# Lab 1

*Sakib Salim*

This lab is due 11:59 PM Satuday 2/9/19.

You should have RStudio installed to edit this file. You will write code in places marked "TO-DO" to complete the problems. Some of this will be a pure programming assignment. The tools for the solutions to these problems can be found in the class practice lectures. I want you to use the methods I taught you, not for you to google and come up with whatever works. You won't learn that way.

To "hand in" the homework, you should compile or publish this file into a PDF that includes output of your code. Once it's done, push by the deadline to your repository in a directory called "labs".

- Print out the numerical constant pi with ten digits after the decimal point using the internal constant `pi`.

```
#TO-DO
pi
```

```
## [1] 3.141593
```

```
options(digits=11)
```

- Sum up the first 100 terms of the series 1 + 1/2 + 1/4 + 1/8 + ...

```
#TO-DO
sum((1/2)^(0:99))
```

```
## [1] 2
```

- Find the product of the first 100 terms of `1 * 1/2 * 1/4 * 1/8 * ...`

```
#TO-DO
prod((1/2)^(0:99))
```

```
## [1] 0
```

- Find the product of the first 500 terms of `1 * 1/2 * 1/4 * 1/8 * ...` Answer in English: is this answer correct?

```
#TO-DO
prod((1/2)^(0:499))
```

```
## [1] 0
```

```
print ("No, this answer is not correct. It must be nonzero.")
```

```
## [1] "No, this answer is not correct. It must be nonzero."
```

- Figure out a means to express the answer more exactly. Not compute exactly, but express more exactly.

```
#TO-DO
print("(1/2)^(sum(0:499))")
```

```
## [1] "(1/2)^(sum(0:499))"
```

```
print("The powers add for multiplication of the same base, so that the final product is the sum of all
```

```
## [1] "The powers add for multiplication of the same base, so that the final product is the sum of all
```

- Use the left rectangle method to numerically integrate x^2 from 0 to 1 with rectangle size 1e-6.

```
#TO-DO
((1e-6)*sum(seq(0,1,by=1e-6)^2))
```

```
## [1] 0.33333383333
```

```
#delta-x (1e^-6) is constant so it comes out of the Riemann sum
```

- Calculate the average of 100 realizations of standard Bernoullis in one line using the `sample` function.

```
#TO-DO
sum(sample(0:1, 100, replace = TRUE, c(0.5,0.5)))/100
```

```
## [1] 0.55
```

- Calculate the average of 500 realizations of Bernoullis with p = 0.9 in one line using the `sample` function.

```
#TO-DO
sum(sample(0:1, 500, replace = TRUE, c(0.1,0.9)))/500
```

```
## [1] 0.904
```

- Calculate the average of 1000 realizations of Bernoullis with p = 0.9 in one line using `rbinom`.

```
#TO-DO
sum(rbinom(1000, 1, p = 0.9))/1000
```

```
## [1] 0.871
```

- Use the `strsplit` function and `sample` to put the sentences below in random order.

```
lorem = "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi posuere varius volutpat. Morbi
#TO-DO
paste(paste(sample((strsplit(lorem, "[.] "))[[1]][1:10]), collapse= ". "), ". ", sep = "")
```

```
## [1] "Integer dapibus mi lectus, eu posuere arcu ultricies in. Morbi faucibus ligula id massa ultricie
```

```
# split by periods, access list element and sample from the 10 sentence vectors and then paste sentence
```

