Statistical Inference Course Project

Edilmo Palencia

Overview

In this report we analize the exponetial distribution looking at the behaviour of the mean, the variance and the distribution. In order to achive this, we are going to run multiple simulations and compare their behaviors with the theorical one.

Simulations

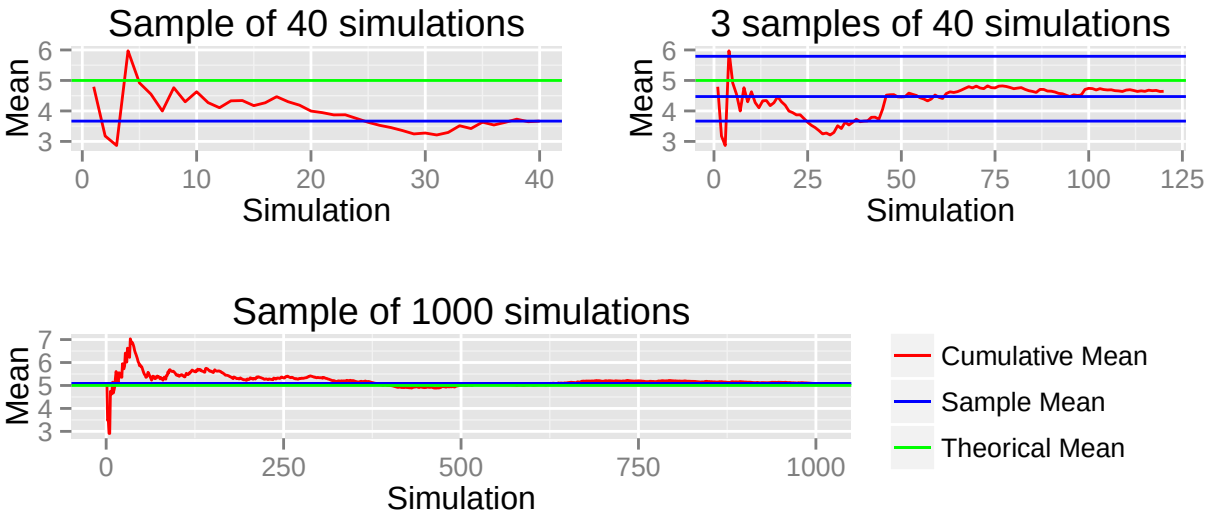
In the next chunk of code we are going to:
- Set all the parameters of the evaluation: the rate or lambda value to use for all the simulations; the size of the large sample; the size of short samples; the amount of short samples to generate.
- Generate all the simulations: one large sample of 1000 simulations; one thousand short samples of 40 simulations.
- Compute the mean and the standar deviation for all the samples.

```
# Generate the large sample
simulation.th <- rexp(n.th, lambda)
# Compute the mean of the large sample
simulation.th.mean <- mean(simulation.th)
# Compute the standar deviation of the large sample
simulation.th.sd <- sd(simulation.th)
# Generate a list of short samples
simulations.me <- lapply(rep(lambda,n.me), function(l){ rexp(n.sa,lambda)})
# Compute the mean of the short samples
simulations.me.means <- sapply(simulations.me, mean)
# Compute the mean of the short samples
simulations.me.sd <- sapply(simulations.me, sd)
```

Sample Mean versus Theoretical Mean

The next chunk of code is a example of how was generated the 3 figures showed below. Specifically, the code showed correspond to the last figure.

```
# Compute the cumulative average of the simulation
y <- cumsum(simulation.th)/(1:n.th)
# Create the ggplot object with the data
g.svm.3 <- ggplot(data.frame(x = 1 : n.th, y = y), aes(x = x, y = y))
# Add a red line for the cumulative average of the simulation
g.svm.3 <- g.svm.3 + geom_line(aes(colour = "Cumulative Mean"))
# Add a blue line for the sample mean of the simulation
g.svm.3 <- g.svm.3 + geom_hline(aes(yintercept = simulation.th.mean, colour = "Sample Mean"), show_guid
# Add a green line for the theorical mean
g.svm.3 <- g.svm.3 + geom_hline(aes(yintercept = mean.theorical, colour = "Theorical Mean"), show_guide
# Add the labels of the axis and the title
g.svm.3 <- g.svm.3 + labs(x = "Simulation", y = "Mean") + ggtitle("Sample of 1000 simulations")
# Add the legend
g.svm.3 <- g.svm.3 + scale_colour_manual("", breaks=c("Cumulative Mean", "Sample Mean", "Theorical Mean
```

