



Workflow in RMarkdown

Workflow Info



- Chapters Discussing Workflow
 - Chapter 2: Basics
 - Chapter 4: Rscripts
 - Chapter 6: Projects
- Our Focus is on Workflow Within

Rmarkdown
- Today's Lecture on RMarkdown
 - Running R Code
 - Objects
 - Functions

Essential Reads

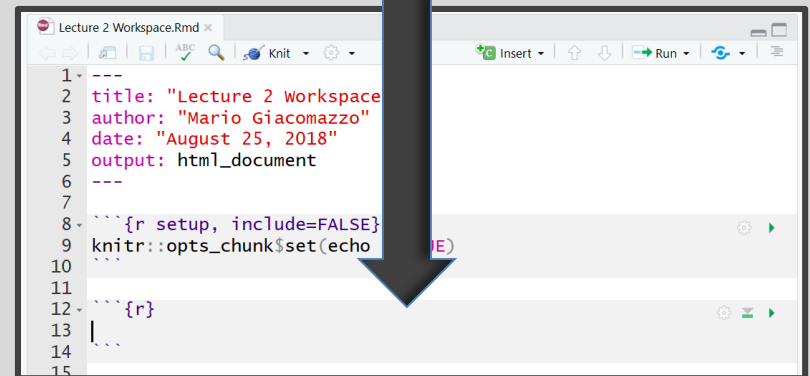
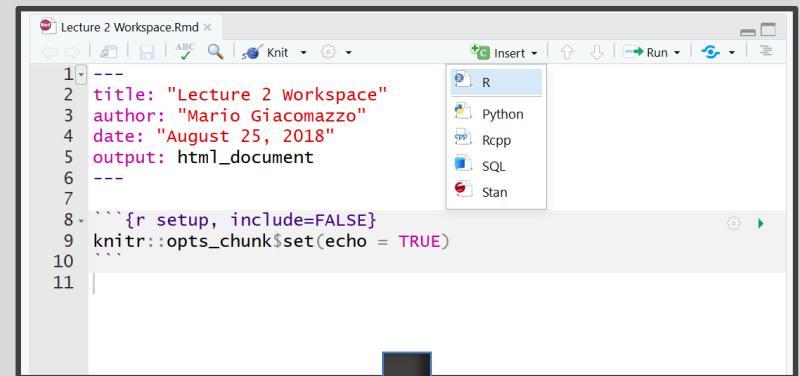


- Highly Advised Reading
 - Chapter 21: RMarkdown
 - Basics
 - Text Formatting
 - Code Chunks
 - Chapter 22: More ggplot Info
 - Labeling
 - Annotating
 - Scaling
 - Zooming
 - Themes
 - Saving Graphics

Placing Code in RMarkdown



- Code Chunks (Mini Rscripts)
 - R, Python, SQL, Rcpp (C++)
 - Inserting R Chunks
 - Method 1:



- Method 2: Ctrl+Alt+I

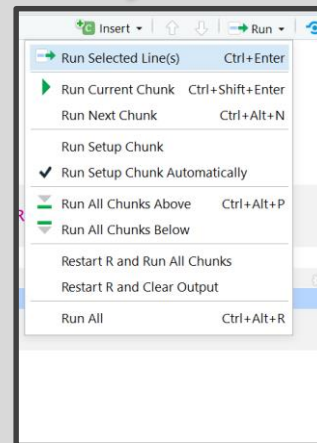
Running Code in RMarkdown



- Various Ways
 - Highlighted Code

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

Ctrl+Enter

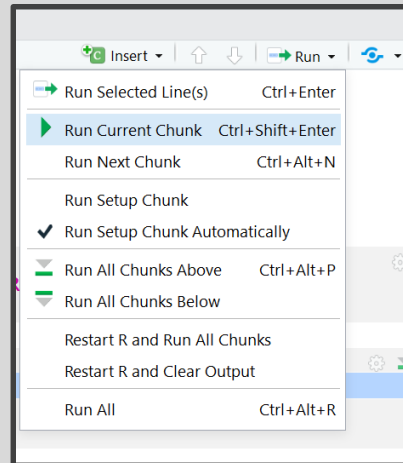


```
~/    
> x=3  
> x  
[1] 3  
> |
```

