



STOR 320 Workflow in RMarkdown

Lecture 2

Yao Li

Department of Statistics and Operations Research

UNC Chapel Hill



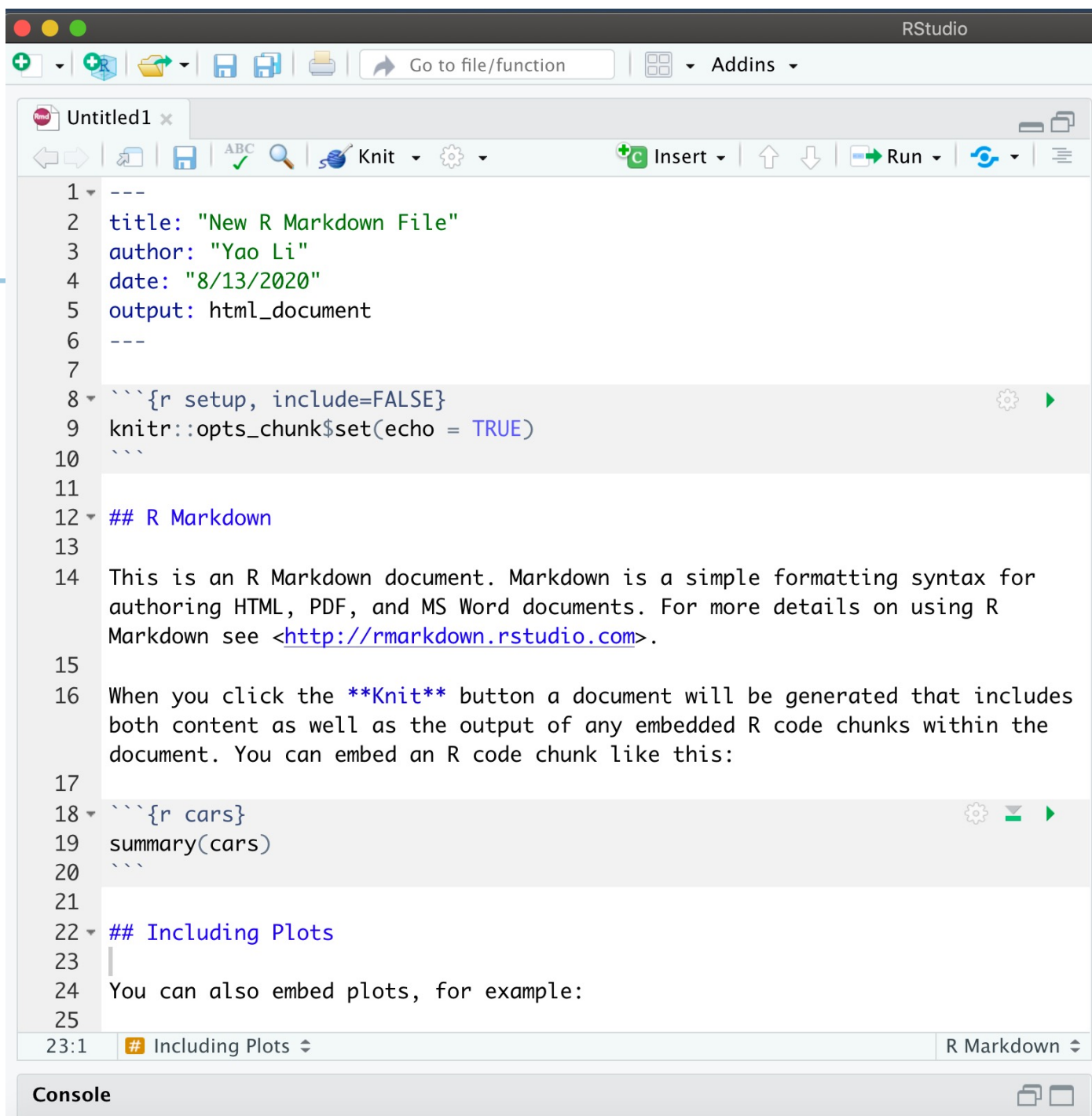
Workflow Information

- Chapters Discussing Workflow
 - Chapter 4: Basics
 - Chapter 6: Rscripts
 - Chapter 8: Projects
- Our Focus is on Workflow Within RMarkdown
- Today's Lecture on RMarkdown
 - Running R Code
 - Objects
 - Functions



