Course Project for 'Statistical Inference' on coursera.org
Part 1 - Study of the exponential distribution

R. Rodríguez

The instructions for this part of the project say the following:

> The exponential distribution can be simulated in R with rexp(n, lambda) where lambda is the rate parameter. The mean of exponential distribution is 1/lambda and the standard deviation is also also 1/lambda. Set lambda = 0.2 for all of the simulations. In this simulation, you will investigate the distribution of averages of 40 exponential(0.2)s. Note that you will need to do a thousand or so simulated averages of 40 exponentials.
>
> Illustrate via simulation and associated explanatory text the properties of the distribution of the mean of 40 exponential(0.2)s. You should
>
> 1. Show where the distribution is centered at and compare it to the theoretical center of the distribution.
> 2. Show how variable it is and compare it to the theoretical variance of the distribution.
> 3. Show that the distribution is approximately normal.
> 4. Evaluate the coverage of the confidence interval for 1/lambda: $\bar{X} \pm 1.96 S_n$.

NOTE 1: All R code in this report is wrapped into functions, which are called when necessary. The definition of the functions can be found in the appendix at the end of the document.

NOTE 2: Sourcecode for this document available at https://github.com/ricrogz/StatisticInference

The first thing to do is to set the random number generator seed, to make everythong reproducible, and run the simulations to generate the data and calculate the averages.

```
set.seed(12345)
lambda  <- 0.2; n_draws <- 40; n_sim   <- 1000
data <- genData(lambda, n_draws, n_sim)
```

Now we can start to solve the questions.

Our first duty is to find the value around which the distribution of averages of our simulated data is centered, which, according to the lessons, should be the mean of the generated averages, and compare it to the theoretical center of the distribution, which is the mean of the exponential distribution (which we know that can be calculated as `1/lambda`).

```
calcCenters()
```

```
## Simulation Center  Theorical Center
##             4.972             5.000
```

Both of the values that we get, 4.972 for the center (mean) of averages of the simulated data, and 5 for the theoretical center (mean) of the exponential distribution are indeed very close, but are not identical. According to the Law of Large Numbers, if we evaluated an infinite number of simulations, both values would be identical, but since we have only evaluated a finite number of simulations (namely 1000 simulations), despite this number is large enough, the values are close, but not identical.

Our second task is to study the variability of the averages. We can easily ask R to calculate the standard deviation of our averages, and we also know that, due to the Central Limit Theorem, the theorical standard deviation of the averages distribution should be equal to $\sigma/\sqrt{n}$, where σ is the standard deviation of the exponential distribution, which we know is equal to `1/lambda`, and n is the number of draws or samples taken to calculate the averages; in our case, n = 40.

```
calcSDs()
```

```
## Simulation S.D.  Theorical S.D.
##           4.972           5.000
```

Also in this case, both standard deviations, 0.7847 for our simulations, and 0.7906 for the theorical value are very close, as it was expected.

The next task is to show that the distribution of averages we have obtained is approximately normal. To demonstrate this we will make two plots, which are shown in Appendix 1.

In the first figure, we plot our simulated data against a normal probability density distribution curve. Due to the space limitations. The appropriate parameters for this normal distribution we are going to compare with are already known to us: they are the values we have calculated in our two previous tasks. In this way, we are going to use the `sim_center` value as mean of the normal distribution, and the `sim_sd`as standard deviation parameter.

Visually, the profile of the density curve matches the profile of the simulation data in a very close way, but this is not enough. The second plot is a QQ plot, in which the simulated data is compared quantile-by-quantile with a standard normal distribution. For this, the distribution of averages of the simulations is transformed by means of the relationship $\bar{Z} = \frac{\bar{X}-\mu_X}{\sigma_X}$. As with the other figure, this one has been pushed into the Appendix 1.

In the Quantile-by-Quantile plot, we can see that most of the simulation data lay over or very close to the line which defines a perfect match with the theorical quantiles. The most notable exceptions are at the extremes of the plot, and these represent only a marginal fraction of the data, so, from this quantile-by-quantile comparison we can conclude that our simulated data indeed is very close to a normal distribution.

Finally, our last task is to evaluate the coverage of the confidence interval defined by $\bar{X} \pm 1.96\frac{S}{sqrtn}$ (this interval covers 95% of the probability range, leaving only 2.5% at each side of the interval). To calculate the confidente interval we use the mean and standard deviation from our simulated data (in this exercise we know the parameters for the exponential distribution behind our simulation, but in the "real world" we would not know, or would not be certain about the parameters of the distribuion from which our samples come from).

```
calcConfInterval (1.96)
```

```
## Lower Limit Upper Limit
##       3.434       6.510
```

When checking our simulated average values against this interval, using our function `checkConfInterval()` , we see that the 95.3% of the simulated averages are within this confidence interval, which is consistent with the definition of the interval (95% of confidence).