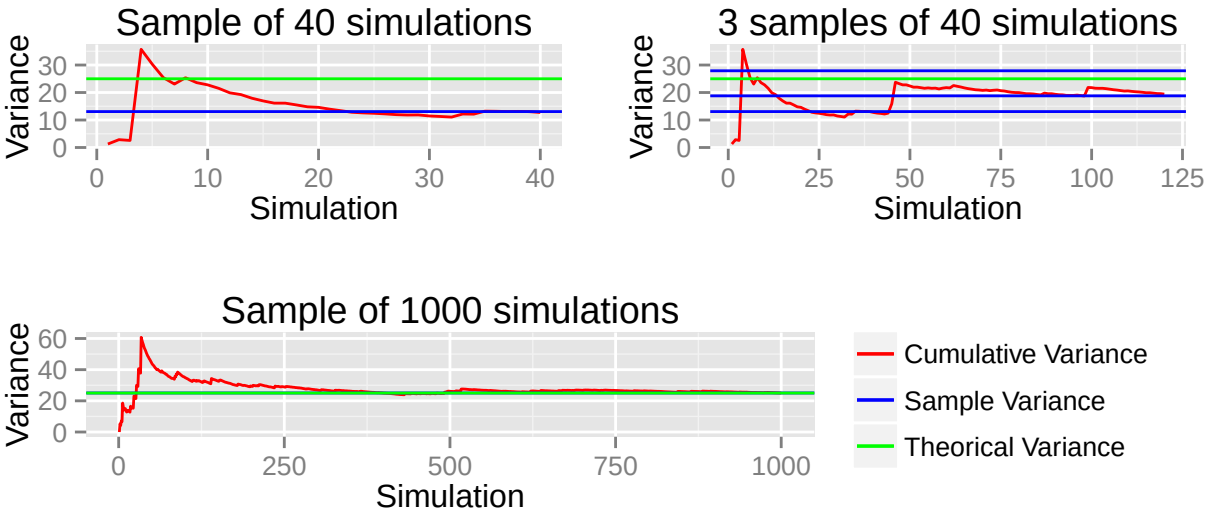


These figures allows see how the mean of a sample becomes more and more equal to the theorical mean when we increase the amount of simulations present in the sample. The red lines ilustrate this, they represent the cumulative average of the sample (Law of Large Numbers). The blue and green line show the Sample and the Theorical means respectivaly.

- The first figure show the behaviour of a short sample of Means of 3.6662 and a difference with the theorical of -1.3338.
- The second figure show the behaviour of tree short sample of Means of 3.6662, 5.792434, 4.471801, and a difference with the theorical of -1.3338, 0.7924345, -0.5281988 respectivaly.
- The third figure show the behaviour of a large sample of Means of 5.088809 and a difference with the theorical of 0.08880917.

Sample Variance versus Theoretical Variance

Following a procedure very similar to the one used to compare the means, it was generated the figures showed bellow.



These figures allows see how the variance of a sample becomes more and more equal to the theorical variance when we increase the amount of simulations present in the sample. The red lines ilustrate this, they represent the cumulative variance of the sample (Law of Large Numbers). The blue and green line show the Sample and the Theorical variances respectivaly.

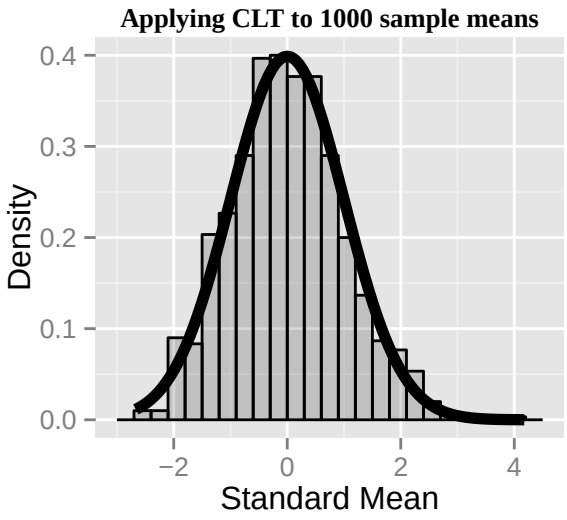
- The first figure show the behaviour of a short sample of Variance of 13.06583 and a difference with the

