Statistical Inference Simulation Project

Mark Bulkeley

April 2, 2016

Overview

This report will explore the exponential distribution from both a theoretical and simulation standpoint, making use of the Central Limit Theorem. The report will use the stastical language R as the basis for the computations. It will be done in a reproducible fashion, so the reader can validate all of the results independently. All of the figures can be found in the Appendix.

Simulations

Before we do simulations, we should note that we are going to compare the theoretical exponential distribution with a sampled one using a fixed lambda. The key parameters for our simulation are set below:

```
lambda <- 0.2           # of exponential distribution
set.seed(seed = 39548)  # set for reproducibility
samples <- 1000         # number of simulations run
draws <- 40             # number of draws from exp. distribution for each simulation
```

To see what the theoretical distribution looks like, see Figure 1. You will note that the mean of the distribution is 5. In the sampling, we've taken the mean of 40 random draws. The code to do the sampling is compact, so we show it here:

```
result <- do.call(rbind, lapply(1:samples, function(x){
    smpl <- rexp(n = draws, rate = lambda)
    return(data.table(mn=mean(smpl), vr=var(smpl), iteration=x))
}))
```

Sample Mean versus Theoretical Mean

As mentioned before the theoretical mean is simply 1 / lambda, or in our case 5. The mean of sample means has come in very close to the theory at 5.013. That said, the distribution of means is reasonably wide and some estimates are lower than 3.5 and some higher than 7. Figure 2 shows the distribution of the sample means and how it compares to the theoretical (population) mean.

Sample Variance versus Theoretical Variance

The standard deviation of an exponential distribution is the same as its mean, `sd = 1/lambda`. So, given that variance is the square of the standard deviation, theoretical (population) variance of the exponential distribution is `var = sd^2` or 25. Our mean realized (sample) variance from the 1000 samples is 25.246, which is quite close. See Figure 3 for a perspective on how close. Note that one is the population variance and one is the sample variance, our best estimate of the population variance. The mean of sample variances turns out to be a reasonably good estimate of the population mean, but the same caveat for the means applies; there are some very high and some very low estimates. However, in terms of bootstrapping (though it should be understood that is not exactly what we have done), the number of iterations and draws are sufficient here to estimate the population well.

Distribution: Approximately Normal

Specifically, here we are going to look at distribution of sample means to see if it approximates the normal distribution. First, we are going to look at it in terms of theoretical quantiles and realized quantiles for different p values and then look at a graphical view in Figure 4. For the normal distribution, we use them sample mean and sample standard deviation to make the comparison.

Table 1: Practical Comparison of Normality

p	Realized	Theoretical	Absolute Difference (Percent Of Theoretical)
1%	3.364	3.140	7.1
5%	3.775	3.688	2.3
10%	4.000	3.981	0.5
25%	4.452	4.470	0.4
50%	4.959	5.013	1.1
75%	5.547	5.556	0.2
90%	6.096	6.044	0.8
95%	6.405	6.337	1.1
99%	7.024	6.886	2.0

As we can see, the percent difference, especially near the mean is very low between the theoretical normal and the distribution of the sample means. Figure 4 also highlights this by showing that the density functions are almost on top of each other, when graphed. Alternately, we could run a statistical test such as the Shapiro-Wilk Normality Test. When run, the results should be significant if the `p` value is less than 0.1. Our results:

```
shapiro.test(result$mn)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  result$mn
## W = 0.99137, p-value = 1.358e-05
```

The p-value is much lower than 0.1, so we should be able to assume normality.

Conclusions

Our approach has applied a bootstrap-like approach to compare theoretical mean and variance to that which would be inferred by sampling. It has shown that the bootstrap-like approach can result in very close estimates of the actual population statistics and thus can prove to be a valuable modeling tool in the data analysis.

Appendix: Figures

Figure 1: Theoretical Exponential Distribution

```
plotRange <- 1:1e5/1e3
dt.plot <- data.table(x = plotRange, y = dexp(x=plotRange, rate = lambda))
ggplot(data=dt.plot, mapping=aes(x = x, y = y)) +
    geom_hline(yintercept = 0) +
    geom_line(lwd=2) +
    scale_x_continuous(name = "X") +
    scale_y_continuous(name = "Probability Density Function of\nExponential Distribution") +
    geom_vline(xintercept = 1 / lambda, col = "red") +
    annotate(geom = "text", x = 1 / lambda + 1, y = 0.1,
             label = paste0("Mean of distribution is ", 1/lambda), hjust = 0, col="red") +
    ggtitle("The Exponential Distribution\nLambda = 0.2")
```