Essential Reads

- Highly Advised Reading
 - Chapter 27: RMarkdown
 - Basics
 - Text Formatting
 - Code Chunks
 - Chapter 28: More ggplot Info
 - Labeling
 - Annotating
 - Scaling
 - Zooming
 - Themes
 - Saving Graphics



```
1 ---
2 title: "New R Markdown File"
3 author: "Yao Li"
4 date: "8/13/2020"
5 output: html_document
6 ---
7
8 ```{r setup, include=FALSE}
9 knitr::opts_chunk$set(echo = TRUE)
10 ```
11
12 ## R Markdown
13
14 This is an R Markdown document. Markdown is a simple formatting syntax for
15 authoring HTML, PDF, and MS Word documents. For more details on using R
16 Markdown see <http://rmarkdown.rstudio.com>.
17
18 When you click the Knit button a document will be generated that includes
19 both content as well as the output of any embedded R code chunks within the
20 document. You can embed an R code chunk like this:
21
22 ```{r cars}
23 summary(cars)
24 ```
25
26 ## Including Plots
27
28 You can also embed plots, for example:
```



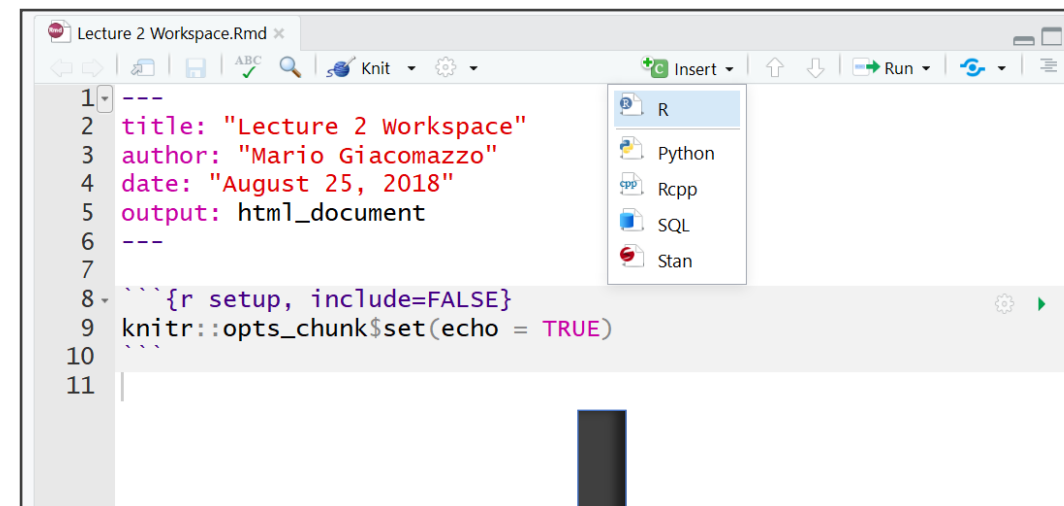
Rmarkdown File

[Cheat Sheet](#)

Placing Code in RMarkdown

- Code Chunks (Mini Rscripts)
 - R, Python, SQL, Rcpp (C++)
 - Inserting R Chunks

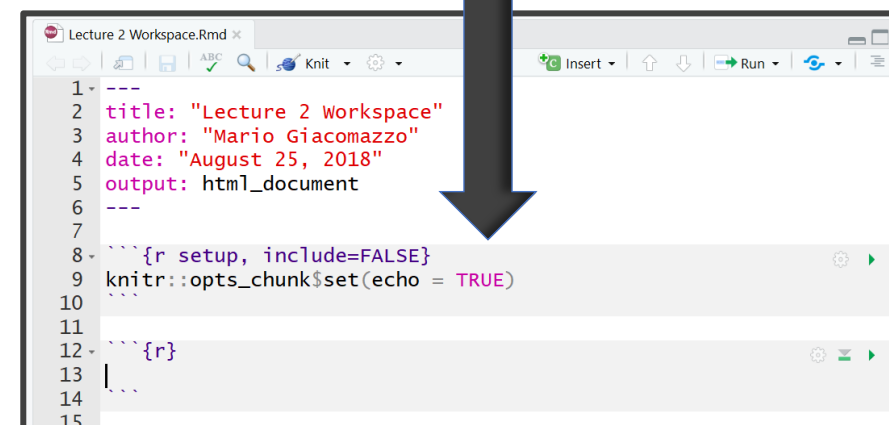
- Method 1:



- Method 2: Ctrl+Alt+I

- Method 3: Type ````${r}````

Put R code here





Inline Code in RMarkdown

```
```{r}
a <- c(1,2,3)
```
```

The sum of vector `a` is ``r sum(a)``.