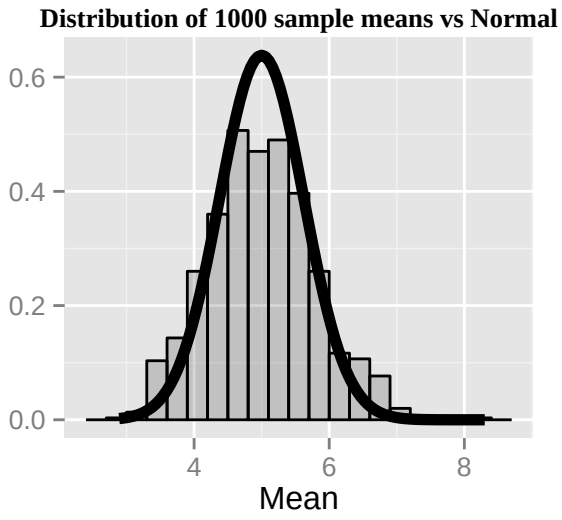
theorical of -11.93417.
- The second figure show the behaviour of tree short sample of Variances of 13.06583, 27.91333, 18.7878, and a difference with the theorical of -11.93417, 2.913329, -6.212203 respectivaly.
- The third figure show the behaviour of a large sample of Means of 25.04994 and a difference with the theorical of 0.04993878.

Distribution

The two figures below show that the distribution of the means of the short samples is aproximately normal.
- The first figure normalize the means following the CLT and draw a standard normal to compare with.
- The second figure compares the distribution of the means directly with a normal of mean equal to the theorical mean for the exponential, standard deviation equal to σ^2/n.

```
# Compute the normalize means of the simulations following CLT
x <- ((sqrt(n.sa)*(simulations.me.means-mean.theorical))/sd.theorical)
# Create the ggplot object with the data
g.dis.1 <- ggplot(data.frame(x = x), aes(x = x))
# Draw a histogram
g.dis.1 <- g.dis.1 + geom_histogram(alpha = .20, binwidth=.3, colour = "black", aes(y = ..density..))
# Draw a standard normal
g.dis.1 <- g.dis.1 + stat_function(fun = dnorm, size = 2)
# Add the labels of the axis and the title
g.dis.1 <- g.dis.1 + labs(x = "Standard Mean", y = "Density") + ggtitle("Applying CLT to 1000 sample mea
```





Appendix

In this appendix we are including all the chuncks of codes in strict order.

Simulations

```
# Set value of lambda for all the experiments
lambda <- 0.2
# Set the size of a large sample
```

```
n.th <- 1000
# Set the size for short samples
n.sa <- 40
# Set the amount of short samples
n.me <- 1000
# Set the theorical mean for the exponential distribution
mean.theorical <- 1/lambda
# Set the theorical standar deviation for the exponential distribution
sd.theorical <- 1/lambda
# Set the theorical variance for the exponential distribution
variance.theorical <- sd.theorical^2
```

```
# Generate the large sample
simulation.th <- rexp(n.th, lambda)
# Compute the mean of the large sample
simulation.th.mean <- mean(simulation.th)
# Compute the standar deviation of the large sample
simulation.th.sd <- sd(simulation.th)
# Generate a list of short samples
simulations.me <- lapply(rep(lambda,n.me), function(l){ rexp(n.sa,lambda)})
# Compute the mean of the short samples
simulations.me.means <- sapply(simulations.me, mean)
# Compute the mean of the short samples
simulations.me.sd <- sapply(simulations.me, sd)
```

