MG 221: Assignment I 2020

Due Date: **December 10, 2020**

The purpose of this assignment is to understand the concept of Sampling Distributions and learn the Central Limit Theorem (CLT) hands on. Consider the following R function:

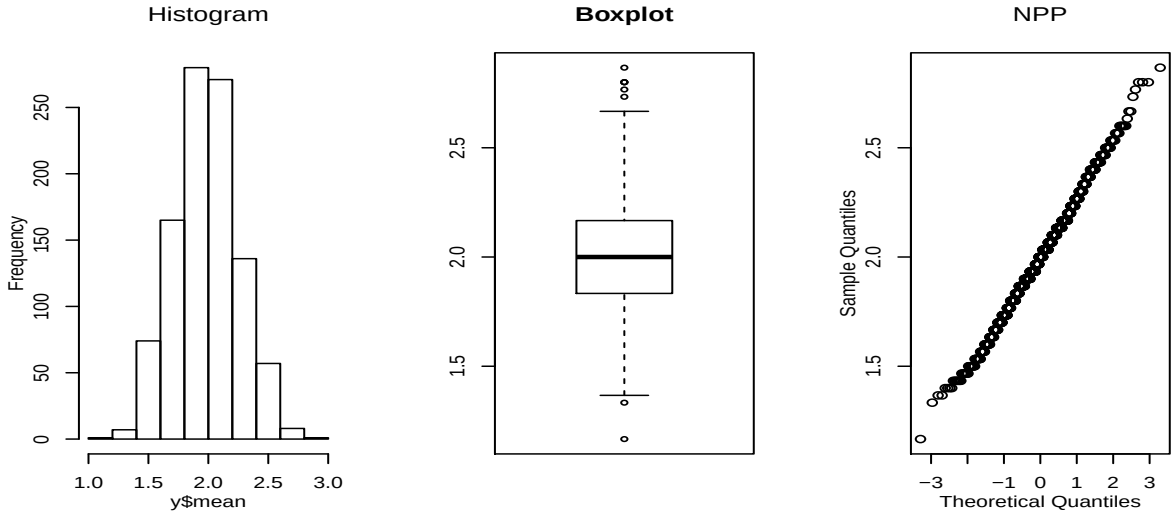
```
simpois<-function(n,nn,lambda)
{
 mn<-NULL
 mdn<-NULL
 sdn<-NULL
 min<-NULL
 max<-NULL
 iqr<-NULL
 for(i in 1:nn)
 {
  x<-rpois(n,lambda)
  mn[i]<-mean(x)
  mdn[i]<-median(x)
  sdn[i]<-sd(x)
  min[i]<-range(x)[1]
  max[i]<-range(x)[2]
  iqr[i]<-IQR(x)
 }
 return(list(mean=mn,median=mdn,sd=sdn,minm=min,maxm=max,iqr=iqr))
}
```

There are several ways to actually define this function in R. The most obvious way to enter this is to type the above code line by line in the R-prompt by hitting "Enter" after each line, during which the R-prompt changes to a "+" from its usual ">". This approach has the obvious draw-back of not being able to correct the typos, if any, in an already entered line. Other approaches include typing the code as plain text using some text-editor and then copying and pasting it in the R-workspace, or saving it in a file named something like `sim.r` and then calling it in R by typing `source(‘‘sim.r’’)` in the R-prompt. An existing or already defined function like `simpois` in the R-workspace can also be edited by issuing a R-command like `simpois<-vi(simpois)` in the R-prompt in a Unix environment, or by invoking some other text-editor in place of `vi` in other operating systems.

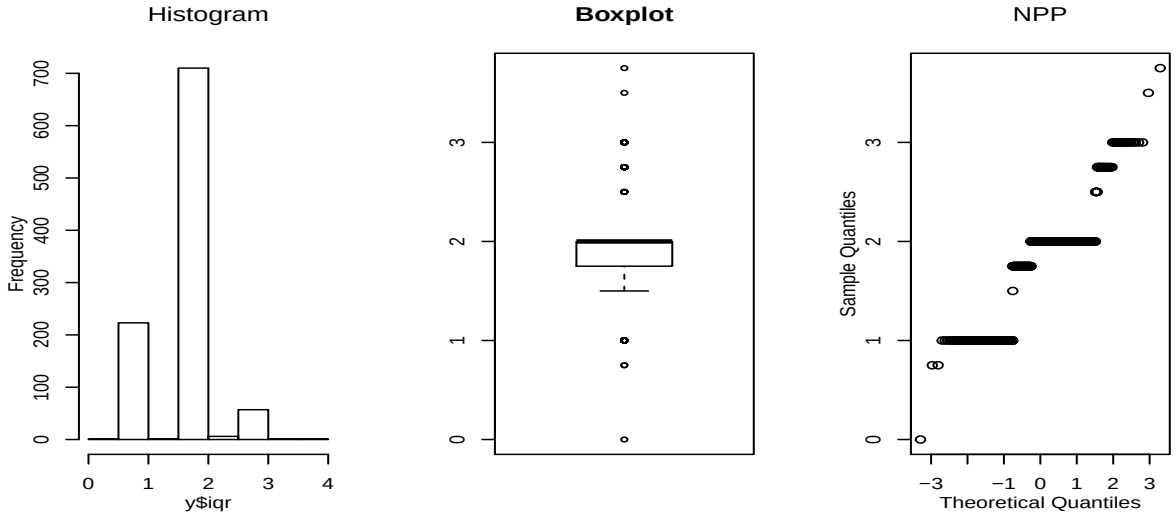
The function `simpois` draws a sample of size `n` from a Poisson distribution with mean `lambda` `nn` times; for each sample computes six sample statistics, namely the mean, median, standard deviation, minimum, maximum and inter quartile range; stores these six vectors (each one of length `nn`) in a list; and returns this list as the value of the function. Here is how this function is used to study CLT through numerical experimentation.

```
> y<-simpois(30,1000,2)
```

```
> hist(y$mean,main="Histogram",font.main=1)
> boxplot(y$mean,main="Boxplot",font.main=1)
> qqnorm(y$mean,main="NPP",font.main=1)
```




```
> hist(y$iqr,main="Histogram",font.main=1)
> boxplot(y$iqr,main="Boxplot",font.main=1)
> qqnorm(y$iqr,main="NPP",font.main=1)
```



