Introduction and Basics of R

Data Analysis with R and Python

Deepayan Sarkar

Review: Software for Statistics

- Computation is an essential part of modern statistics
  - Handling large datasets
  - Visualization

Review: Software for Statistics

- Computation is an essential part of modern statistics
    - Handling large datasets
    - Visualization
    - Simulation
    - Iterative methods

Review: Software for Statistics

- Computation is an essential part of modern statistics
  - Handling large datasets
  - Visualization
  - Simulation
  - Iterative methods
- Many options, but we will focus on R and Python
  - Available as Free / Open Source Software
  - Easy to try out on your own
  - Contains all essential data analysis tools
  - Active community

Interacting with R

- R is most commonly used as a REPL (Read-Eval-Print-Loop)
  - When it is started, R Waits for user input
  - User inputs expression
  - R evaluates and prints result
  - Waits for more input

Interacting with R

- R is most commonly used as a REPL (Read-Eval-Print-Loop)
    - When it is started, R Waits for user input
    - User inputs expression
    - R evaluates and prints result
    - Waits for more input
- Python supports exactly the same model

Interacting with R

- R is most commonly used as a REPL (Read-Eval-Print-Loop)
  - When it is started, R Waits for user input
  - User inputs expression
  - R evaluates and prints result
  - Waits for more input
- Python supports exactly the same model
- Several *interfaces* are available to help in this process
- Recommended options (representing slightly different approaches)
  - RStudio
  - Jupyter

Running R

- On starting up, R shows you something like this:

```
R version 4.2.2 Patched (2022-11-17 r83375) -- "Innocent and Trusting"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

- The `>` represents a *prompt* indicating that R is waiting for input

Running Python

- Python has a similar but much shorter start-up message:

```
Python 3.9.12 (main, Apr  5 2022, 01:53:17)
[Clang 12.0.0 ] :: Anaconda, Inc. on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- The `>>>` similarly represents the *prompt* indicating that Python is waiting

The R REPL essentially works like a calculator

```r
34 * 23 + 10
```

```
[1] 792
```

```r
27 + 1 / 7
```

```
[1] 27.14286
```

```r
2^10
```

```
[1] 1024
```

```r
exp(2)
```

```
[1] 7.389056
```

The Python REPL is very similar

```python
34 * 23 + 10
```

```
792
```

```python
27 + 1 / 7
```

```
27.142857142857142
```

But not exactly the same

```
2^10
```

```
8
```

But not exactly the same

```
2^10
```

```
8
```

```
2 ** 10
```

```
1024
```

But not exactly the same

```
exp(2)
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'exp' is not defined
```

NumPy for scientific computing in Python

- Python with NumPy

```python
from numpy import *
exp(2)
```

```
np.float64(7.38905609893065)
```

```python
sin(pi / 6)
```

```
np.float64(0.49999999999999994)
```

- Compare with R

```r
exp(2)
```

```
[1] 7.389056
```

```r
sin(pi / 6)
```

```
[1] 0.5
```

Why learn both R and Python?

- Both are just tools… skills are more important

Why learn both R and Python?

- Both are just tools… skills are more important
- Complementary strengths

Why learn both R and Python?

- Both are just tools… skills are more important
- Complementary strengths
- Easy interoperability

What about error messages?

```r
34 +* 23
```

```
Error: unexpected '*' in "34 +*"
```

```r
27 // 7
```

```
Error: unexpected '/' in "27 //"
```

```r
arcsin(1/2)
```

```
Error in arcsin(1/2): could not find function "arcsin"
```

Essential features of R

R supports mathematical functions

```r
sqrt(5 * 125)
```

```
[1] 25
```

```r
log(120)
```

```
[1] 4.787492
```

```r
factorial(10)
```

```
[1] 3628800
```

```r
log(factorial(10))
```

```
[1] 15.10441
```

R supports mathematical functions

```
choose(15, 5)
```

```
[1] 3003
```

```
factorial(15) / (factorial(10) * factorial(5))
```

```
[1] 3003
```

R supports mathematical functions

```r
choose(15, 5)
```

```
[1] 3003
```

```r
factorial(15) / (factorial(10) * factorial(5))
```

```
[1] 3003
```

```r
choose(1500, 2)
```

```
[1] 1124250
```

```r
factorial(1500) / (factorial(1498) * factorial(2))
```

```
[1] NaN
```

R supports variables

```r
x <- 2
y <- 10
x^y
```

```
[1] 1024
```

```r
y^x
```

```
[1] 100
```

```r
factorial(y)
```

```
[1] 3628800
```

```r
log(factorial(y), base = x)
```

```
[1] 21.79106
```

