An Intro to R for Non-Programmers II

William F. Lamberti [1]

George Mason University

September 26, 2017

[1] MS in Statistical Sciences
Student in PhD Computational Sciences
and Informatics at George Mason University

Outline

Introduction

- 1 hour course (7:30 PM - 8:30 PM)
- For those who know some of the basics of R but want to learn more
- Will reference `RGalleon.com` pages for additional information
- Will *not* go over the "why" of statistics (simply do not have the time)
- We plan to have two additional sessions on October 24 and November 14
- If you have any questions, please feel free to ask at any point

About Me

- BA in Mathematical Statistics from The College of New Jersey
- Master's in Statistics from George Mason University
- Likes include running, reading, video games, cooking, hiking

Assumed Material

Assumed material covered in first lecture series.
Available here:
http://www.rgalleon.com/talks/george-mason-university-february-28-2017/

Exercise

- Download R and setup R
- Go to `http://www.rgalleon.com/talks/george-mason-university-february-28-2017/`
- Download the exercise associated files
- **Goal: Calculate the mean and standard deviation from the data.**
- Bonus: Create a histogram of the data.
- You have 10 minutes. I will be available if you have any questions.
- We will go over the answer together after 10 minutes have passed

Example Code

```
1  dir()
2
3  #Loading data
4  load("RLecture_ex_data.RData")
5
6  ls()
7
8  length(data)
9
10 data
11
12 mean(data)
13 sd(data)
14
15 hist(data) #histogram
16
17 windows() #makes 2nd space for 2nd graphic
18 hist(data, main="Histogram of Exercise 1 Data
       ", col='blue')
```

Random Variables: Can Simulate in R

- R can simulate, or create, random observations from a variety of distributions
  - Normal
  - Poisson
  - Binomial
  - etc....
- Used for a variety of reasons
  - Testing new algorithms
  - Confirming results under "normal" conditions
  - Creating HW examples for students
- General framework is r"distribution"()
  - rnorm()
  - rpois()
  - rbinom()

Example Code: rpois()

```
1 #checking how rpois() works
2 ?rpois()
3
4 #creating 100 Poisson(lamba=5) R.V.
5 pois.data<-rpois(n=100, lambda=5)
6
7 #plotting the data
8 plot(pois.data)
9 hist(pois.data)
```

Seeds: What do they grow into?

- Seeds allow for reproducing the same results
- Used in:
  - machine learning for techniques such as cross validation
    - Taking random samples from a large data set
- Utilized via set.seed("Integer")

Example Code: set.seed()

```
1 #setting the seed
2 set.seed(12345)
3
4 #creating 100 Poisson(lamba=5) R.V.
5 pois.data<-rpois(n=100, lambda=5)
6
7 #plotting the data
8 plot(pois.data)
9 hist(pois.data)
```

Exercise

- Plot 1,000 Normal(0, 1) random variables as a histogram
- Set the seed to 43110
- Use blue bars with cyan a cyan outline
- Change the main title to "1,000 N(0, 1) Data"
- Change the x axis title to "x"
- You have 10 minutes

Example Code

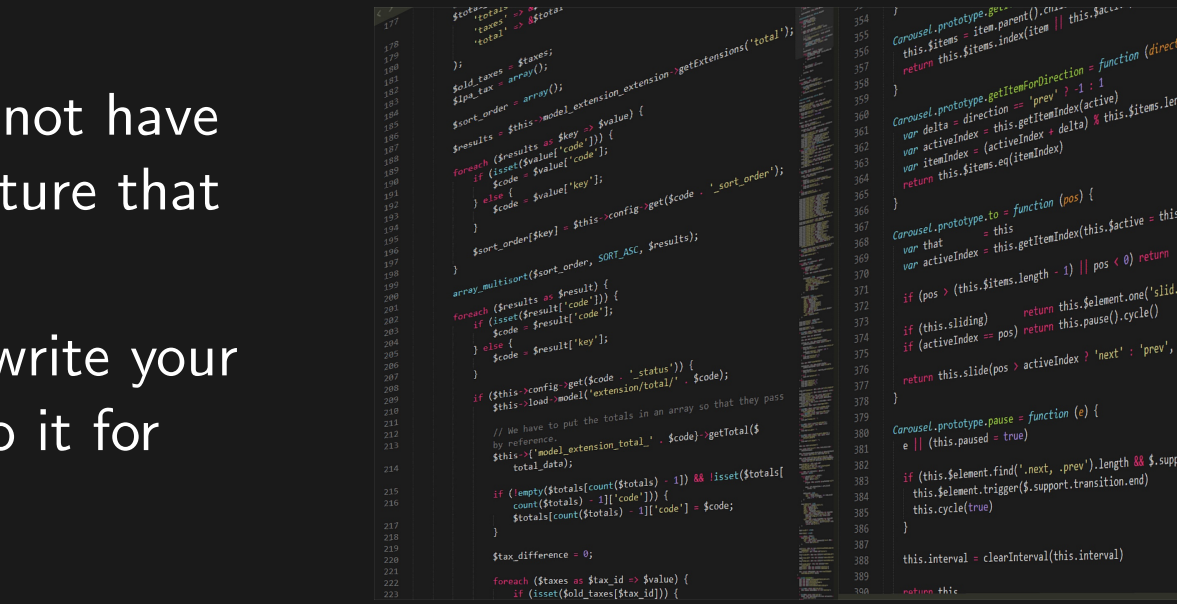
```r
#setting seed
set.seed(43110)

#creating data
norm.data<-rnorm(n=1000, 0, 1)

#creating plot
hist(norm.data, main="1,000 N(0,1)", xlab="x"
    ,col="blue", border="cyan")
```

Custom Functions: What are they?