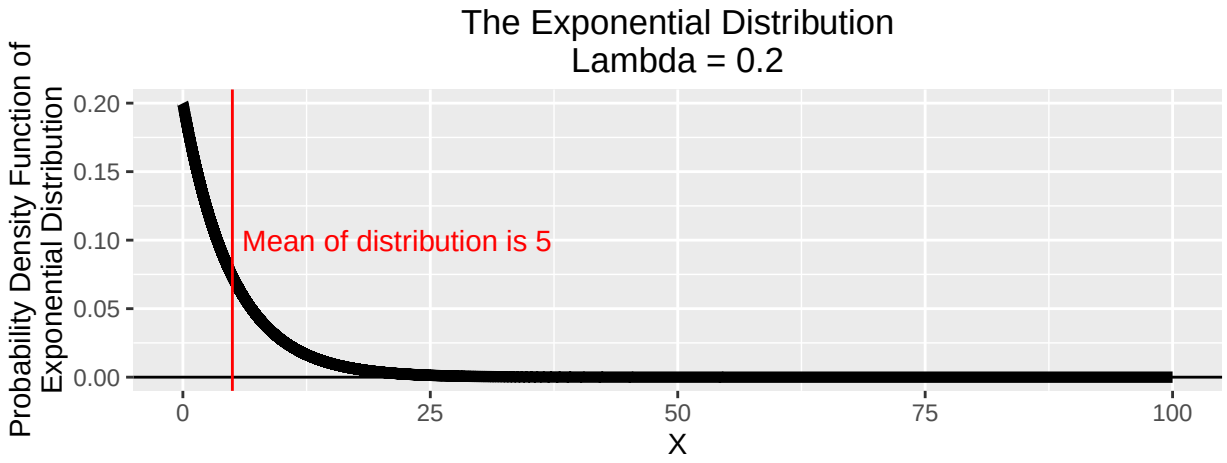


Figure 2: Distribution of Means

```
ggplot(data=result, mapping=aes(x=mn)) +
    geom_hline(yintercept = 0) +
    geom_density(lwd = 2) +
    geom_vline(xintercept = mean(result$mn), col = "red") +
    annotate(geom = "text", x = mean(result$mn) + 0.1, y = 0.2,
             label = paste0("Mean of sampled\ndistribution is ",round(mean(result$mn),3)),
             hjust = 0, col = 'red', size = 2.5) +
    geom_vline(xintercept = 1/lambda, col = "blue") +
    annotate(geom = "text", x = 1/lambda - 0.1, y = 0.2,
             label = paste0("Theoretical Mean of\ndistribution is ",1/lambda),
             hjust = 1, col = 'blue', size = 2.5) +
    scale_y_continuous(name = "Density") +
    scale_x_continuous(name = "Mean of 40 Samples") +
    ggtitle("Sampled Exponential Mean\nLambda = 0.2")
```

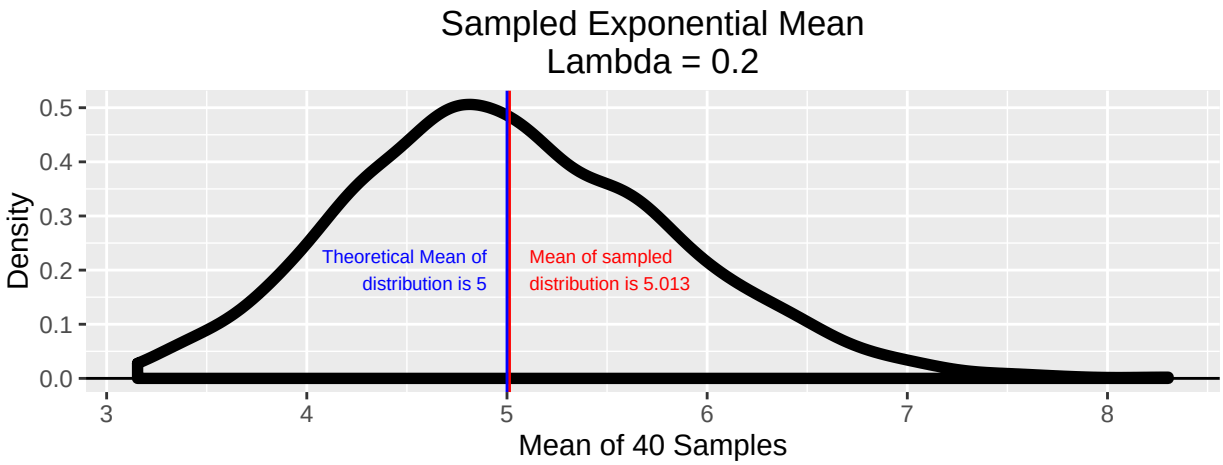


Figure 3: Distribution of Variance

```
ggplot(data=result, mapping=aes(x=vr)) +
    geom_hline(yintercept = 0) +
    geom_density(lwd = 2) +
    geom_vline(xintercept = mean(result$vr), col = "red") +
    annotate(geom = "text", x = mean(result$vr) + 1, y = 0.02,
             label = paste0("Mean Variance of sampled\ndistribution is ",
                            round(mean(result$vr),3)),
             hjust = 0, col = 'red', size = 2.5) +
    geom_vline(xintercept = (1/lambda)^2, col = "blue") +
    annotate(geom = "text", x = (1/lambda)^2 - 1, y = 0.02,
             label = paste0("Theoretical Variance of\ndistribution is ",(1/lambda)^2),
             hjust = 1, col = 'blue', size = 2.5) +
    scale_y_continuous(name = "Density") +
    scale_x_continuous(name = "Variance of 40 Samples") +
    ggtitle("Sampled Exponential Variance\nLambda = 0.2") +
    coord_cartesian(xlim = quantile(result$vr, c(0.01, 0.99)))
```

