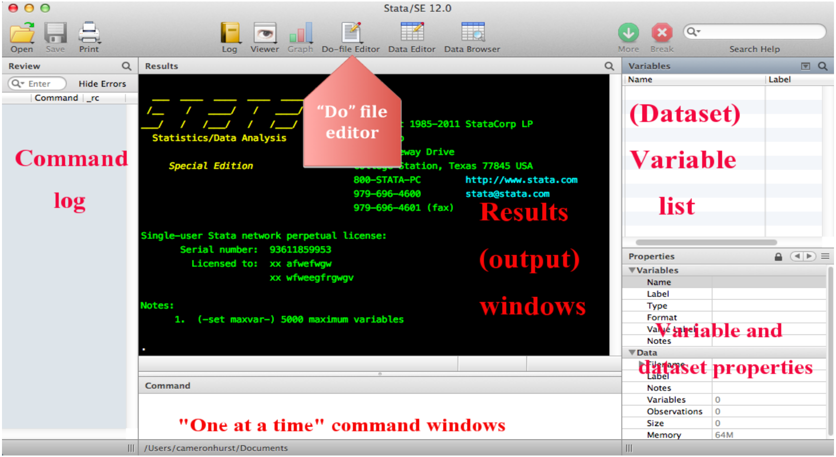
Properties

▼Variables
Name
Label
Type
Format
Value Label
Notes

▼Data
▶ Filename
Label
Notes
Variables 0
Observations 0
Size 0
Memory 64M

# Basic architecture

In **Red** is the basic purpose of each windows.



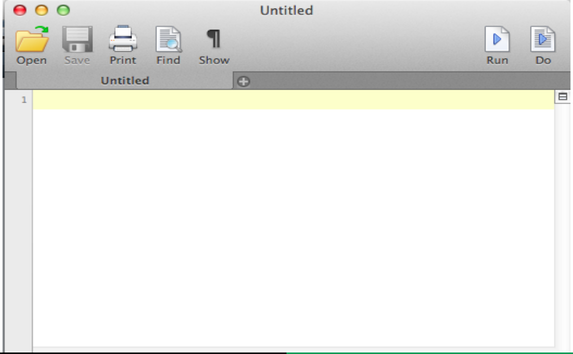
```
  Statistics/Data Analysis                   1985-2011 StataCorp LP
                                              way Drive
     Special Edition                          Station, Texas 77845 USA
                                    800-STATA-PC        http://www.stata.com
                                    979-696-4600        stata@stata.com
                                    979-696-4601 (fax)

Single-user Stata network perpetual license:
       Serial number:  93611859953
         Licensed to:  xx afwefwgw
                       xx wfweegfrgwgv

Notes:
      1.  (-set maxvar-) 5000 maximum variables

.
```

| Properties | |
|---|---|
| Variables | 0 |
| Observations | 0 |
| Size | 0 |
| Memory | 64M |

/Users/cameronhurst/Documents

# Stata "Do" files

The other important window in Stata is the **Do file editor**. This allows us to create "Do" files; Files containing our Stata syntax (Stata programs). If you click the Do file editor you will get a blank text (.do) file (more later).



# Reading data into Stata

In discussing data input to Stata I will cover two methods:

1. Pull down menus (Point-and-click); and
2. Syntax

I will also demonstrate how to read data in two different formats:

1. Stata data ('dta") files; and
2. Comma delimted ("csv") text files

### Warning: Avoiding problems in data input

There are many other types of text files, but I have always found using commas as seperators (delimiters) as the safest (for example if you use spaces you will have problems with text variables that also contain spaces). A CSV file can easily be created using excel.

# DMHT data: Reading into Stata

You have been provided with two data files:

1. `dmht5000.dta`, the Stata datafile
2. `dmht5000.csv`, the comma seperated text file

Now we will use Stata to input these files

**Hint: SAVE YOUR SYNTAX!!**

- When you use the pull down menus in Stata, syntax is generated (in results windows)
- Open a new do file and copy and paste this syntax (without the leading ".") into your new do file
- You will never have to use pull-down menus again

## DMHT data: Reading *dta* files into Stata

Now let's use the pull down menus to import our data...