- In class we generated the variable criminality with levels "none", "infraction", "misdimeanor" and "felony". Create a variable `x_2` here with 100 random elements (equally probable) and ensure the proper ordinal ordering.

```
#TO-DO
Crim = c("none", "infraction", "misdemeanor", "felony")
CrimFactor = factor(Crim, ordered = T, levels = c("none", "infraction", "misdemeanor", "felony"))
x_2 = sample(CrimFactor, 100, replace = T)
x_2
```

```
##   [1] misdemeanor infraction  misdemeanor felony      felony
##   [6] infraction  misdemeanor none        misdemeanor misdemeanor
##  [11] misdemeanor felony      felony      none        felony
##  [16] misdemeanor none        none        felony      misdemeanor
##  [21] infraction  infraction  infraction  none        none
##  [26] felony      infraction  misdemeanor felony      misdemeanor
##  [31] misdemeanor infraction  none        infraction  felony
##  [36] infraction  felony      none        felony      felony
##  [41] misdemeanor felony      none        infraction  infraction
##  [46] felony      felony      infraction  infraction  felony
##  [51] none        misdemeanor infraction  felony      none
##  [56] misdemeanor felony      felony      misdemeanor infraction
```

```
##  [61] misdemeanor infraction  none        none        infraction
##  [66] none        infraction  felony      misdemeanor felony
##  [71] misdemeanor felony      felony      none        misdemeanor
##  [76] none        felony      infraction  felony      misdemeanor
##  [81] none        infraction  felony      felony      misdemeanor
##  [86] misdemeanor infraction  misdemeanor infraction  none
##  [91] felony      misdemeanor felony      none        none
##  [96] none        misdemeanor infraction  felony      felony
## Levels: none < infraction < misdemeanor < felony
```

- Convert this variable to binary where 0 is no crime and 1 is any crime. Answer in English: is this the proper binary threshold?

```
#TO-DO
x_2 = sample(factor(sample(c("0", "1"), 100, replace = T), ordered = T))
x_2
```

```
##   [1] 1 0 1 0 1 1 1 0 1 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1
##  [36] 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 1 1 1 1 1 1 0 1 0 1 1 1 0 0 0 1 0 0 1 0
##  [71] 0 0 1 0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 1 1 1 0 0 1 0 0 1 0 1 1
## Levels: 0 < 1
```

```
print("This is probably not the best threshold, as we would like more details as to the severity of the
```

```
## [1] "This is probably not the best threshold, as we would like more details as to the severity of th
```

- Convert this variable to an unordered, nominal factor variable.

```
#TO-DO
x_2 = factor(as.numeric(levels(x_2)[x_2]) )
x_2
```

```
##   [1] 1 0 1 0 1 1 1 0 1 0 0 1 0 1 0 0 1 0 0 1 0 1 0 0 1 1 1 0 1 1 0 0 1 1 1
##  [36] 0 0 0 0 0 0 0 0 1 1 0 1 0 0 0 1 1 1 1 1 1 0 1 0 1 1 1 0 0 0 1 0 0 1 0
##  [71] 0 0 1 0 1 0 0 1 0 0 0 1 0 0 0 1 0 0 1 1 1 0 0 1 0 0 1 0 1 1
## Levels: 0 1
```