Knit to HTML

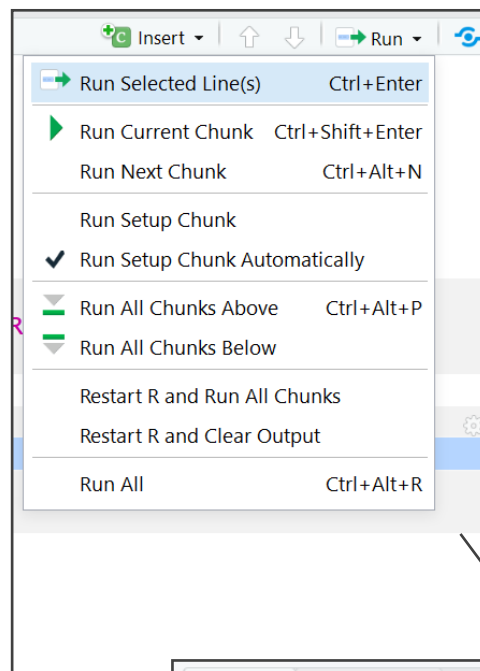
```
a <- c(1,2,3)
```

The sum of vector *a* is 6.

Running Code in RMarkdown

- Various Ways
 - Highlighted Code

```
{r}  
x=3  
x
```

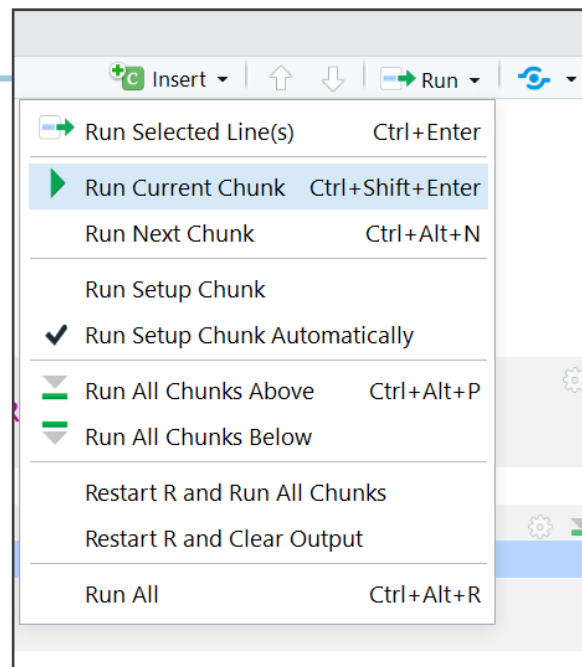


Ctrl+Enter

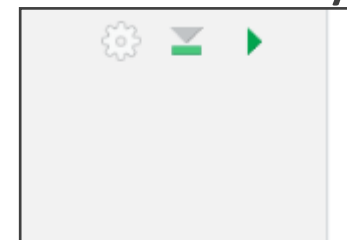
```
Console Terminal x  
~/  
> x=3  
> x  
[1] 3  
> |
```

Running Code in RMarkdown

- Various Ways (Cont.)
 - Chunking It (Recommended)



Press Play



Ctrl+Shift+Enter





Order

- Order Matters

The screenshot shows a Jupyter Notebook with two code cells. The first cell contains the following code:

```
{r}  
#Created Variables x and y assigned to 3 and 4 respectively  
x=3  
y=4  
print(c(x,y))
```

The second cell contains the following code:

```
{r}  
x+y #Addition  
x-y #Subtraction  
x*y #Multiplication  
x/y #Division  
x^y #Powers  
x%%y #Modulus (x mod y)
```

An arrow points from the 'Why?' box to the 'x' in the first line of the second cell. The error message 'Error: object 'x' not found' is displayed at the bottom of the notebook.