Sample Mean versus Theoretical Mean

```
library(ggplot2)
# Compute the cumulative average of the simulation
y <- cumsum(simulations.me[[1]])/(1:n.sa)
# Create the ggplot object with the data
g.svm.1 <- ggplot(data.frame(x = 1 : n.sa, y = y), aes(x = x, y = y))
# Add a red line for the cumulative average of the simulation
g.svm.1 <- g.svm.1 + geom_line(colour = "red")
# Add a blue lines for the sample mean
g.svm.1 <- g.svm.1 + geom_hline(yintercept = simulations.me.means[[1]], colour = "blue", show_guide = F
# Add a green line for the theorical mean
g.svm.1 <- g.svm.1 + geom_hline(yintercept = mean.theorical, colour = "green", show_guide = FALSE)
# Add the labels of the axis and the title
g.svm.1 <- g.svm.1 + labs(x = "Simulation", y = "Mean") + ggtitle("Sample of 40 simulations")
```

```
# Compute the cumulative average of the simulation
y <- cumsum(c(simulations.me[[1]], simulations.me[[2]], simulations.me[[3]]))/(1:(3*n.sa))
# Create the ggplot object with the data
g.svm.2 <- ggplot(data.frame(x = 1 : (3*n.sa), y = y), aes(x = x, y = y))
# Add a red line for the cumulative average of the simulation
g.svm.2 <- g.svm.2 + geom_line(colour = "red")
# Add a blue line for the samples means
g.svm.2 <- g.svm.2 + geom_hline(yintercept = simulations.me.means[[1]], colour = "blue", show_guide = F
g.svm.2 <- g.svm.2 + geom_hline(yintercept = simulations.me.means[[2]], colour = "blue", show_guide = F
g.svm.2 <- g.svm.2 + geom_hline(yintercept = simulations.me.means[[3]], colour = "blue", show_guide = F
# Add a green line for the theorical mean
g.svm.2 <- g.svm.2 + geom_hline(yintercept = mean.theorical, colour = "green", show_guide = FALSE)
```

```r
# Add the labels of the axis and the title
g.svm.2 <- g.svm.2 + labs(x = "Simulation", y = "Mean") + ggtitle("3 samples of 40 simulations")
```

```r
# Compute the cumulative average of the simulation
y <- cumsum(simulation.th)/(1:n.th)
# Create the ggplot object with the data
g.svm.3 <- ggplot(data.frame(x = 1 : n.th, y = y), aes(x = x, y = y))
# Add a red line for the cumulative average of the simulation
g.svm.3 <- g.svm.3 + geom_line(aes(colour = "Cumulative Mean"))
# Add a blue line for the sample mean of the simulation
g.svm.3 <- g.svm.3 + geom_hline(aes(yintercept = simulation.th.mean, colour = "Sample Mean"), show_guid
# Add a green line for the theorical mean
g.svm.3 <- g.svm.3 + geom_hline(aes(yintercept = mean.theorical, colour = "Theorical Mean"), show_guide
# Add the labels of the axis and the title
g.svm.3 <- g.svm.3 + labs(x = "Simulation", y = "Mean") + ggtitle("Sample of 1000 simulations")
# Add the legend
g.svm.3 <- g.svm.3 + scale_colour_manual("", breaks=c("Cumulative Mean", "Sample Mean", "Theorical Mean
```

```r
library(grid)
grid.newpage()
pushViewport(viewport(layout = grid.layout(2, 2)))
print(g.svm.1, vp = viewport(layout.pos.row = 1, layout.pos.col = 1))
print(g.svm.2, vp = viewport(layout.pos.row = 1, layout.pos.col = 2))
print(g.svm.3, vp = viewport(layout.pos.row = 2, layout.pos.col = c(1,2)))
```