Appendix 1 : Figures

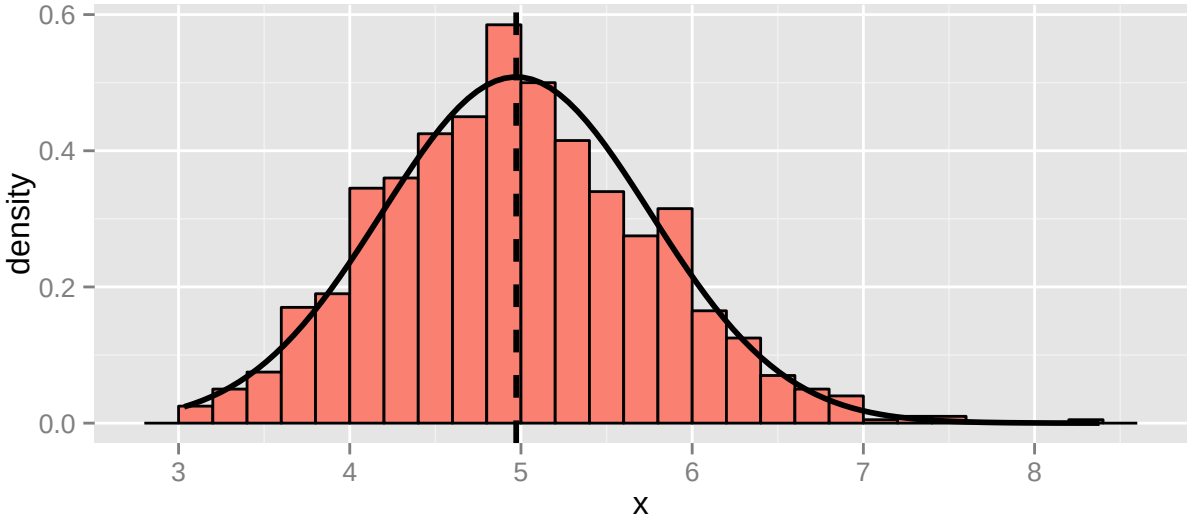

Figure 1: Distribution of averages of 40 simulated exponentials with lambda = 0.2

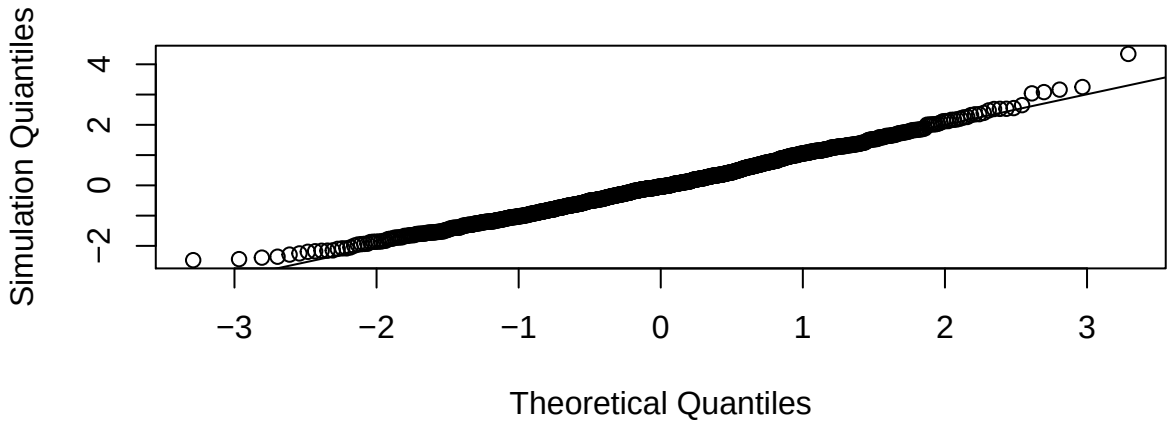

Figure 2: QQ plot of averages of 40 simulated exponentials with lambda = 0.2

Appendix 2 : Code

```
genData <- function (lambda, n_draws, n_sim) {
  # Generation of 'n_sim' cases of averages of 'n_draws' data samples from an
  # exponential distribution with parameter lambda (stored as data$x).
  # Column sz is merely informative and is not used.
  data <- data.frame(
    x  = apply(matrix(rexp(n_draws * n_sim, lambda), n_sim), 1, mean),
    sz = factor(rep(n_draws,n_sim))
  )
}

calcCenters <- function () {
  # calculate the centres (means) both for the simulated data and the theorical
  # predicition
  sim_center     <<- mean(data$x);
  theoric_center <<- 1/lambda;
  c("Simulation Center"=sim_center, "Theorical Center"=theoric_center)
}

calcSDs <- function () {
  # calculate the standard deviations both for the simulated data and the theorical
  # predicition
  sim_sd     <<- sd(data$x);
  theoric_sd <<- (1/lambda) / sqrt(n_draws);
  c("Simulation S.D."=sim_center, "Theorical S.D."=theoric_center)
}

plotDists <- function () {
  # use ggplot to plot both the distribution of simulated averages, and a matching
  # normal distribution with mean "sim_center" and standard deviation "sim_sd",
  # adding a vertical dashed line indicating the center.
  g <- ggplot(data, aes(x = x))
  g <- g + geom_histogram(binwidth=.2, colour="black",
                          fill="salmon", aes(y = ..density..))
  g <- g + stat_function(fun = dnorm, size = 1, args = list(sim_center, sim_sd))
  g + geom_vline(aes(xintercept=sim_center), linetype="dashed", size=1)
}

plotQQnorm <- function () {
  # make a QQ plot, to compare the distribution of averages of simulated data
  # with a normal distribution Quantile-by-Quantile-wise.
  # This time, the simulations distribution is converted to resemble a N(0,1)
  # distribution  transform simulation data to resemble Z(0,1) distribution:
  qqx <- (data$x - sim_center) / sim_sd

  # plot
  qqnorm(qqx, ylab = "Simulation Quiantiles", main="")
  qqline(qqx)
}

calcConfInterval <- function (SigmaFactor) {
  # calculate confidente interval
```

```
  conf_interval <<- sim_center + c("Lower Limit"=-1, "Upper Limit"=1) * SigmaFactor * sim_sd
  conf_interval
}

checkConfInterval <- function () {
  # take the averages and the confidence intervals and, first, calculate how many
  #of the averages are inside the interval  and return what fraction of the total
  # averages data it represents
  coverage <<- length( data$x[data$x >= conf_interval[1] &
                                data$x <= conf_interval[2]]  ) / n_sim
}
```