1. Open Stata
2. Using the pull down menu: *File*→*Open*
3. Go to the directory you have stored your dta file and click it.

Note that in the results window you will see a line appear saying something like:

```
.use "c:\whatevermydirectoryiscalled\dmht5000.dta"
```

This is the syntax we could use to open the data next time (without the leading ".")

# Reading in data: Syntax

Before we start using syntax, we need a new ".do" file (A Stata syntax file) to hold our syntax.

1. Go to *File→New-Do file* (or just click the do file editor...as before).
2. A blank file should appear, on the first line type `clear`
3. On the next line (copy and) paste the line that Stata generated to read in you data:

   ```
   use "c:\whatevermydirectoryiscalled\dmht5000.dta"
   ```
4. You should now have two lines of stata code (the first clears old data from memory and the second reads in you data)
5. Highlight the text(using mouse), and run using the "do" icon

# Reading in data: Syntax for *dta* file

## My first Stata ”do” file

```
*This is a comment (Stata ignores) blah blah
clear
use "c:\whatevermydirectoryiscalled\dmht5000.dta"
```

- You should now have two lines of stata code (the first clears old data from memory and the second reads in your data)
- Highlight the text(using mouse), and run using the ”do” icon

## Pitfall:

(1) Don't forget to remove the period (”.”) that Stata puts in it's generated syntax
(2) Stata syntax is case specific (e.g. $\texttt{age} \neq \texttt{Age}$)

# Reading in data: Syntax for *csv* text file

## My first Stata ”do” file

```
*Read in data from a csv file
use "c:\whatevermydirectoryiscalled\dmht5000.csv", comma, clear
```

- ”comma” tells Stata that it is a **comma** delimited text file
- `clear` clears all old data from memory

## Hint: Document your code

Documenting your syntax (do) files (using comments) is VERY good practice. You will save a lot of time (and pain) later.

## Pitfall: Version control

Older versions of stata use the `insheet` command rather than the `use` command from importing text files.

# Saving data: Syntax for "dta" file

Once we have read in our data from a text file we may want to save it as a dta file (e.g. we may have created some new variables). To do this:

## My first Stata "do" file

```
*Save data in dta format
save "c:\mydirectory\dmht5000new.dta"
```

## Hint:

- Personally, I don't bother do this, I just save my Stata syntax in a **do file** and rerun the whole file again (which usually saves a lot of space)
- Avoid overwriting your orginal datafile (maybe add the date to the file name)

# "Learning" syntax

Before we move onto R, I would just like to make one comment:

**You are not expected to memorize Stata or R syntax!!!!**

You are just expected to remember which session we covered a particular topic, and refer back to those notes (or other resources...e.g. Stata or R help files).

**Hint: Programming in Stata and R**

USING (not memorizing) syntax is the best way to learn it

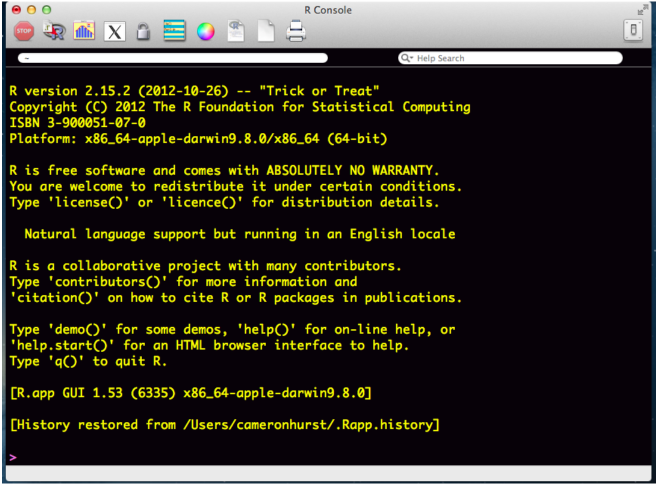
# The R statistical programming language