First note that we are calling `simpois` with arguments `n`=30, `nn`=1000 and `lambda`=2 and storing the results in `y`. This means we are generating a sample of size 30 from a Poisson(2) distribution 1000 times and the 1000 sample means, medians, standard deviations, minimums, maximums and IQRs are being returned in the list `y`. The named components of a

list are accessed using the "$" operator in R. Since here the components of the list are named as mean, median, sd, minm, maxm and iqr, `y$mean` and `y$iqr` respectively give the 1000 sample means and IQR.

Next to see whether the sampling distributions of these sample statistics are at least approximately Normal, we are employing three simple graphical techniques namely the histogram, box and whiskers plot and Normal Probability Plot (NPP). For Normal distribution the histogram should be bell shaped, the box and whiskers plot should be symmetric about the central line within the box, and the NPP should be a straight line. Thus we see that though the sampling distribution of the sample mean of Poisson(2) is approximately Normal, that of the sample IQR is not, for n=30.

Whether the sampling distribution of a sample statistic is approximately Normal or not, depends primarily on three things - (1) the nature of the original population distribution, like Poisson in the given example; (2) the nature of the sample statistic, like mean and IQR in the given example; and (3) the sample size n, which is 30 in the given example. Larger the `nn` better one gets the idea about the distribution of the statistic, but it has nothing to do with the Normal approximation per say. For this assignment you may take `nn` to be 1000 but to understand its role I suggest you work with a few more values like 100, 2000, 5000 and 10000. Here is the assignment:

1. Generate observations from the following distributions:

Uniform(0,1)	`runif(n,a,b)` generates n observations from Uniform(a,b)
B(12,0.01) B(20,0.5) B(18,0.95)	`rbinom(n,m,p)` generates n observations from B(m,p)
Poisson(0.001) Poisson(1) Poisson(25)	`rpois(n,lambda)` generates n observations from Poisson(λ)
Exponential(0.001) Exponential(1) Exponential(25)	`rexp(n,lambda)` generates n observations from Exponential(λ)
Gamma(0.001,1) Gamma(2,1) Gamma(30,1)	`rgamma(n,alpha,lambda)` generates n observations from Gamma(α, λ)
Beta(2,2) Beta(20,2) Beta(2,8) Beta(0.5,0.5) Beta(0.5,0.2) Beta(0.2,5)	`rbeta(n,alpha,beta)` generates n observations from Beta(α, β)[1]
Cauchy(0,1)	`rcauchy(n)` generates n observations from Cauchy(0,1)
Normal(0,1)	`rnorm(n)` generates n observations from Normal(0,1)

[1]See §4-12 of Montgomery and Runger for the definition of a Beta(α, β) distribution.

2. Confine yourself to the six sample statistics as in the example *viz.* the mean, median, standard deviation, minimum, maximum and inter quartile range.

3. For each parent distribution and for each statistic, numerically experiment whether its sampling distribution is approximately Normal or not. In case it is, explore the minimum value of n for which one gets this normality.

4. Write a brief report summarising your findings, highlighting when and how quickly one achieves normality for what kind of statistic and parent distribution. Also comment on the situations when one cannot achieve normality. Obviously you can comment only on the distribution of the six statistics under consideration, and thus prepare the summary report with six sections, one for each statistic; but your discussion must not be addressed to the particular parent distributions listed in the table above. The listed distributions are meant to serve only as examples from which you should be able to generalise about the features of the parent distributions in terms of their existence of moments, symmetry etc. for achieving normality for a given statistic. In order to do this, it will be essential for you to be able to visualise the parent distributions. Towards this end,

   ```
   > x<-seq(0,1,0.01)
   > plot(x,dbeta(x,2,2),type="l")
   ```

   for example plots the p.d.f. of $\beta(2,2)$. Similar commands for computing the p.m.f./p.d.f. of Binomial, Poisson, Uniform, Normal, Exponential and Gamma respectively are `dbinom, dpois, dunif, dnorm, dexp` and `dgamma`. For plotting p.m.f.s as in the document "Probability Models" in the "Handouts" page, use the function `pmfplot` defined in "pmfplot.r" in the "R Files" page in the MG 221 home-page.