

# 1. Introduction to R and Rstudio

Introduction to Social Network Analysis in R

---

Dr. Uma Ravat

University of California at Santa Barbara

# Welcome

## Introductions - Let's get to know one another

1. Introduction to R
2. Introduction to Rstudio
3. R Markdown documents

# Welcome

---

# Welcome

umaravat@ucsb.edu

# Introduction to Social Network Analysis(SNA) in R

1. Introduction to R as we will use R language for SNA for the rest of the lecture series
2. Introduction to basic concepts in SNA and visualization of networks.
3. Metrics - Individual nodes.
4. Metrics - Whole network.
5. (Time permitting) Network models, algorithms and Inference.
6. SNA in Education, Surveys and Data Manipulation.
7. Ongoing Research Project with Keio University.

## **Introductions - Let's get to know one another**

---

# Introductions - Let's get to know one another

Go ahead and tell everybody

1. your name,
2. one of the following:
  - Your Hometown
  - Your favorite food or snack
  - Your hobbies and interests
  - An interesting fact about yourself
  - Two Truths and a Lie
  - Or anything else!
3. Why are you interested in this lecture series?

After you're done sharing, pick someone who hasn't shared yet to go next

# 1. Introduction to R

---



# What *is* R?



- R is an open-source(free) statistical **programming language**
- R is also an environment for statistical computing and **graphics**
- It's easily extensible with *packages* (more on this later)
- R is based on the S language, which was developed by Bell laboratories in the 90's
- Home page: <http://www.r-project.org>

## Why *use* R?

- Open source (free)
- Runs on just about any platform
- Great visualization capabilities (ggplot2)
- Read/write from/to various data sources
- Scripting language (interpreted)
- Massive library of data manipulation and statistical packages (including great network packages like statnet and igraph)

## Why *use* R?

- several functions are vectorized in R
  - vectorized code saves time (any typing)
- There is an optimized engine—a basic linear algebra system (BLAS)—that is highly efficient at solving linear algebra problems
- A lot of R functions are written in C (or variants)

For more details see: <https://www.noamross.net/archives/2014-04-16-vectorization-in-r-why/>

# R and RStudio

R: Engine



RStudio: Dashboard



- R is a programming language.
- RStudio is a convenient interface for R called an **IDE** (integrated development environment), e.g. *"I write R code in the RStudio IDE"*



## 2. Introduction to Rstudio

---

RStudio
Project: (None)

Environment History Connections Tutorial

Files Plots Packages Help Viewer Presentation

Run Publish

Source Visual Outline

```

39 We can easily change which countries are being plotted by changing
   which countries the code above filter's for. Note that the
   country name should be spelled and capitalized exactly the same
   way as it appears in the data. See the [Appendix](#appendix) for a
   list of the countries in the data.
40
41 ```{r plot-yearly-yes-issue, fig.width=10, fig.height=6,
   message=FALSE}
42 unvotes %>%
43   filter(country %in% c("UK & NI", "US", "Turkey")) %>%
44   mutate(year = year(date)) %>%
45   group_by(country, year, issue) %>%
46   summarize(percent_yes = mean(vote == "yes")) %>%
47   ggplot(mapping = aes(x = year, y = percent_yes, color =
   country)) +
48   geom_point(alpha = 0.4) +
49   geom_smooth(method = "loess", se = FALSE) +
50
51 1: UN Votes

```

Console Render Background Jobs

.../yt-01a-un-votes/unvotes.Rmd

```

--toc-depth 3 --variable toc_float=1 --variable toc_selectors=h1,h2,h3
--variable toc_collapsed=1 --variable toc_smooth_scroll=1 --variable to
c_print=1 --template /Library/Frameworks/R.framework/Versions/4.2/Resour
ces/library/rmarkdown/rmd/default.html --no-highlight --variable highl
ightjs=1 --variable theme=bootstrap --mathjax --variable 'mathjax-url=ht
tps://mathjax.rstudio.com/latest/MathJax.js?config=TeX-AMS-MML_HTMLorM
L' --include-in-header /var/folders/wq/cjgk7v356dq1lj3j90802q9h0000gq/
T/Rtmpbfufuw/rmarkdown-str10d53f7g771d.html
output file: unvotes.knit.md