- R is an extremely verstaile, open-source (free) statistical programming langauage.
- Biostatisticians and researchers all around the world contribute to R by uploading their 'libraries' to the R website (`cran.r-project.org`)
- For this reason almost every analytical method used is possible in R
- BUT their is no friendly front-end (very little point and click). When you open R you are presented with a blank screen

# R vs. R-studio

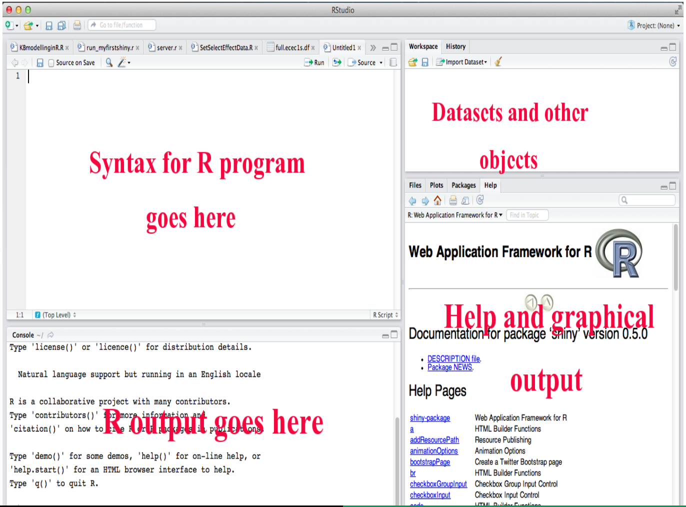
- R-studio (`www.rstudio.com`) is a front-end that wraps around R to give a little more functionaility
- It provides different windows so we can keep everything together on the same page (screen)
- We will use R-studio just to protect you from the very "low-level" of R
- However, I will show you both a screen shot of R and one of R studio

# R:Screenshot



```
R version 2.15.2 (2012-10-26) -- "Trick or Treat"
Copyright (C) 2012 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
Platform: x86_64-apple-darwin9.8.0/x86_64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

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.

[R.app GUI 1.53 (6335) x86_64-apple-darwin9.8.0]

[History restored from /Users/cameronhurst/.Rapp.history]

>
```

# R studio:Screenshot



Datascts and other

objccts

Syntax for R program

goes here

R output goes here

Help and graphical

output

```
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

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.
```

Documentation for package ‘shiny’ version 0.5.0

- DESCRIPTION file.
- Package NEWS.

Help Pages

| | |
|---|---|
| shiny-package | Web Application Framework for R |
| a | HTML Builder Functions |
| addResourcePath | Resource Publishing |
| animationOptions | Animation Options |
| bootstrapPage | Create a Twitter Bootstrap page |
| br | HTML Builder Functions |
| checkboxGroupInput | Checkbox Group Input Control |
| checkboxInput | Checkbox Input Control |
| code | HTML Builder Functions |

# R: Reading in a csv file

- Let’s start using R (in R-studio) by reading in the csv file containing the dmht data
- Note that R uses objects (entities used to store data and functions). i.e. It is an Object-oriented programming language

## R: Reading in a csv file

```
#This is a comment blah blah
dmht.df<-read.csv("c:/mydirectory/dmht5000.csv")
```

## Pitfalls: Things to be careful of

- Note that the ”#” rather than ”*” is used for comments in R
- The forwardslash (/) or double backslash (\\) is used for directories in R

# R: Reading in a csv file

Alternatively we could have set the ”working directory” first

### R: Reading in a csv file

```
#Set working directory
setwd("c:/mydirectory")


#Now read in datafile
dmht.df<-read.csv("dmht5000.csv")
```

### R ”objects”

- Note Rs use of objects: For example, I have stored the dmht data into a data frame (R dataset) called `dmht.df`.
- putting the ”.df” after the data frame is optional, but is a good idea (it reminds me of what it is e.g. my.dog, my.cat etc)

# Output from R

We can also output our data from R, but we can also do something more.

