SC_MD_Nov-07_1

Srijan Kundu

2022-11-07

Question 1: Suppose we want to generate 20 random observations from a $N(2,1)$ distribution, and compute the mean of those observations.

```
samp1 = rnorm(20, 2, 1)
samp1
```

```
##  [1] 1.0232650 1.4288732 1.8412335 2.4906066 2.6616639 0.2293558 2.8026751
##  [8] 3.1943932 0.6525791 2.0410693 1.7223666 1.0457068 3.1594723 1.0744405
## [15] 2.4274900 4.4616661 0.4134723 1.0827748 2.8623808 2.6879925
```

```
mean(samp1)
```

```
## [1] 1.965174
```

Question 2: Suppose we want to repeat this procedure $R = 10$ times.

Steps:

1. Draw a random sample of size $n = 20$ from $N(2,1)$ distribution.
2. Compute mean of the observations from *step 1*.
3. Repeat *Step 1 and 2* R times.

To perform such repetitive tasks, we can make use of the function `replicate`.

The syntax is: `replicate(n, expr)`; where `n` stands for the number of replications, and `expr` stands for the expression that we have to evaluate multiple times.

```
R1 = 10
samp2 = replicate(R1, mean(rnorm(20, 2, 1)))
samp2
```

```
##  [1] 2.102033 2.060247 1.848025 2.199593 1.859461 1.996673 2.135170
1.945755
##  [9] 1.848818 2.077950
```

Question 3: Suppose we want to compute the mean of these R sample means, and the variance.

```
mean(samp2)
```

```
## [1] 2.007373
```

```
var(samp2)
```

```
## [1] 0.01625983
```

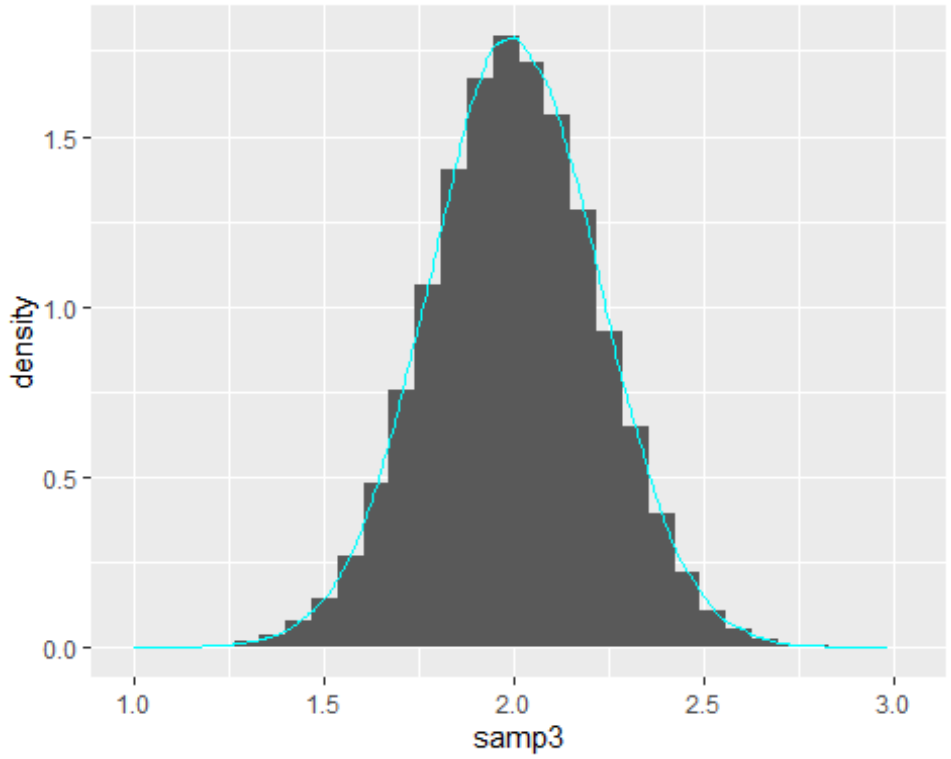
```
(R1-1)*var(samp2)/R1
```

```
## [1] 0.01463385
```