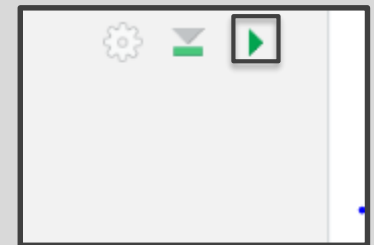
Running Code in RMarkdown



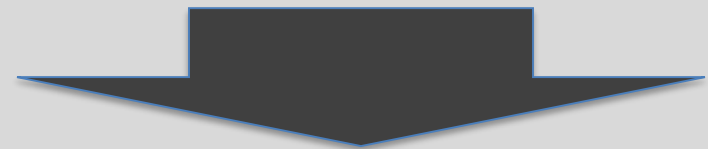
- Various Ways (Cont.)
 - Chunking It (Recommended)



Press
Play



Ctrl+Shift+Enter



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

[1] 3

Running Code in RMarkdown



- Order Matters

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

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

Why?

Environment is empty

Error: object 'x' not found

Running Code in RMarkdown



- Order Matters (Cont.)
 - Run First Chunk

```
{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		
Values		
x	3	
y	4	

- 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

Running Code in RMarkdown



- Order Matters (Cont.)
 - Super Chunky

```
{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|
```

Runs All Previous Chunks

Running Code in RMarkdown



- Order Matters (Cont.)
 - Super Chunky (Cont.)

```
{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

Then, Run Current Chunk

Objects in R



- Many Types of Objects
 - 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
```

Objects in R



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

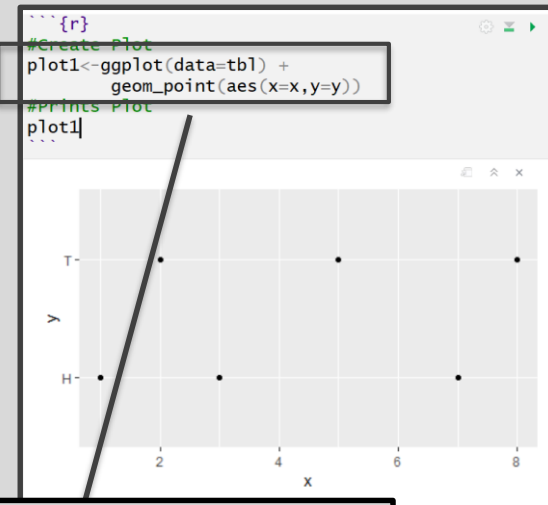
```
{r}  
#Create Dataframe named df  
df<-data.frame(x=x,y=y)  
#Print df  
df
```

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

6 rows

Objects in R

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



Creates Long List

```
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



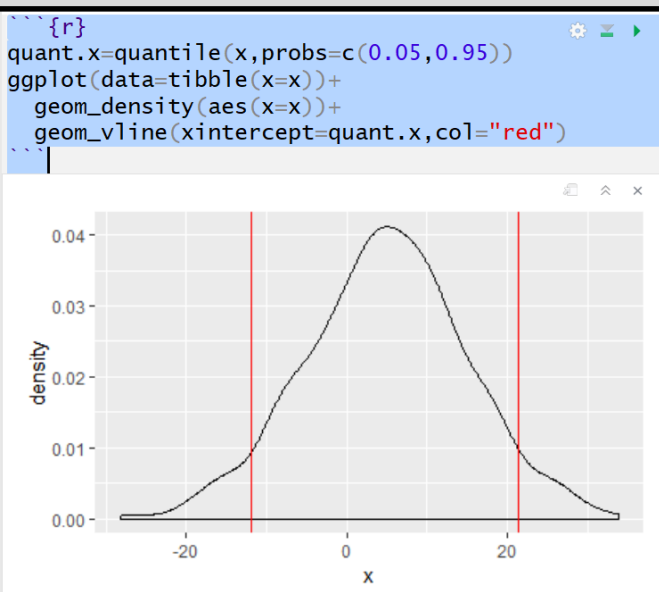
- Many Types of Functions
 - You: Input Objects and Specify Arguments (Defaults Exist)
 - Function: Outputs Objects
 - Example `> quantile()`
 - Input: Vector and Specified Percentiles
 - Output: Desired Percentiles
 - For online help, `> ?quantile`

Functions in R



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

```
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
```



Closing



Disperse
and Make
Reasonable
Decisions