R can compute on vectors

```
N <- 15
x <- seq(0, N)
N
```

```
[1] 15
```

```
x
```

```
 [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

```
1:N
```

```
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15
```

```
choose(N, x)
```

```
 [1]    1   15  105  455 1365 3003 5005 6435 6435 5005 3003 1365  455  105   15    1
```

Example: Evaluating Binomial probabilities

- Binomial distribution:

$$P(X = x) = \binom{n}{x} p^x (1 - p)^{n-x}$$

Example: Evaluating Binomial probabilities

- Binomial distribution:

$$P(X = x) = \binom{n}{x} p^x (1-p)^{n-x}$$

```
N <- 15
x <- seq(0, N)
p <- 0.25
choose(N, x) * p^x * (1-p)^(N-x)
```

```
 [1] 1.336346e-02 6.681731e-02 1.559070e-01 2.251991e-01 2.251991e-01 1.651460e-01
 [7] 9.174777e-02 3.932047e-02 1.310682e-02 3.398065e-03 6.796131e-04 1.029717e-04
[13] 1.144130e-05 8.800998e-07 4.190952e-08 9.313226e-10
```

Summary functions that work on vectors

```r
p.x <- dbinom(x, size = N, prob = p)
p.x
```

```
 [1] 1.336346e-02 6.681731e-02 1.559070e-01 2.251991e-01 2.251991e-01 1.651460e-01
 [7] 9.174777e-02 3.932047e-02 1.310682e-02 3.398065e-03 6.796131e-04 1.029717e-04
[13] 1.144130e-05 8.800998e-07 4.190952e-08 9.313226e-10
```

```r
x * p.x
```

```
 [1] 0.000000e+00 6.681731e-02 3.118141e-01 6.755972e-01 9.007963e-01 8.257299e-01
 [7] 5.504866e-01 2.752433e-01 1.048546e-01 3.058259e-02 6.796131e-03 1.132688e-03
[13] 1.372956e-04 1.144130e-05 5.867332e-07 1.396984e-08
```