Why?
Environment is empty

Error: object 'x' not found



Order

- Order Matters (Cont.)
 - Run First Chunk
 - Then, Run Second Chunk

```
{r}
#Created Variables x and y assigned to 3 and 4
respectively
x=3
y=4
print(c(x,y))
```

[1] 3 4

```
{r}
x+y #Addition
x-y #Subtraction
x*y #Multiplication
x/y #Division
x^y #Powers
x%%y #Modulus (x mod y)
```

[1] 7
[1] -1
[1] 12
[1] 0.75
[1] 81
[1] 3

```
{r}
#Created variables x and y assigned to 3 and 4 respectively
x=3
y=4
print(c(x,y))
```

[1] 3 4

| Environment | | History | Connections |
|--------------------|---|----------------|-------------|
| Global Environment | | Import Dataset | |
| Values | | | |
| x | 3 | | |
| y | 4 | | |



Run All Previous Chunks

```
{r}
#Created Variables x and y assigned to 3 and 4 respectively
x=3
y=4
print(c(x,y))

[1] 3 4

{r}
x+y #Addition
x-y #Subtraction
x*y #Multiplication
x/y #Division
x^y #Powers
x%%y #Modulus (x mod y)

[1] 7
[1] -1
[1] 12
[1] 0.75
[1] 81
[1] 3

{r}
log(x) #Logarithm of x
abs(x-y) #Absolute value of x-y
exp(x) #e^x|
```

- Order Matters (Cont.)

Runs All Previous Chunks



Run All Previous Chunks

```
{r}
#Created Variables x and y assigned to 3 and 4 respectively
x=3
y=4
print(c(x,y))

[1] 3 4

{r}
x+y #Addition
x-y #Subtraction
x*y #Multiplication
x/y #Division
x^y #Powers
x%%y #Modulus (x mod y)

[1] 7
[1] -1
[1] 12
[1] 0.75
[1] 81
[1] 3

{r}
log(x) #Logarithm of x
abs(x-y) #Absolute value of x-y
exp(x) #e^x

[1] 1.098612
[1] 1
[1] 20.08554
```

- Order Matters (Cont.)

Then, Run Current Chunk



Chunk Options

```
```${r,eval=F}  
p3<-p2+geom_smooth(COMPLETE_INSIDE)
p3
```
```



| Option | Run code | Show code | Output | Plots | Messages | Warnings |
|-------------------|----------|-----------|--------|-------|----------|----------|
| eval = FALSE | - | | - | - | - | - |
| include = FALSE | | - | - | - | - | - |
| echo = FALSE | | - | | | | |
| results = "hide" | | | - | | | |
| fig.show = "hide" | | | | - | | |
| message = FALSE | | | | | - | |
| warning = FALSE | | | | | | - |

[Chunk Options](#)



Objects in R: Vector and Matrix

```
## {r}
#Numeric Vector Named x
x=c(3,2,1,5,7,8)
#Prints x
x
#Third Element of x
x[3]
#Character Vector Named y
y=c("H","T","H","T","H","T")
#Fifth Element of y
y[5]
#3x2 Matrix Named z
z=matrix(c(3,2,1,5,7,8),
        nrow=2,ncol=3,byrow=T)
#Prints z
z
#First Row of z
z[1,]
#1st and 3rd Column of z
z[,c(1,3)]
```

```
[1] 3 2 1 5 7 8
[1] 1
[1] "H"
      [,1] [,2] [,3]
[1,]    3    2    1
[2,]    5    7    8
[1] 3 2 1
      [,1] [,2]
[1,]    3    1
[2,]    5    8
```

- Many Types of Objects
 - Vector and Matrix



Objects in R: Dataframe

```
{r}
#Create Tibble named tbl
tbl<-tibble(x=x,y=y)
#Print tbl
tbl
```

| x | y |
|-------|-------|
| <dbl> | <chr> |
| 3 | H |
| 2 | T |
| 1 | H |
| 5 | T |
| 7 | H |
| 8 | T |