- Convert this variable into three binary variables without any information loss and put them into a data matrix.

```
#TO-DO
x_3 = sample(factor(as.numeric(levels(x_2)[x_2])))
x_4 = sample(factor(as.numeric(levels(x_2)[x_2])))
newmat = matrix(c(as.numeric(levels(x_2)[x_2]), as.numeric(levels(x_3)[x_3]), as.numeric(levels(x_4)[x_
newmat
```

```
##       [,1] [,2] [,3]
##  [1,]    1    0    1
##  [2,]    0    0    1
##  [3,]    1    1    1
##  [4,]    0    0    0
##  [5,]    1    0    0
##  [6,]    1    1    1
##  [7,]    1    1    0
##  [8,]    0    1    0
##  [9,]    1    0    1
## [10,]    0    0    0
## [11,]    0    0    0
## [12,]    1    1    0
```

```
##  [13,]    0    0    0
##  [14,]    1    0    1
##  [15,]    0    0    1
##  [16,]    0    1    1
##  [17,]    1    1    0
##  [18,]    0    1    0
##  [19,]    0    1    0
##  [20,]    1    0    0
##  [21,]    0    0    0
##  [22,]    1    1    0
##  [23,]    0    0    0
##  [24,]    0    1    0
##  [25,]    1    1    1
##  [26,]    1    0    0
##  [27,]    1    0    1
##  [28,]    0    0    0
##  [29,]    1    1    0
##  [30,]    1    1    0
##  [31,]    0    1    1
##  [32,]    0    1    0
##  [33,]    1    0    0
##  [34,]    1    1    0
##  [35,]    1    0    0
##  [36,]    0    1    0
##  [37,]    0    1    0
##  [38,]    0    0    0
##  [39,]    0    0    1
##  [40,]    0    0    1
##  [41,]    0    0    0
##  [42,]    0    1    1
##  [43,]    0    0    1
##  [44,]    1    0    1
##  [45,]    1    1    1
##  [46,]    0    1    0
##  [47,]    1    0    0
##  [48,]    0    1    0
##  [49,]    0    1    1
##  [50,]    0    0    1
##  [51,]    1    0    1
##  [52,]    1    0    1
##  [53,]    1    1    1
##  [54,]    1    0    0
##  [55,]    1    1    1
##  [56,]    1    0    1
##  [57,]    0    1    1
##  [58,]    1    0    1
##  [59,]    0    1    1
##  [60,]    1    0    1
##  [61,]    1    1    0
##  [62,]    1    0    1
##  [63,]    0    0    0
##  [64,]    0    0    0
##  [65,]    0    0    1
##  [66,]    1    1    1
```

```
##  [67,]    0    1    1
##  [68,]    0    0    1
##  [69,]    1    1    0
##  [70,]    0    0    0
##  [71,]    0    1    0
##  [72,]    0    0    0
##  [73,]    1    1    0
##  [74,]    0    1    1
##  [75,]    1    0    0
##  [76,]    0    1    0
##  [77,]    0    1    1
##  [78,]    1    0    1
##  [79,]    0    1    0
##  [80,]    0    1    1
##  [81,]    0    1    0
##  [82,]    1    0    0
##  [83,]    0    0    1
##  [84,]    0    0    1
##  [85,]    0    1    1
##  [86,]    1    0    0
##  [87,]    0    1    0
##  [88,]    0    1    0
##  [89,]    1    0    0
##  [90,]    1    0    1
##  [91,]    1    1    0
##  [92,]    0    1    0
##  [93,]    0    0    1
##  [94,]    1    0    0
##  [95,]    0    0    0
##  [96,]    0    0    1
##  [97,]    1    0    1
##  [98,]    0    0    0
##  [99,]    1    1    1
## [100,]    1    0    1
```

- What should the sum of each row be (in English)? Verify that.

```
#TO-DO
#The row sum is the number of crimes committed by the first person in each of the three iterations
sum(newmat[1,1:3])
```

```
## [1] 2
```

- How should the column sum look (in English)? Verify that.

```
#TO-DO
#The column sum is the number of crimes committed among the 100 individuals
sum(newmat[1:100,1])
```

```
## [1] 46
```