```r
sum(x * p.x) / sum(p.x)
```

```
[1] 3.75
```

```r
N * p
```

```
[1] 3.75
```

Example: Factorials revisited

```r
choose(1500, 2)
```

```
[1] 1124250
```

```r
log(choose(1500, 2))
```

```
[1] 13.93263
```

```r
factorial(1500) / (factorial(1498) * factorial(2)) # fails
```

```
[1] NaN
```

Example: Factorials revisited

```r
choose(1500, 2)
```

```
[1] 1124250
```

```r
log(choose(1500, 2))
```

```
[1] 13.93263
```

```r
factorial(1500) / (factorial(1498) * factorial(2)) # fails
```

```
[1] NaN
```

```r
factorial(1500)
```

```
[1] Inf
```

Factorials using vectorized arithmetic

```r
factorial(10)
```

```
[1] 3628800
```

```r
prod(seq(1, 10))
```

```
[1] 3628800
```

Factorials using vectorized arithmetic

```r
factorial(10)
```

```
[1] 3628800
```

```r
prod(seq(1, 10))
```

```
[1] 3628800
```

```r
log(prod(seq(1, 10)))
```

```
[1] 15.10441
```

```r
sum(log(1:10))
```

```
[1] 15.10441
```

Factorials using vectorized arithmetic

```
factorial(1500)
```

```
[1] Inf
```

```
prod(seq(1, 1500))
```

```
[1] Inf
```

```
log(prod(seq(1, 1500)))
```

```
[1] Inf
```

```
sum(log(1:1500))
```

```
[1] 9474.406
```

Factorials using vectorized arithmetic

```
sum(log(1:1500)) - sum(log(1:1498)) - sum(log(1:2))
```

```
[1] 13.93263
```

```
exp(sum(log(1:1500)) - sum(log(1:1498)) - sum(log(1:2)))
```

```
[1] 1124250
```

Factorials using vectorized arithmetic

```
sum(log(1:1500)) - sum(log(1:1498)) - sum(log(1:2))
```

```
[1] 13.93263
```

```
exp(sum(log(1:1500)) - sum(log(1:1498)) - sum(log(1:2)))
```

```
[1] 1124250
```

- Compare with

```
log(choose(1500, 2))
```

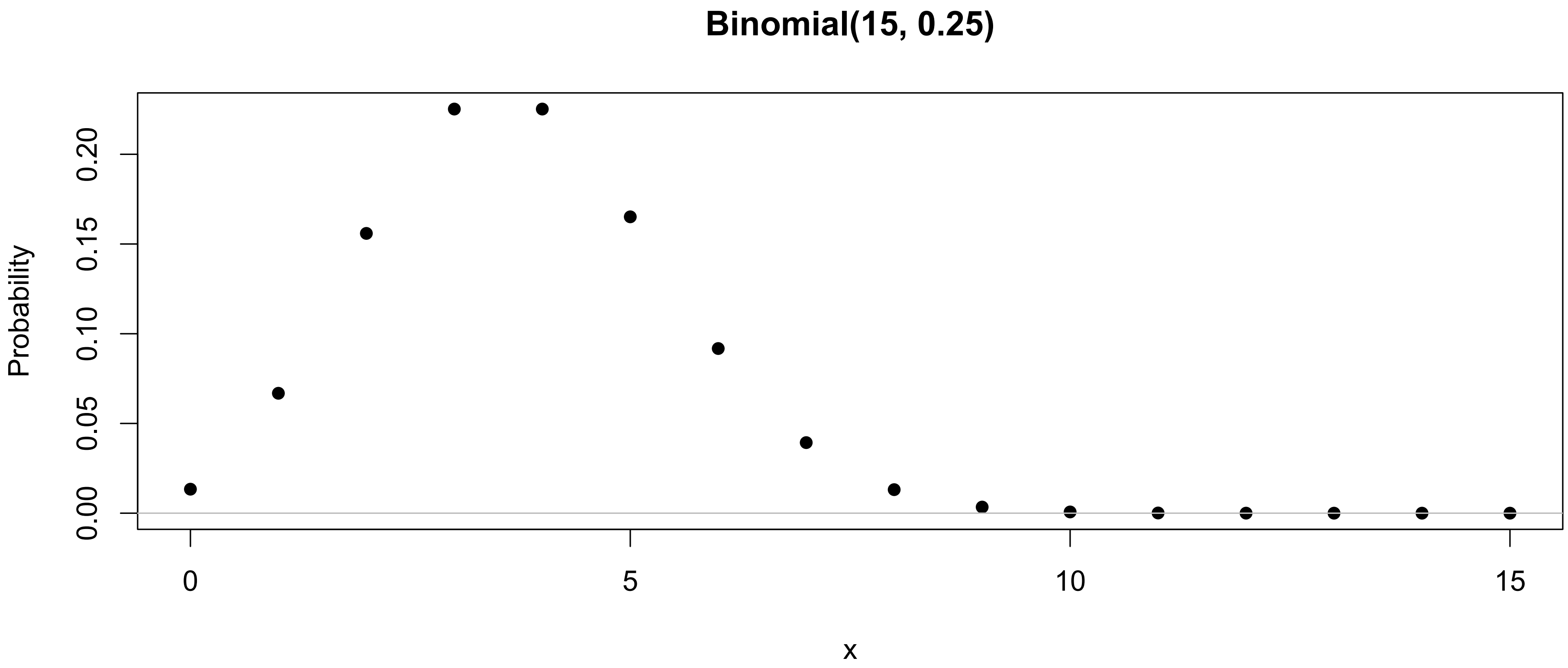
```
[1] 13.93263
```

```
choose(1500, 2)
```

```
[1] 1124250
```

R can draw graphs

```
plot(x, p.x, ylab = "Probability", pch = 16)
title(main = sprintf("Binomial(%g, %g)", N, p))
abline(h = 0, col = "grey")
```



R can simulate random variables

```
cards <- as.vector(outer(c("H", "D", "C", "S"), 1:13, paste, sep = "-"))
cards
```

```
 [1] "H-1"  "D-1"  "C-1"  "S-1"  "H-2"  "D-2"  "C-2"  "S-2"  "H-3"  "D-3"  "C-3"  "S-3" 
[13] "H-4"  "D-4"  "C-4"  "S-4"  "H-5"  "D-5"  "C-5"  "S-5"  "H-6"  "D-6"  "C-6"  "S-6" 
[25] "H-7"  "D-7"  "C-7"  "S-7"  "H-8"  "D-8"  "C-8"  "S-8"  "H-9"  "D-9"  "C-9"  "S-9" 
[37] "H-10" "D-10" "C-10" "S-10" "H-11" "D-11" "C-11" "S-11" "H-12" "D-12" "C-12" "S-12"
[49] "H-13" "D-13" "C-13" "S-13"
```

R can simulate random variables

```r
cards <- as.vector(outer(c("H", "D", "C", "S"), 1:13, paste, sep = "-"))
cards
```