- The output object for R is called an `RData` file.
- Not only will the RData file contain our data, but also: results, models, tables
- This is very useful if we are:
  1. Working with a very big dataset (that takes a while to read in)
  2. We are only working on one or two datafiles (e.g. Your work is concerned with a single project....PhD dataset hint hint)

**R: Saving data and other outputs**

```
save.image("dmhtDataAndResults.RData")
```

# Data management in Stata and R

Now you have introduced the basic I/O of Stata and R, let's consider some basic data manipulation

- Subsetting data: Variables
- Subsetting data: Observations
- Creating new variables

# Subsetting data: Dropping variables/pbservations in Stata

## Stata: Dropping variables

```
*Drop the bmi and smoking variables
drop bmi smoke

*Alternatively we can use the ’keep’ command
*to choose a small set of variables to keep
```

## Stata: Dropping observations

```
*Drop males
drop if sex==1

*Alternatively we could ’keep’ females
keep if sex==2
```

**Note**: '==' means equals to

# Subsetting data: Dropping variables in R

In R we simply create a new dataset (from the old). Note that R uses a [ROW, COLUMN] way of referencing the [OBS, VARS]

## R: Dropping variables

```
#Drop smoke (col 7) and bmi (col 9)
mydmht.subset.df<-mydmht.df[,c(7,9)]
```

## R: Dropping observations

```
#Drop males
mydmht.subset.df<-mydmht.df[mydmht.df$sex==2,]
```

Nothing before/after the ”,” means keep all rows/columns
”$” means 'belonging to'

# More subseting in R

R's use of indexing is a very powerful tool:

## Other uses of indicies

- If we only want the first three columns:

```
subset.df<-mydmht.df[,c(1:3)]
```

- If we only want the rows 10 to 20:

```
subset.df<-mydmht.df[c(10:20),]
```

- If we only want all the rows **except** 10 to 20:

```
subsetdf<-mydmht.df[-c(10:20),]
```

- If we only want patients in the 2nd bmi group:

```
subset.df<-mydmht.df[mydmht.df[,9]==2,]
```

  or...

```
subset.df<-mydmht.df[mydmht.df$bmi==2,]
```

# Collapsing continuous variables: Stata

Often we want to collapse a continuous variable into a categorical 1

### Stata: Discretizing continuous variables

```
*Create a three class ’duration’ variable
*1=recent(< 5 years), 2=intermediate(5-10),
3=long term(>=10)
egen duraclass = cut(duradm), at(0,5,10,100)
icodes
```

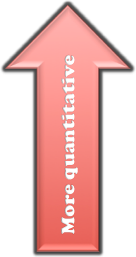
# Collapsing continuous variables: R

There are always 100s of different ways of doing things in R. I prefer to create new variables manual (as below)

### R:Discretizizing continuous variables

```
#Create new variable "duraclass"
mydmht.df$duraclass<-mydmht.df$duradm
#Assign old values to bring missing values across
mydmht.df$duraclass[mydmht.df$duradm<5]<-0
mydmht.df$duraclass[mydmht.df$duradm>=5 & +
mydmht.df$duradm<10]<-1
mydmht.df$duraclass[mydmht.df$duradm>=10]<-2
```

I will use this introduction (to stats packages) as an oppurtunity to reiterate the important link between **data type** (measurement scale) and the **statistical approach** we use to summarize (now) and formally analyze (later sessions) our data. Recall, we have several measurement scales on which variables can be measured:



- Continuous (aka quantitative/numerical)
- Categorical
  - Interval/counts
  - Ordinal (multi-category with' natural' order)
  - Nominal (multi-category without order)
  - Binary (two categories)