```

Output created: unvotes.html

Environment History Connections Tutorial

Files Plots Packages Help Viewer Presentation

Run Publish

We can easily change which countries are being plotted by changing which countries the code above filter's for. Note that the country name should be spelled and capitalized exactly the same way as it appears in the data. See the Appendix for a list of the countries in the data.

```

unvotes %>%
  filter(country %in% c("UK & NI", "US", "Turkey")) %>%
  mutate(year = year(date)) %>%
  group_by(country, year, issue) %>%
  summarize(percent_yes = mean(vote == "yes")) %>%
  ggplot(mapping = aes(x = year, y = percent_yes, color = countr
y)) +
  geom_point(alpha = 0.4) +
  geom_smooth(method = "loess", se = FALSE) +
  facet_wrap(~issue) +
  scale_y_continuous(labels = percent) +
  labs(
    title = "Percentage of 'Yes' votes in the UN General Assembl
y",
    subtitle = "1946 to 2015",
    x = "% Yes",
    y = "Year",
    color = "Country"
  )

```

Percentage of 'Yes' votes in the UN General Assembly  
1946 to 2015

# Tour of RStudio panes

The image shows the RStudio interface with four panes, each labeled with a text overlay:

- Scripts**: The top-left pane shows an R script with comments and code for installing and loading packages, and importing data.
- Environment/History**: The top-right pane shows the Environment and History tabs. The Environment tab is active, displaying the Global Environment.
- Files/Plots/Packages/Help**: The bottom-right pane shows the Files, Plots, Packages, and Help tabs. The Files tab is active, displaying a file explorer view of the current project.
- Console/Terminal**: The bottom-left pane shows the Console and Terminal tabs. The Console tab is active, displaying the R prompt and output.



# R packages

- **Packages** are the fundamental units of reproducible R code. They include reusable R functions, the documentation that describes how to use them, and sample data
- There are over 18,000 R packages available on **CRAN** (the Comprehensive R Archive Network)<sup>1</sup>
- We will use various packages in this course such as base R, graphics, igraph, igraphdata etc.

1 Community contributed packages are stored at CRAN  
Comprehensive R Archive Network

## Your Turn: 1. Tour of Rstudio panes

Download files from url provided.

Go to RStudio and open the file named  
`1_WA_IntroRstudioR.Rmd`

We will go over 1. Tour of four Rstudio panes

If you finish, react with a “thumbs up” in Zoom and help your class mates who may need help finishing.

*To understand computations in R, two slogans are helpful:*

*Everything that exists is an object.*

*Everything that happens is a function call.*

— John Chambers

Even a *function* is an *object*.

## R essentials (continued)

- **Functions** are (most often) verbs, followed by what they will be applied to in parentheses:

```
do_this(to_this)
```

Here `do_this` is the function and `to_this` is the **argument** to the function

```
do_that(to_this, to_that, with_those)
```

Here `do_that` is the function and `to_this`, `to_that`, `with_those` are the three **arguments** to the `do_that` function

- **Packages** are installed with the `install.packages` function and loaded with the `library` function, once per session:

```
install.packages("package_name")  
library(package_name)
```

We will use `Rmarkdown` documents to write our R code or programs.

Before we go on a tour of R essentials, let's look at `Rmarkdown` documents first

### 3. R Markdown documents

---

### 3. R Markdown documents

- with `.Rmd` extension
- **rmarkdown** is an R package
  - write code and prose in **reproducible** computational documents

[rmarkdown.rstudio.com](https://rmarkdown.rstudio.com)