6 rows

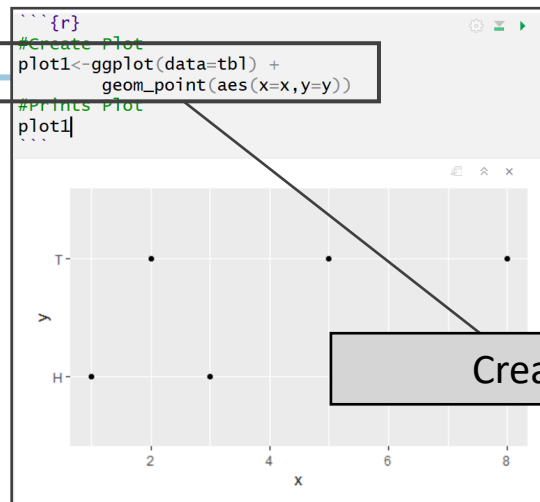
```
{r}
#Create Dataframe named df
df<-data.frame(x=x,y=y)
#Print df
df
```

| x | y |
|-------|--------|
| <dbl> | <fctr> |
| 3 | H |
| 2 | T |
| 1 | H |
| 5 | T |
| 7 | H |
| 8 | T |

6 rows

- Many Types of Objects (Cont.)
 - Tibble/Dataframe

Objects in R: Lists



- Many Types of Objects (Cont.)
 - Lists (Combines Different Objects)

Global Environment

Data

df 6 obs. of 2 variables

plot1 List of 9

```
data :Classes 'tbl_df', 'tbl' and 'data.frame': 6 obs. of 2 v...
..$ x: num [1:6] 3 2 1 5 7 8
..$ y: chr [1:6] "H" "T" "H" "T" ...
layers :List of 1
..$ :Classes 'LayerInstance', 'Layer', 'ggproto', 'gg' <ggpro...
aes_params: list
compute_aesthetics: function
compute_geom_1: function
compute_geom_2: function
compute_position: function
compute_statistic: function
data: waiver
draw_geom: function
finish_statistics: function
geom: <ggproto object: Class GeomPoint, Geom, gg>
aesthetics: function
default_aes: uneval
draw_group: function
draw_key: function
draw_layer: function
draw_panel: function
extra_params: na.rm
handle_na: function
```

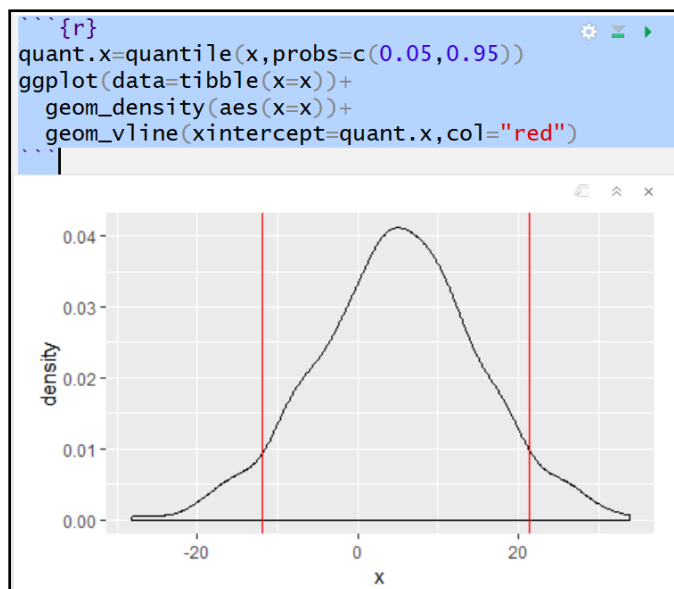



Functions in R

- Many Types of Functions
 - You: Input Objects and Specify Arguments (Defaults Exist)
 - Function: Outputs Objects
 - Example `> quantile()`
 - Input: Vector and Specified Probabilities
 - Output: Desired Percentiles
 - For online help, `> ?quantile`

Functions in R

```
Console Terminal x
~/
> #Randomly Draw 1000 Samples from
> #Normal Distribution with Mean=5 and SD=10
> x=rnorm(1000,mean=5,sd=10)
> mean(x) #Prints Sample Mean
[1] 4.905269
> sd(x) #Prints Sample SD
[1] 10.01766
> quantile(x) #Default Quantiles (Min,Quantiles,Max)
      0%      25%      50%      75%     100%
-28.232597 -1.480456  5.022031 11.433746 33.929228
> quantile(x,probs=c(0.05,0.95)) #Middle 90%
      5%      95%
-11.98847 21.30757
```



- Many Types of Functions (Cont.)
 - Example (Cont.)



Rmarkdown Training

Now, let us

PRACTICE

Download the Rmd for Tutorial 2 to Your Computer from the Course Website and open the file in RStudio