- Generate a matrix with 100 rows where the first column is realization from a normal with mean 17 and variance 38, the second column is uniform between -10 and 10, the third column is poisson with mean 6, the fourth column in exponential with lambda of 9, the fifth column is binomial with n = 20 and p = 0.12 and the sixth column is a binary variable with 24% 1's.

```r
#TO-DO
randmat = matrix(c(rnorm(100, 17, 38), runif(100, -10, 10), rpois(100, 6), rexp(100, 9), rbinom(100, 20
randmat
```

```
##                  [,1]            [,2] [,3]             [,4] [,5] [,6]
##   [1,]  21.8261526342 -1.247980799526    7 2.3206012065e-02    5    1
##   [2,] -12.1934699081  1.785278101452    5 1.7858641283e-01    2    0
##   [3,] -17.8088523453 -5.214774338529    9 1.1178000812e-01    0    0
##   [4,]  36.5820399881  9.802806801163    4 6.4535109585e-02    8    0
##   [5,]  57.4783499193  4.360340116546    9 5.6900448783e-01    4    1
##   [6,]  61.2038198961 -8.693957151845    6 3.9601574911e-02    4    1
##   [7,]  42.1273515717  5.983231663704    7 7.8864223003e-02    1    0
##   [8,]  -2.0874232958 -2.066848413087    7 2.9923532262e-02    6    0
##   [9,]   9.1793494451 -8.772478397004    3 4.3266082997e-02    2    0
##  [10,]  -8.9745393394  9.026104928926    6 5.6526938091e-02    3    0
##  [11,]  32.0920988680  4.191527608782    9 6.6344164829e-02    1    0
##  [12,]  -7.8774713519  1.856862879358    7 1.1774510165e-01    3    0
##  [13,]  90.2144668727  1.984188929200    4 1.5756108425e-01    2    0
##  [14,] -19.0087856149  3.315059482120    2 3.2895633806e-02    3    0
##  [15,]  12.3426675289  6.621012156829   11 7.6104391004e-02    4    1
##  [16,]  -2.2742126901  1.363836745732    6 1.0960793836e-01    2    0
##  [17,] -38.3461494420 -6.596863619052    8 3.3036318358e-01    2    0
##  [18,] -16.3369000041 -4.712039749138    9 6.8824952779e-02    3    1
##  [19,]  -1.8062200043  4.409150546417    6 1.2845338870e-01    2    0
##  [20,]  66.4352375745  9.653833694756    4 1.3780415158e-02    4    1
##  [21,] -31.2266115543  6.395671311766    4 2.4507535613e-01    2    1
##  [22,] -10.1225509822 -3.437118474394    6 1.0980588199e-01    1    0
##  [23,] -32.6867974792 -2.989947791211    3 5.5075208346e-02    1    0
##  [24,]  30.2714402394  7.850577868521    6 3.9275455696e-01    0    0
##  [25,] -72.1888946029  4.733671052381    3 7.1198857309e-02    4    1
##  [26,]  37.1265432533  0.852512172423    8 1.3273233910e-01    2    0
##  [27,]   8.7839812362  5.032979897223    4 1.4448782677e-01    4    0
##  [28,]  72.5217858392  9.769176393747    5 1.2870226055e-02    5    0
##  [29,]  20.6903278999 -8.034184956923    5 4.8363786346e-02    2    1
##  [30,]  50.2211058277  0.141234225594    4 1.2747416086e-01    1    0
##  [31,] -55.8909014425  3.745843796059    6 1.7369285847e-02    3    0
##  [32,]  14.4237201890  8.668029131368    9 1.5521921528e-02    3    0
##  [33,]  -2.6386169685 -4.961615051143    6 2.2436598149e-01    4    1
##  [34,]  31.3945001040 -4.851408726536    7 1.0975426166e-01    2    0
##  [35,] -18.0610108906  8.012178237550    4 2.0797693417e-02    1    0
##  [36,] -13.7090896651  2.298343018629    3 1.4458304938e-01    2    0
##  [37,]  12.1254098244  5.127217695117    3 1.4571148446e-01    2    1
##  [38,]  -6.7336901741 -8.721242272295    7 1.4766935238e-01    3    1
##  [39,]  49.6522647782 -5.718388990499   10 1.8211583151e-01    1    0
##  [40,]  78.2179584136 -2.519420082681   10 9.6715374362e-02    0    0
##  [41,]  59.9045861973 -6.914584971964    5 4.2357162426e-02    3    0
##  [42,]   4.9486013750 -3.793702791445   13 3.2572083375e-01    0    0
##  [43,]   8.5927128373 -8.776718368754    8 7.9242929812e-02    1    0
##  [44,]  25.2845504233  2.538404352963    8 1.1192077412e-01    1    0
##  [45,]  64.6488067719 -6.679145134985    7 7.7246998747e-05    4    1
##  [46,]   7.4372862218  5.252996371128    5 6.8048103506e-02    2    0
##  [47,]  23.1985378681 -7.944142785855    6 1.9003919325e-01    3    0
##  [48,] -40.4729566443  0.295791518874    6 1.4731390515e-01    2    0
##  [49,]  19.6414524231  1.417060922831    4 1.5565009745e-01    1    0
```