Take a look at the **Rmarkdown gallery**

# Anatomy of RMarkdown documents

The image shows a screenshot of an R Markdown document editor. The document content is as follows:

```
1: ---
2: title: "UK Votes"
3: author: "Your name here"
4: date: "'r Sys.Date()"
5: output:
6:   html_document:
7:     fig: yes
8:     toc: false
9: ---
10:
11: ## Introduction
12:
13: How do various countries vote in the United Nations General Assembly, how have their voting patterns evolved throughout time, and how similarly or differently do they view certain issues? Answering these questions (at a high level) is the focus of this analysis.
14:
15: ## Packages
16:
17: We will use the tidyverse, tidyr, and ggplot2 packages for the data wrangling and visualization, and the DT package for interactive display of tabular output.
18:
19: ```{r load-packages, warning=FALSE, message=FALSE}
20: library(tidyverse)
21: library(tidyr)
22: library(ggplot2)
23: library(DT)
24:
25: ## Data
26:
27: The data we're using originally came from the "rpubs" package, but it's been modified a bit (by joining the various data frames provided in the package) to help you get started with the analysis.
28:
29: ```{r load-data}
30: votes <- read_csv("data/votes.csv")
31:
32:
33: ## UK voting patterns (shorting)
34:
35: Let's create a data visualization that displays how the voting record of the UK & NI changed over time on a variety of issues, and compares it to two other countries: US and Turkey.
36:
37: We can easily change which countries are being plotted by changing which countries the code above "filter" for. Note that the country name should be spelled and capitalized exactly the same way as it appears in the data. See the Depends\(\) function for a list of the countries in the data.
38:
39: ```{r plot-yearly-votes, fig.width=8, fig.height=8, message=FALSE}
40: library(tidyverse)
41: filter(country %in% c("UK & NI", "US", "Turkey")) %>%
42:   mutate(year = year(date)) %>%
43:   group_by(country, year, issue) %>%
44:   summarise(percent = sum(vote == "yes") / n()) %>%
45:   apply(mapping ~ aes(x = year, y = percent, color = country)) +
46:   geom_point(alpha = 0.5) +
47:   geom_smooth(method = "loess", se = FALSE) +
48:   facet_wrap(issue) +
49:   scale_y_continuous(labels = percent) +
50:   labs(
51:     title = "Percentage of 'yes' votes in the UN General Assembly",
52:     subtitle = "1946 to 2015"
53:   )
54: print(votes)
```

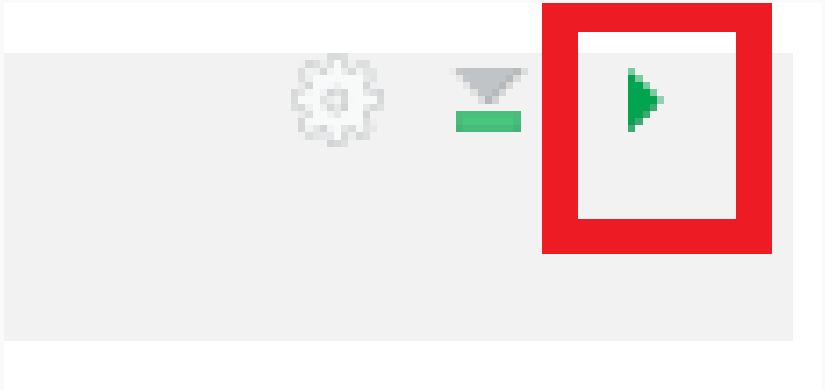
Handwritten annotations on the image:

- Knit** (red arrow pointing to the Knit button)
- I yam I** (purple text)
- narrative** (green arrow pointing to the Introduction section)
- I code chunk for loading packages** (blue text pointing to the first code chunk)
- I code chunk for plotting graph** (blue text pointing to the second code chunk)
- hyper link** (orange arrow pointing to the [Depends\(\) function](#) link)
- Run the code chunk** (green arrow pointing to the Run button)

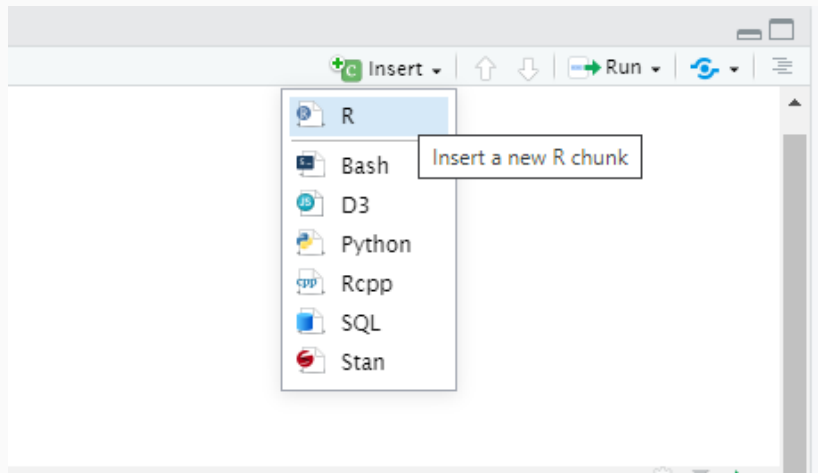
- Code goes in code chunks, defined by three backticks
- narrative goes outside of chunks
- Simple markdown syntax for text



## Demo - Run Code



## Demo - Adding Chunks



# R Markdown help in RStudio

## R Markdown Cheat Sheet Help -> Cheatsheets

### R Markdown :: CHEAT SHEET

#### What is R Markdown?



**.Rmd files** - An R Markdown (.Rmd) file is a record of your research. It contains the code that a scientist needs to reproduce your work along with the narration that a reader needs to understand your work.