- Sometimes R does not have an operation of feature that you want
- One solution is to write your own **function** to do it for you



Custom Functions: How to write a function?

```
1 #function format
2 name<-function(argument1, argument2){
3
4   do things with arguments here
5
6 }
```

Custom Functions: Add 2 Numbers

```
1 #function to add two numbers
2 add<-function(x, y){
3
4 temp<-x+y
5 return(temp)
6
7 }
```

Custom Functions: Simulate Rolling a 6 Sided Die

```
1  #function to add two numbers
2  diceroll6<-function(nroll){
3
4    temp<-c()
5     temp<-sample(1:6, nroll, replace=TRUE)
6     print(temp)
7
8  }
9
10 #using the function
11 diceroll(2)
12
13 diceroll(5)+4
14
15 set.seed(567)
16 diceroll(2)
17
18 set.seed(567)
19 diceroll(2)
```

Custom Functions: Write Your Own Function

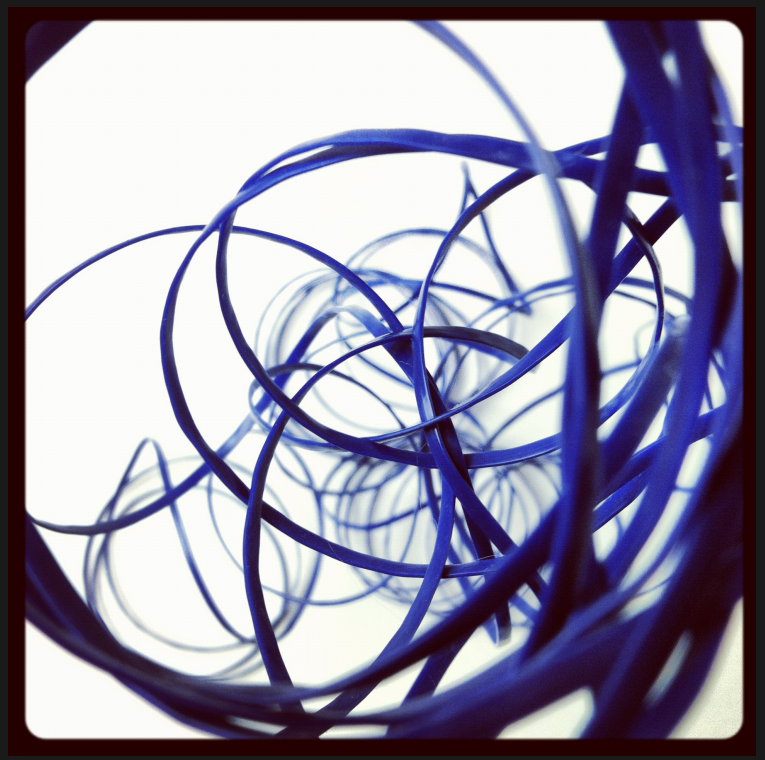
- Write your own function that can roll a m sided die n times
- You have 10 minutes

Custom Functions: Simulate Rolling a m Sided Die

```
1  #function to add two numbers
2  diceroll<-function(m, nroll){
3
4    temp<-c()
5     temp<-sample(1:m, nroll, replace=TRUE)
6     print(temp)
7
8  }
9
10 #using the function
11 diceroll(20, 2)
12
13 diceroll(100, 1)
14
15 diceroll(4, 3)
16
17 DICEROLL(2, 1)
```

Loops: What do they do?

- Sometimes you want to perform a calculation many times
- Is possible to "hard code" it for every time you want it done
- Or you can have a loop do it for you
- Different kinds
- for loops
    - do something n times



Loops: for Loop Framework

```
1 #for loop framework
2 for(i in 1:n){
3
4   do something here n times
5
6 }
```

Loops: for Loop Example Part 1

```
1  #for loop example
2
3  #setting seed
4  set.seed(43112)
5
6  #creating data
7  x1<-rnorm(100, 0, 1)
8  x2<-rnorm(100, 1, 2)
9  x3<-rnorm(100, 2, 3)
10
11 #combining into one matrix
12 X<-cbind(x1, x2, x3)
13 head(X)
14
15 #array to hold means
16 temp<-c()
```

Loops: for Loop Example Part 2

```
1
2 #for loop section
3 for(i in 1:3){
4
5     #calculate the mean for each column
6   temp[i]<-mean(X[,i])
7
8 }
```

Loops: One Last Example

- Create a 10 by 10 matrix with random Poisson observations with $\lambda = 4$
- calculate the variance of each sample using for loops
- set the seed to 813
- You have 10 minutes

Loops: Poisson for Loop Part 1

```
1  #creating matrix
2
3  p<-matrix(nrow=10, ncol=10)
4
5  for(i in 1:10){
6
7    for(j in 1:10){
8
9      p[j,i]<-rpois(n=1, lambda=4)
10
11     }
12
13 }
```

Loops: Poisson for Loop Part 2

```
1 #calculating variances
2
3 vars<-c()
4
5 for(i in 1:10){
6
7 vars[i]<-var(p[,i])
8
9 }
```

Any Questions?