

# Intro to Social Science Data Analysis

## Seminar 1: Introduction to R and RStudio

**Christopher Gandrud**

September 6, 2012

- 1 What is the seminar for?
- 2 Getting Started with RStudio
- 3 Getting Started with R

# Seminar Purpose (1)

- ▶ This course is about learning skills that will help you **gather**, **analyse**, and **present** social science data.
- ▶ The best way to develop these skills is by **using** them.
- ▶ The seminar is an opportunity for you to **practice** using these tools. Here you can:
  - ▶ Ask me questions,
  - ▶ Collaborate with your fellow students.

# Seminar Purpose (1)

- ▶ This course is about learning skills that will help you **gather**, **analyse**, and **present** social science data.
- ▶ The best way to develop these skills is by **using** them.
- ▶ The seminar is an opportunity for you to **practice** using these tools. Here you can:
  - ▶ Ask me questions,
  - ▶ Collaborate with your fellow students.

# Seminar Purpose (1)

- ▶ This course is about learning skills that will help you **gather**, **analyse**, and **present** social science data.
- ▶ The best way to develop these skills is by **using** them.
- ▶ The seminar is an opportunity for you to **practice** using these tools. Here you can:
  - ▶ Ask me questions,
  - ▶ Collaborate with your fellow students.

# Seminar Purpose (1)

- ▶ This course is about learning skills that will help you **gather**, **analyse**, and **present** social science data.
- ▶ The best way to develop these skills is by **using** them.
- ▶ The seminar is an opportunity for you to **practice** using these tools. Here you can:
  - ▶ Ask me questions,
  - ▶ Collaborate with your fellow students.

# Seminar Purpose (1)

- ▶ This course is about learning skills that will help you **gather**, **analyse**, and **present** social science data.
- ▶ The best way to develop these skills is by **using** them.
- ▶ The seminar is an opportunity for you to **practice** using these tools. Here you can:
  - ▶ Ask me questions,
  - ▶ Collaborate with your fellow students.

# Format (1)

- ▶ In the lecture & seminar I will give you **general tools**.
- ▶ In the seminar I will give you a **goal** to complete with these tools (and others).



# Format (1)

- ▶ In the lecture & seminar I will give you **general tools**.
- ▶ In the seminar I will give you a **goal** to complete with these tools (and others).

Note: There is **rarely only one correct answer**.

I want you to **creatively** use the tools and resources available to you.

I do not want you to just copy a list of instructions.

## Open Rstudio



## Look around the main Panel.

- ▶ **Console:** Where you can enter R code.
- ▶ **Workspace/History:** Where you can see your objects and the history of commands.
- ▶ **Files/Plots/Packages/Help:** Navigate files, see the graphs you make and your packages, read help files.

## Look around the main Panel.

- ▶ **Console:** Where you can enter R code.
- ▶ **Workspace/History:** Where you can see your objects and the history of commands.
- ▶ **Files/Plots/Packages/Help:** Navigate files, see the graphs you make and your packages, read help files.

## Look around the main Panel.

- ▶ **Console:** Where you can enter R code.
- ▶ **Workspace/History:** Where you can see your objects and the history of commands.
- ▶ **Files/Plots/Packages/Help:** Navigate files, see the graphs you make and your packages, read help files.

### Create a new **source code** file:

- ▶ Click: **File** → **New** → **R Script**
- ▶ Usually write your R code here and **save your source files to Dropbox.**
- ▶ This will make your life a **lot easier.**

### Create a new **source code** file:

- ▶ Click: File → New → R Script
- ▶ Usually write your R code here and **save your source files to Dropbox.**
- ▶ This will make your life a **lot easier.**



### Create a new **source code** file:

- ▶ Click: File → New → R Script
- ▶ Usually write your R code here and **save your source files to Dropbox.**
- ▶ This will make your life a **lot easier.**

### Create a new **notebook**:

- ▶ Notebooks allow you to record **what** you do and **how** you do it.
- ▶ When you have your source code file open, click: File  
→ Compile Notebook...
- ▶ Compile a notebook when you are finished.
- ▶ We will do more of this in Week 4.

### Create a new **notebook**:

- ▶ Notebooks allow you to record **what** you do and **how** you do it.
- ▶ When you have your source code file open, click: File  
→ Compile Notebook...
- ▶ Compile a notebook when you are finished.
- ▶ We will do more of this in Week 4.

### Create a new **notebook**:

- ▶ Notebooks allow you to record **what** you do and **how** you do it.
- ▶ When you have your source code file open, click: File  
→ Compile Notebook...
- ▶ Compile a notebook when you are finished.
- ▶ We will do more of this in Week 4.

### Create a new **notebook**:

- ▶ Notebooks allow you to record **what** you do and **how** you do it.
- ▶ When you have your source code file open, click: File  
→ Compile Notebook...
- ▶ Compile a notebook when you are finished.
- ▶ We will do more of this in Week 4.

# Commenting

**Hint:** You can make your code easier to read by **regularly commenting** on it.

Use the # (hash). For example,

```
# This is a comment
```

## Objects

- ▶ R is a computer **language**, mostly used for statistical analysis.
  - ▶ The rules for writing the R language is called its **syntax**.
- ▶ R is an *object-oriented language*.
- ▶ **Objects** are like R's nouns: they are **things**.

## Objects

- ▶ R is a computer **language**, mostly used for statistical analysis.
  - ▶ The rules for writing the R language is called its **syntax**.
- ▶ R is an *object-oriented language*.
- ▶ **Objects** are like R's nouns: they are **things**.