Sample Variance versus Theoretical Variance

```r
# Compute the cumulative variance of the simulation
y <- cumsum((simulations.me[[1]]-simulations.me.means[[1]])^2)/(1:n.sa)
# Create the ggplot object with the data
g.svm.1 <- ggplot(data.frame(x = 1 : n.sa, y = y), aes(x = x, y = y))
# Add a red line for the cumulative variance of the simulation
g.svm.1 <- g.svm.1 + geom_line(colour = "red")
# Add a blue lines for the sample variance
g.svm.1 <- g.svm.1 + geom_hline(yintercept = ((simulations.me.sd[[1]])^2), colour = "blue", show_guide =
# Add a green line for the theorical variance
g.svm.1 <- g.svm.1 + geom_hline(yintercept = variance.theorical, colour = "green", show_guide = FALSE)
# Add the labels of the axis and the title
g.svm.1 <- g.svm.1 + labs(x = "Simulation", y = "Variance") + ggtitle("Sample of 40 simulations")
```

```r
# Compute the cumulative variance of the simulation
y <- cumsum(c((simulations.me[[1]]-simulations.me.means[[1]])^2, (simulations.me[[2]]-simulations.me.mea
# Create the ggplot object with the data
g.svm.2 <- ggplot(data.frame(x = 1 : (3*n.sa), y = y), aes(x = x, y = y))
# Add a red line for the cumulative variance of the simulation
g.svm.2 <- g.svm.2 + geom_line(colour = "red")
# Add a blue line for the samples variances
g.svm.2 <- g.svm.2 + geom_hline(yintercept = ((simulations.me.sd[[1]])^2), colour = "blue", show_guide =
g.svm.2 <- g.svm.2 + geom_hline(yintercept = ((simulations.me.sd[[2]])^2), colour = "blue", show_guide =
g.svm.2 <- g.svm.2 + geom_hline(yintercept = ((simulations.me.sd[[3]])^2), colour = "blue", show_guide =
# Add a green line for the theorical variance
g.svm.2 <- g.svm.2 + geom_hline(yintercept = variance.theorical, colour = "green", show_guide = FALSE)
```

```
# Add the labels of the axis and the title
g.svm.2 <- g.svm.2 + labs(x = "Simulation", y = "Variance") + ggtitle("3 samples of 40 simulations")
```

```
# Compute the cumulative variance of the simulation
y <- cumsum((simulation.th-simulation.th.mean)^2)/(1:n.th)
# Create the ggplot object with the data
g.svm.3 <- ggplot(data.frame(x = 1 : n.th, y = y), aes(x = x, y = y))
# Add a red line for the cumulative variance of the simulation
g.svm.3 <- g.svm.3 + geom_line(aes(colour = "Cumulative Variance"))
# Add a blue line for the sample variance of the simulation
g.svm.3 <- g.svm.3 + geom_hline(aes(yintercept = ((simulation.th.sd)^2), colour = "Sample Variance"), sl
# Add a green line for the theorical variance
g.svm.3 <- g.svm.3 + geom_hline(aes(yintercept = variance.theorical, colour = "Theorical Variance"), sh
# Add the labels of the axis and the title
g.svm.3 <- g.svm.3 + labs(x = "Simulation", y = "Variance") + ggtitle("Sample of 1000 simulations")
# Add the legend
g.svm.3 <- g.svm.3 + scale_colour_manual("", breaks=c("Cumulative Variance", "Sample Variance", "Theori
```

```
grid.newpage()
pushViewport(viewport(layout = grid.layout(2, 2)))
print(g.svm.1, vp = viewport(layout.pos.row = 1, layout.pos.col = 1))
print(g.svm.2, vp = viewport(layout.pos.row = 1, layout.pos.col = 2))
print(g.svm.3, vp = viewport(layout.pos.row = 2, layout.pos.col = c(1,2)))
```

Distribution

```
# Compute the normalize means of the simulations following CLT
x <- ((sqrt(n.sa)*(simulations.me.means-mean.theorical))/sd.theorical)
# Create the ggplot object with the data
g.dis.1 <- ggplot(data.frame(x = x), aes(x = x))
# Draw a histogram
g.dis.1 <- g.dis.1 + geom_histogram(alpha = .20, binwidth=.3, colour = "black", aes(y = ..density..))
# Draw a standard normal
g.dis.1 <- g.dis.1 + stat_function(fun = dnorm, size = 2)
# Add the labels of the axis and the title
g.dis.1 <- g.dis.1 + labs(x = "Standard Mean", y = "Density") + ggtitle("Applying CLT to 1000 sample mea
```

```
# Assign the means
x <- simulations.me.means
# Create the ggplot object with the data
g.dis.2 <- ggplot(data.frame(x = x), aes(x = x))
# Draw a histogram
g.dis.2 <- g.dis.2 + geom_histogram(alpha = .20, binwidth=.3, colour = "black", aes(y = ..density..))
# Draw a normal with theorical mean and the modified standard deviation following CLT
g.dis.2 <- g.dis.2 + stat_function(fun = dnorm, args = list(mean=mean.theorical, sd=(variance.theorical/
# Add the labels of the axis and the title
g.dis.2 <- g.dis.2 + labs(x = "Mean", y = "") + ggtitle("Distribution of 1000 sample means vs Normal") +
```

```
grid.newpage()
pushViewport(viewport(layout = grid.layout(1, 2)))
print(g.dis.1, vp = viewport(layout.pos.row = 1, layout.pos.col = 1))
print(g.dis.2, vp = viewport(layout.pos.row = 1, layout.pos.col = 2))
```