```
##  [50,]  26.9755681627 -1.582322246395   11 2.3366327535e-03    0    0
##  [51,]  83.0786348491 -7.362349145114    6 8.0428787399e-02    2    1
##  [52,]  30.9891633493 -1.368341003545    5 4.9062683070e-03    2    0
##  [53,] -22.8789114421  6.919279592112    2 5.7007082169e-02    5    0
##  [54,]  -1.0167501624  5.959154730663    4 9.5376571826e-02    2    0
##  [55,]   2.5980396272 -2.031115847640    7 3.0779885603e-01    4    0
##  [56,]  63.1741312002  1.655443715863    4 1.8716717461e-01    1    0
##  [57,]   4.8342049685  9.380128909834    3 3.5818841886e-01    4    0
##  [58,]  81.9363207081  5.780920828693    2 3.9974388459e-03    4    0
##  [59,]  45.8101955347  1.361991157755    6 1.8728160665e-03    2    1
##  [60,] -56.5178810065 -1.976946787909    3 2.3755701974e-01    3    0
##  [61,]  50.5589291163 -7.392777563073    5 9.5317162776e-02    0    1
##  [62,]  -4.0279457341 -9.012085520662    7 7.7106668606e-04    3    0
##  [63,]  37.7516143703 -1.939572519623    8 1.4675893510e-02    3    1
##  [64,]  60.8796683261 -0.209721974097    9 4.1875786576e-02    6    0
##  [65,]  17.6108344320  5.430314852856    6 1.9240077326e-02    2    0
##  [66,]  57.1131066757 -1.490784944035    7 6.8067637376e-02    2    0
##  [67,] -31.8746012955  5.223592608236   10 6.2568274637e-02    3    1
##  [68,]  11.1251604879  5.050537465140    5 3.3122049676e-01    3    1
##  [69,] -47.0401463402  8.069009836763    4 9.3843500613e-03    1    0
##  [70,]  -5.2755969839 -1.308260606602    7 7.1092533103e-02    2    1
##  [71,] -16.7146269856  7.951586875133    6 1.9766643518e-01    4    0
##  [72,] -31.2288226747 -7.537770532072    7 1.9944068096e-02    2    0
##  [73,] -17.0534172786  5.176239944994    6 2.8572400554e-01    4    0
##  [74,]  38.4550593554 -0.796836460941    3 1.2205893360e-01    2    0
##  [75,]  30.6058562551 -7.246791943908    3 2.6207961403e-03    4    0
##  [76,]  34.1647523956  7.441832181066    4 7.9745114259e-02    1    0
##  [77,]  25.6348407846 -4.475764892995    3 2.6483332376e-02    3    0
##  [78,] -57.4316266760 -4.747587894090    3 5.8604939686e-02    2    1
##  [79,] 115.9312134164 -1.379621173255    4 2.1640952483e-02    4    0
##  [80,]   9.8917030397 -0.153731121682    8 4.9817112292e-01    2    0
##  [81,]  13.7112700123 -3.271996006370    8 3.2045412946e-02    3    0
##  [82,]  59.8953456016 -3.715006592683    7 9.3894945246e-02    1    1
##  [83,]  30.3367573167  2.293819123879    4 3.1608824070e-03    0    0
##  [84,]  17.6234305452  6.324618016370   10 3.7684570633e-02    2    0
##  [85,]  61.8565876550  1.936516128480    6 4.1440342392e-02    3    0
##  [86,] -23.7828439308 -5.160917341709    6 4.8141529698e-01    0    1
##  [87,] -33.2811245362  7.764787198976    5 4.6292697245e-02    0    0
##  [88,]  47.1944683161 -4.701524423435    7 3.3720159748e-02    0    0
##  [89,]  46.7373030932 -5.125551451929    8 1.2733576043e-02    5    0
##  [90,]  21.2459688713  7.627199799754    5 3.3011750700e-01    2    0
##  [91,]  -9.0328537251  4.397066636011    8 3.8359651756e-02    1    1
##  [92,]  20.1738826134 -2.853063559160    6 5.8655857315e-02    3    0
##  [93,] -22.0338718844 -2.237076857127    1 3.6973405786e-02    2    0
##  [94,]  45.6056434823 -5.004117339849    5 2.2081795831e-01    3    0
##  [95,]  25.4428451016 -3.730100509711    4 1.2820776883e-01    4    0
##  [96,]  29.7716420197  6.867604348809    9 1.7289764093e-02    1    0
##  [97,] -26.7547985139  8.300267299637    5 1.1723099276e-01    4    0
##  [98,] -12.0219257561  0.143002527766    3 1.3175706622e-02    2    0
##  [99,]  -5.2259905200 -0.081469211727    5 2.2918446175e-01    2    0
## [100,] -37.2580858091 -3.992118048482    5 2.6715153134e-02    4    0
```