## Objects

- ▶ R is a computer **language**, mostly used for statistical analysis.
  - ▶ The rules for writing the R language is called its **syntax**.
- ▶ R is an *object-oriented language*.
- ▶ **Objects** are like R's nouns: they are **things**.

## The Basics: Objects (2)

For example:

```
# Add 2 + 2
```

```
2 + 2
```

```
[1] 4
```

```
# Put the answer of 2 + 2 in an object called Answer
```

```
Answer <- 2 + 2
```

# Assignemt

The `<-` is the **assignment operator** it assigns something to an object.

# Tasks 1

Create **5** different objects. Explore their properties.

What can you put into an object?

What could you not put into an object?

# Basic Object Modes

All objects have a **mode** (sometimes called data type or class).

Very basic object types are **numeric**, **character strings**, and **logical** (TRUE or FALSE).

Use the `class` command to find out what an object's class is. For example:

```
class(Answer)

## [1] "numeric"
```

## Basic Object Types

- ▶ The main type of object we will use in this class is called a **data frame**.
  - ▶ We will cover data frames in detail next class.
- ▶ Today, lets look at some more basic R objects:
  - ▶ Vectors
  - ▶ Matrices

# Basic Object Types

- ▶ The main type of object we will use in this class is called a **data frame**.
  - ▶ We will cover data frames in detail next class.
- ▶ Today, lets look at some more basic R objects:
  - ▶ Vectors
  - ▶ Matrices

# Basic Object Types

- ▶ The main type of object we will use in this class is called a **data frame**.
  - ▶ We will cover data frames in detail next class.
- ▶ Today, lets look at some more basic R objects:
  - ▶ Vectors
  - ▶ Matrices



# Basic Object Types

- ▶ The main type of object we will use in this class is called a **data frame**.
  - ▶ We will cover data frames in detail next class.
- ▶ Today, lets look at some more basic R objects:
  - ▶ Vectors
  - ▶ Matrices

## Vectors

Vectors are objects with multiple numbers *or* character strings in a particular order.

They are the “workhorse” of R (Matloff 2011).

```
# A vector of a sequence of numbers
```

```
Sequence <- 1:10
```

```
Sequence
```

```
## [1] 1 2 3 4 5 6 7 8 9 10
```

```
# Non-sequential numbers
```

```
NonSeq <- c(1, 30, 53)
```

```
NonSeq
```

```
## [1] 1 30 53
```

```
# A character string vector
```

```
CharVector <- c("Christopher", "John", "Gandrud")
```

```
CharVector
```

```
## [1] "Christopher" "John"          "Gandrud"
```

# Combine Values Into a Vector

## c

The `c` command combines values into a vector.

## Matrices

Matrices are like vectors, except they have **multiple columns**.

You may remember matrices from math class:

$$\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

**Don't worry**, we never need to do matrix math in this class. R does it for us!

## Matrices 2

We can use the `cbind` function (Column Bind) to combine two of the vectors we created before.

```
# Bind the CharVector and NonSeq objects
```

```
NewMatrix <- cbind(CharVector, NonSeq)
```

```
# Display contents of NewMatrix
```

```
NewMatrix
```

```
##      CharVector      NonSeq
```

```
## [1,] "Christopher" "1"
```

```
## [2,] "John"        "30"
```

```
## [3,] "Gandrud"     "53"
```

# Tasks 3

Create a:

- ▶ Numeric vector
- ▶ Character vector
- ▶ Character matrix
- ▶ Numeric matrix



# Tasks 3

Create a:

- ▶ Numeric vector
- ▶ Character vector
- ▶ Character matrix
- ▶ Numeric matrix

# Tasks 3

Create a:

- ▶ Numeric vector
- ▶ Character vector
- ▶ Character matrix
- ▶ Numeric matrix

# Tasks 3

Create a:

- ▶ Numeric vector
- ▶ Character vector
- ▶ Character matrix
- ▶ Numeric matrix

## Commands & Functions

Commands and Functions tell R to **do something**.

Usually they do something to an object. They are like R's **verbs**.

## Commands, Functions, Arguments (2)

For example:

Lets create a set of 7 numbers: 1, 2, 3, 4, 6, 6, 7:

```
Numbers <- c(1, 2, 3, 4, 6, 6, 7)
```

Now lets take the mean (average) of these 5 numbers with the mean command

```
NumbersMean <- mean(Numbers)
```

```
NumbersMean
```

```
[1] 4.143
```

## Arguments

Arguments modify the command.

For example:

Find what arguments the round command can take by typing:

```
?round
```

This gives us the **help file** for the round command.

We can see that one argument is `digits` which specifies the number of decimal places to round to.

To add the `digits` argument just use the `=` like this:

```
round(NumbersMean, digits = 1)
```

```
[1] 4.1
```

# Nesting Commands

You can **nest** commands inside of one another

This can sometimes make your code shorter.

```
# Find Numbers' mean and round it to one decimal place  
round(mean(Numbers), digits = 1)
```

```
## [1] 4.1
```



## Tasks 3

Find and use **2** other commands. (Hint: `library(help = "base")` or `library(help = "graphics")`)

Explore their properties.

# Installing New Functions

One of R's great strengths is that there are thousands of free, open source add-on packages that let you greatly expand what you can do in R.

To **install** these packages use the `install.packages` command.

Once the package is installed you can **load it** in your R session with the `library` function.

## Tasks 4

Install and load the *WDI* package.

# Syllabus Change

Please check the online version of the syllabus.

I made a small change to the readings for the next two weeks.