```
 [1] "H-1"  "D-1"  "C-1"  "S-1"  "H-2"  "D-2"  "C-2"  "S-2"  "H-3"  "D-3"  "C-3"  "S-3" 
[13] "H-4"  "D-4"  "C-4"  "S-4"  "H-5"  "D-5"  "C-5"  "S-5"  "H-6"  "D-6"  "C-6"  "S-6" 
[25] "H-7"  "D-7"  "C-7"  "S-7"  "H-8"  "D-8"  "C-8"  "S-8"  "H-9"  "D-9"  "C-9"  "S-9" 
[37] "H-10" "D-10" "C-10" "S-10" "H-11" "D-11" "C-11" "S-11" "H-12" "D-12" "C-12" "S-12"
[49] "H-13" "D-13" "C-13" "S-13"
```

```r
sample(cards, 13)
```

```
 [1] "H-12" "C-2"  "C-1"  "C-12" "C-13" "H-4"  "S-3"  "C-6"  "S-8"  "D-6"  "H-9"  "H-1" 
[13] "S-9" 
```

```r
sample(cards, 13)
```

```
 [1] "C-6"  "S-9"  "S-6"  "H-8"  "H-7"  "D-8"  "D-7"  "H-6"  "D-1"  "D-13" "C-12" "D-4" 
[13] "H-2" 
```

Demonstration: The law of large numbers

- Does proportion of tosses that turn up head converge to probability?

Demonstration: The law of large numbers

- Does proportion of tosses that turn up head converge to probability?

```
p <- 0.25
z <- rbinom(100, size = 1, prob = p)
z
```

```
  [1] 0 0 0 1 1 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0
 [43] 0 0 0 1 0 0 0 0 1 1 1 0 1 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0
 [85] 0 0 1 1 0 0 0 1 1 0 0 0 1 0 1 0
```

Demonstration: The law of large numbers

- Does proportion of tosses that turn up head converge to probability?

```
p <- 0.25
z <- rbinom(100, size = 1, prob = p)
z
```

```
 [1] 0 0 0 1 1 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 0 0 0
[43] 0 0 0 1 0 0 0 0 1 1 1 0 1 0 0 1 0 1 0 0 0 0 1 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0
[85] 0 0 1 1 0 0 0 1 1 0 0 0 1 0 1 0
```

```
cumsum(z)
```

```
 [1]  0  0  0  1  2  2  3  3  3  3  3  3  4  4  4  4  5  5  5  5  5  5  5  5  5  5  5  5
[29]  5  5  5  6  6  7  7  7  7  8  8  8  8  8  8  8  8  9  9  9  9  9 10 11 12 12 13 13
[57] 13 14 14 15 15 15 15 15 16 16 17 17 17 17 17 17 17 18 18 18 19 19 19 19 19 19 19 19
[85] 19 19 20 21 21 21 21 22 23 23 23 23 24 24 25 25
```

Demonstration: The law of large numbers

```r
cumsum(z) / 1:100
```

```
  [1] 0.0000000 0.0000000 0.0000000 0.2500000 0.4000000 0.3333333 0.4285714 0.3750000
  [9] 0.3333333 0.3000000 0.2727273 0.2500000 0.3076923 0.2857143 0.2666667 0.2500000
 [17] 0.2941176 0.2777778 0.2631579 0.2500000 0.2380952 0.2272727 0.2173913 0.2083333
 [25] 0.2000000 0.1923077 0.1851852 0.1785714 0.1724138 0.1666667 0.1612903 0.1875000
 [33] 0.1818182 0.2058824 0.2000000 0.1944444 0.1891892 0.2105263 0.2051282 0.2000000
 [41] 0.1951220 0.1904762 0.1860465 0.1818182 0.1777778 0.1956522 0.1914894 0.1875000
 [49] 0.1836735 0.1800000 0.1960784 0.2115385 0.2264151 0.2222222 0.2363636 0.2321429
 [57] 0.2280702 0.2413793 0.2372881 0.2500000 0.2459016 0.2419355 0.2380952 0.2343750
 [65] 0.2461538 0.2424242 0.2537313 0.2500000 0.2463768 0.2428571 0.2394366 0.2361111
 [73] 0.2328767 0.2432432 0.2400000 0.2368421 0.2467532 0.2435897 0.2405063 0.2375000
 [81] 0.2345679 0.2317073 0.2289157 0.2261905 0.2235294 0.2209302 0.2298851 0.2386364
 [89] 0.2359551 0.2333333 0.2307692 0.2391304 0.2473118 0.2446809 0.2421053 0.2395833
 [97] 0.2474227 0.2448980 0.2525253 0.2500000
```

Demonstration: The law of large numbers

- How *fast* does the sample proportion converge?

```
plot(1:100, type = "n", ylim = c(0, 1), ylab = "Sample proportion")
for (i in 1:50) {
    z <- rbinom(100, size = 1, prob = p)
    lines(1:100, cumsum(z) / 1:100, col = sample(colors(), 1))
}
```