Question 4: Make $R = 100$, similarly as above, find the mean, var and draw the histogram.

```
R2 = 100000
samp3 = replicate(R2, mean(rnorm(20, 2, 1)))
#hist(samp3, freq = FALSE)
ggplot(data = NULL, aes(x = samp3)) + geom_histogram(aes(y=..density..))
+geom_density(colour = "cyan")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Suppose we want to write a program for the same problem using a user defined function.

- The syntax for an user defined function is as below:

  f = function(x){

  x*x*x

  }

- Here, f(x) returns x^3.

```
f1 = function(x){
  x^3
```

```
}
u = 1:4
f1(u)

## [1]  1  8 27 64
```

Note that here `f` is the name of the function given by the user and `x` is the argument, which can be a scalar or a vector. The function must be enclosed within {}. The argument inside the function and outside the function need not be given the same name, but they must be of the same type.

```
f2 = function(x){
  n = 5
  x^3+n
}
u = 1:4
f2(u)

## [1]  6 13 32 69

#n
```

Remember that all the constants must be defined globally (outisde the function) so that they can be used for other purposes outside the function.

```
n = 5
f3 = function(x){
  x^3+n
}
u = 1:4
f3(u)

## [1]  6 13 32 69

n

## [1] 5
```

Question 5: Generate a random sample of size $n = 10,50,100,200,500$ from a $N(2,1)$ dist. and compute its mean. Repeat this procedure 1000 times, compute the mean and the variance of the 1000 samples means.

```
R3 = 1000
f4 = function(n){
  mean(rnorm(n, 2, 1))
}
n = c(10, 50, 100, 500)
v1 = c(mean(replicate(R3, f4(10))), mean(replicate(R3, f4(50))),
mean(replicate(R3, f4(100))), mean(replicate(R3, f4(500))))
v2 = c(var(replicate(R3, f4(10))), var(replicate(R3, f4(50))),
var(replicate(R3, f4(100))), var(replicate(R3, f4(500))))
data.frame(Sample_Size = n, Means = v1, Variances = v2)
```

```
##   Sample_Size    Means   Variances
## 1          10 2.003952 0.097248727
## 2          50 1.999250 0.020508799
## 3         100 2.001949 0.009571445
## 4         500 2.001316 0.001965303
```

or,

```
f5 = function(n, R, mu, sigma){
  g_m = g_v = array(0)
  for(i in seq_along(n))
  {
    x = replicate(R, mean(rnorm(n[i], mu, sigma)))
    g_m[i] = mean(x)
    g_v[i] = var(x)*(R-1)/R
  }
  return(data.frame(sample_size = n, Grand_means = g_m, Grand_variances =
g_v))
}
f5(c(10, 50, 100, 500), 1000, 2, 1)
```

```
##   sample_size Grand_means Grand_variances
## 1          10    2.022157     0.101837420
## 2          50    1.998996     0.018985630
## 3         100    2.004041     0.010064166
## 4         500    1.996989     0.001867185
```

or,

```
k = 4
R = 1000
L = function(n){
  u = replicate(R, mean(rnorm(n, 2, 1)))
  m = mean(u)
  v = var(u)*(R-1)/R
  answer = c(n, m, v)
  answer
}
M = matrix(0, k, 3)
n = c(10, 50, 100, 200)
for(i in 1:k)
{
  M[i,] = L(n[i])
}
colnames(M) = c("Sample Size", "Mean", "Variance")
M
```

```
##      Sample Size     Mean    Variance
## [1,]          10 2.003850 0.095960935
## [2,]          50 1.998589 0.019560498
```

```
## [3,]         100 1.995418 0.010778164
## [4,]         200 1.997378 0.004987434
```