For today, just consider the 2 types: **Continuous** and **Categorical**

# Stata: Summary statistics for continuous variables

### Stata Summary stats for continuous variables

```
*Basic summary stats
summarize age

*More summary stats
summarize age, detail

*Now by groups
tabstat age,statistics(mean p50 sd max min) by(sex)
```

Won't bother with Stata output (you have seen it before)

# R: Summary statistics for continuous variables

### R Summary stats for continuous variables

```
#Sumamry stats for a single cont var
summary(mydmht.df$bmi)

# Need library "psych" (need to download first)
library(psych)
#x=contȯutcomes, group=catṗredictor
describe.by(x=mydmht.df$bmi, group=mydmht.df$sex)
```

```
summary(mydmht.df$bmi)
   Min    1st Qu  Median  Mean    3rd Qu  Max      NAs
  13.16   22.60   25.16   25.54   27.96   49.60    362

library(psych)
describe.by(x=mydmht.df$bmi, group=mydmht.df$sex)
group: 1
  var    n  mean   sd median trimmed  mad   min  max
1   1 1370 25.12 4.11  24.69    24.9 3.53 13.16 49.6
----------------------------------------------
group: 2
  var    n  mean   sd median trimmed  mad   min   max
1   1 3268 25.72 4.51  25.39   25.48 4.08 13.16 48.23
```

# Describing categorical variables in Stata and R

For a single categorical variable, we will generally use a frequency table and for associations among categoricals a cross-tabulation

### Stata: Describing categorical variables

```
*Freq-tab
tabulate sex

*X-tab
tabulate sex ht
```

### R: Describing categorical variables

```
# Freq tab
table(mydmht.df$sex)

# X-tab
table(mydmht.df$sex, mydmht.df$ht)
```

```
> table(mydmht.df$sex, mydmht.df$ht)

       0    1
  1  461 1014
  2 1154 2371
```

# Graphics in Stata and R

I will cover just the basics of graphs in Stata and R. We should note that R has thousands of possible graphs (There is a whole website dedicated giving hundreds of examples: `http://gallery.r-enthusiasts.com`.

I will cover:

- Histograms
- (Single) Box (and whisker) plots
- Side-by-side Box plots
- Scatter plots

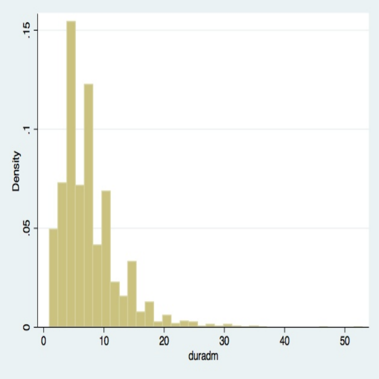
# Stata: 'Univariate' plots

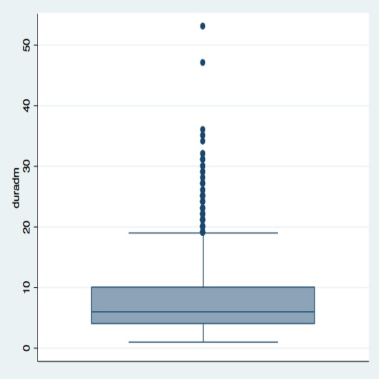
## Stata: Histograms and (single) boxplots

```
⋆Histogram
hist duradm

⋆(single) boxplot
graph box duradm
```

# Stata: 'Univariate' plots

Giving us:





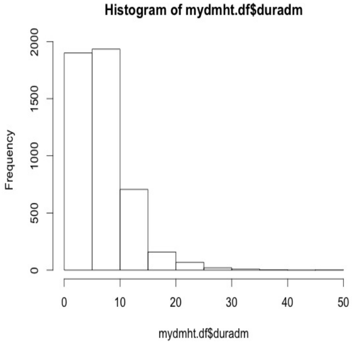
# R: 'Univariate' plots

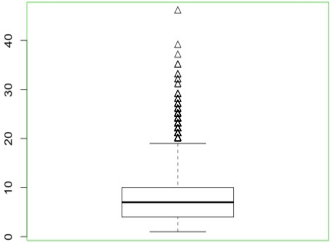
## R: Histograms and (single) boxplots

```
# Histogram
hist(mydmht.df$duradm)

# (Single) boxplot
boxplot(mydmht.df$duradm)
```

# R: 'Univariate' plots

Giving us:





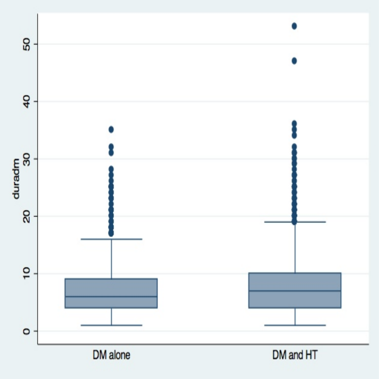
# Stata: 'Bivariate' plots

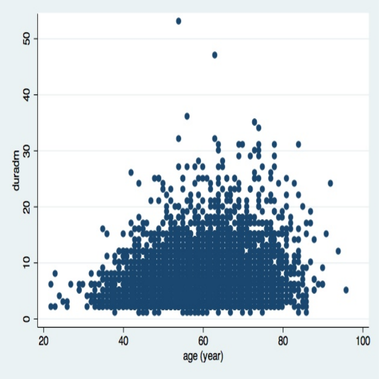
### Stata: Side-by-side boxplots and scatterplots

```
*Side by side boxplot
graph box duradm, over(ht)

*Scatter plot (note y before x)
twoway (scatter duradm age)
```

# Stata: 'Bivariate' plots

Giving us:





# R: 'Bivariate' plots

## R: Side-by-side boxplots and scatterplots

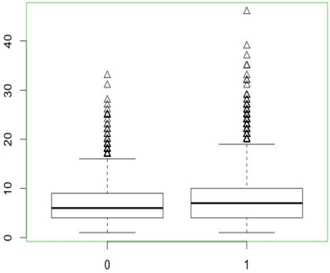
```
# Side-by-side boxplot
boxplot(mydmht.df$duradm ~mydmht.df$ht)

# Scatter plot
plot(mydmht.df$duradm~mydmht.df$age)
```

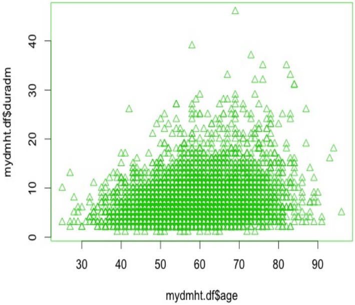
## R formula objects

Note the $\mathbf{y} \sim \mathbf{x}$ in the side by side box and scatter plots. This form of representing the outcome and predictors is VERY important. You will see a lot more of this in the days to come

# R: 'Bivariate' plots

Giving us:





# Hints and concluding remarks

- There are a lot of aspects to both Stata and R that I have not covered
- Somewhat ironically, the harder (more advanced) the stats method, the simpler R is (it is trickier for elementary statistics)
- BUT R is the most powerful/versatile stats package around
- It is also very useful for 'non-standard' statistical analyses (i.e. Advanced methods, or from a specialized area: Clinical epidemiology, Genetic/Genomic studies, etc etc)
- As with everything that is 'syntax-driven', we learn much better from USING rather than trying to memorize (i.e Use these notes as a 'reference manual'
- As with all open source packages and languages, there is a huge amount of help out there on the internet (and many 'you-tube' walk throughs

THANK-YOU!!

Questions??

YOUR TURN

PS: All of the syntax (stata do file and the R syntax file) have been uploaded to classroom