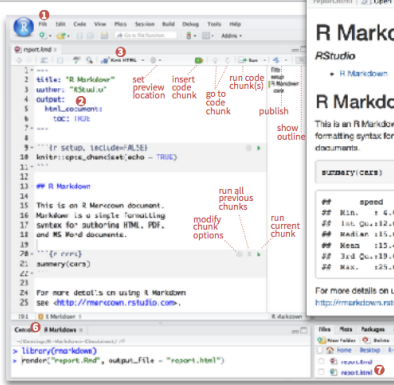
**Reproducible Research** - At the click of a button, or the type of a command, you can rerun the code in an R Markdown file to reproduce your work and export the results as a finished report.

**Dynamic Documents** - You can choose to export the finished report in a variety of formats, including html, pdf, MS Word, or RTR documents; html or pdf based slides, Notebooks, and more.

#### Workflow



- 1 Open a new .Rmd file at File ► New File ► R Markdown. Use the wizard that opens to pre-populate the file with a template
- 2 Write document by editing template
- 3 Knit document to create report; use knit button or `render()` to knit
- 4 Preview Output in IDE window
- 5 Publish (optional) to web server
- 6 Examine build log in R Markdown console
- 7 Use output file that is saved along side .Rmd



#### render

Use `marmdown::render()` to render/knit at cmd line. Important args:

**input** - file to render  
**output\_format**

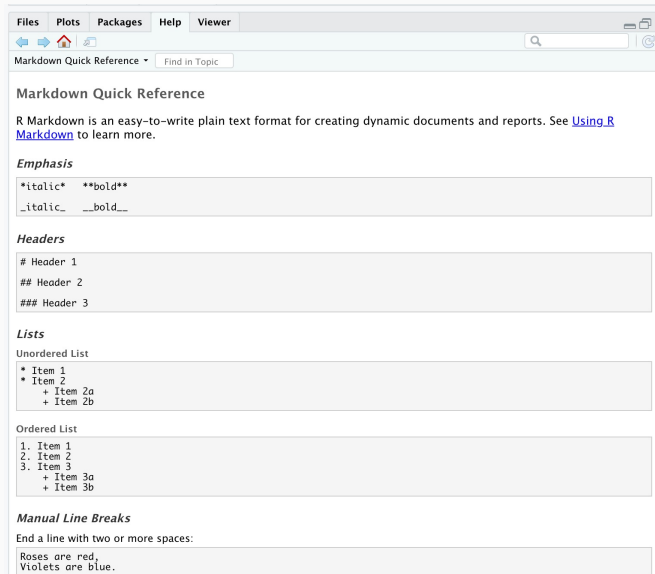
**output\_options** -  
List of render  
options (as in YAML)

**output\_file**  
**output\_dir**

**params** - list of  
params to use

# Markdown Quick Reference

## Help -> Markdown Quick Reference



The screenshot shows a web browser window with a navigation bar at the top containing tabs for 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. The 'Help' tab is active. Below the navigation bar is a search bar with the text 'Find in Topic'. The main content area is titled 'Markdown Quick Reference' and contains the following sections:

**Markdown Quick Reference**

R Markdown is an easy-to-write plain text format for creating dynamic documents and reports. See [Using R Markdown](#) to learn more.

**Emphasis**

```
*italic*  **bold**  
_italic_  __bold__
```

**Headers**

```
# Header 1  
## Header 2  
### Header 3
```

**Lists**

Unordered List

```
* Item 1  
* Item 2  
  + Item 2a  
  + Item 2b
```

Ordered List

```
1. Item 1  
2. Item 2  
3. Item 3  
  + Item 3a  
  + Item 3b
```

**Manual Line Breaks**

End a line with two or more spaces:

```
Roses are red,  
Violets are blue.
```

## (Optional) Your Turn: Rmarkdown basics

- We will not cover this here.
- You may do this to familiarize yourself with Rmarkdown syntax and basics
- Open `2_WA_RmarkdownBasics.Rmd` and complete on your own
- Ask questions!

Go to RStudio and we will now go over 2. Tour of R essentials.

# Lecture 1 Summary: Introduction to R

- Rstudio
- R
- Rmarkdown essentials.

## Next session:

2. Introduction to basic concepts in SNA and visualization of networks.

In preparation for next session on Friday, you may take a look at network datasets available in `igraph` package and read the description of networks contained in this package at:

<https://cran.r-project.org/web/packages/igraphdata/igraphdata.pdf>