```
x = rnorm(100)
y = ifelse(x>0, 1, 0)
y
```

```
##   [1] 1 1 0 1 0 1 1 1 1 0 0 0 0 0 1 1 1 1 1 1 0 0 0 1 0 1 1 1 1 1 0 0 1 0
0 1 1
##  [38] 0 0 1 1 1 0 0 1 1 0 0 0 1 0 1 0 1 1 1 1 1 0 0 1 0 1 1 0 0 1 0 0 1 0
1 0 0
##  [75] 0 0 0 1 1 0 0 1 1 0 0 1 0 1 0 1 0 0 1 0 0 0 0 1 0 0
```

```
x = rnorm(100)
y = ifelse(x<0, 1, ifelse(x>1 & x < 1.5, 2, ifelse(x > 1.5, 3, 4)))
data.frame(x, y)
```

```
##                x y
## 1    0.758437412 4
## 2    0.304409313 4
## 3    0.782619034 4
## 4    1.490472671 2
## 5   -0.547027613 1
## 6   -0.791646418 1
## 7    0.693983990 4
## 8    0.755893707 4
## 9   -0.132964691 1
## 10  -0.398604001 1
## 11  -1.754347054 1
## 12   0.875808230 4
## 13  -0.147758059 1
## 14   0.671748224 4
## 15   1.878728800 3
## 16   0.484576607 4
## 17   1.212264177 2
## 18  -1.029166280 1
## 19  -0.003909573 1
## 20   0.544801440 4
## 21   1.478281280 2
## 22   0.781721155 4
## 23  -1.016317957 1
## 24   0.512595682 4
## 25  -0.759743313 1
## 26   0.345022477 4
## 27   0.380527308 4
## 28  -0.226254954 1
## 29  -1.252536743 1
## 30  -0.102447220 1
## 31   1.234754003 2
## 32   1.601930639 3
## 33  -1.534862679 1
```

```
## 34   0.900411281 4
## 35  -0.867630692 1
## 36  -0.142205040 1
## 37  -0.421786504 1
## 38   0.482464143 4
## 39   0.485395418 4
## 40  -0.069868926 1
## 41  -0.379450584 1
## 42   0.084727843 4
## 43   0.585445335 4
## 44   0.840297041 4
## 45  -0.632620309 1
## 46  -0.917284353 1
## 47   0.236950935 4
## 48   1.420387668 2
## 49  -0.357000279 1
## 50  -0.141090730 1
## 51   0.882295781 4
## 52   2.279913933 3
## 53   0.370931149 4
## 54  -0.210180961 1
## 55   0.167237984 4
## 56  -0.467296790 1
## 57   0.688585499 4
## 58  -0.259921487 1
## 59   0.091144453 4
## 60   0.844352360 4
## 61   0.832125835 4
## 62   1.416536775 2
## 63  -1.027206447 1
## 64   0.919729422 4
## 65   1.386728215 2
## 66   0.506278360 4
## 67  -0.231917804 1
## 68   0.036517965 4
## 69  -0.885957020 1
## 70   1.620487659 3
## 71  -2.020120576 1
## 72  -0.426833580 1
## 73  -0.544983076 1
## 74   0.402019129 4
## 75  -1.826805560 1
## 76  -0.947835267 1
## 77   0.198793598 4
## 78  -0.900402044 1
## 79  -0.764853575 1
## 80   1.472774010 2
## 81  -1.451189570 1
## 82   0.381018629 4
## 83   0.324062546 4
```

```
## 84  -0.607417941 1
## 85   0.940269672 4
## 86   1.994368777 3
## 87   0.408408116 4
## 88  -0.745107687 1
## 89   1.579404083 3
## 90  -1.047884143 1
## 91   0.072534410 4
## 92  -0.176415616 1
## 93  -1.279761678 1
## 94  -0.462383070 1
## 95   1.570488365 3
## 96  -0.909589716 1
## 97  -0.500900909 1
## 98  -1.003759081 1
## 99  -0.534975788 1
## 100  0.856475204 4
```