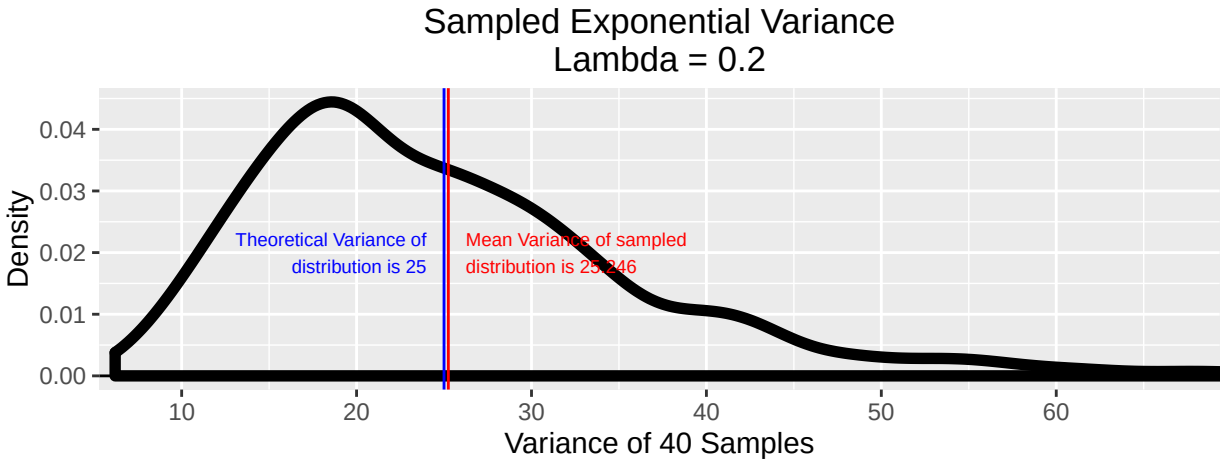
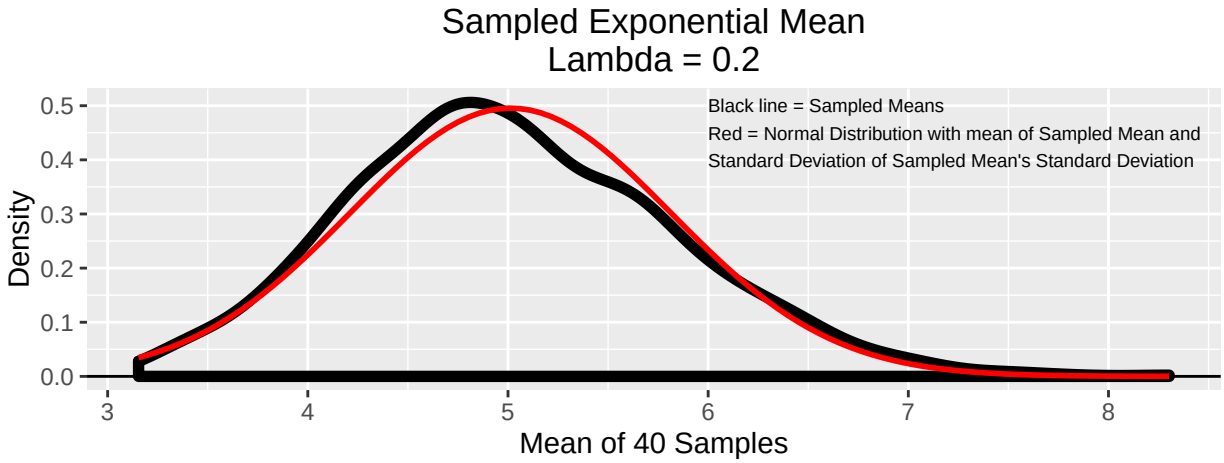


Figure 4: Normality of Sample Means

```
ggplot(data=result, mapping=aes(x=mn)) +
    geom_hline(yintercept = 0) +
    geom_density(lwd = 2) +
    stat_function(fun=dnorm, args=list(mean=mean(result$mn), sd=sd(result$mn)),
                  size=1.0, col = "red") +
    scale_y_continuous(name = "Density") +
    scale_x_continuous(name = "Mean of 40 Samples") +
    ggtitle("Sampled Exponential Mean\nLambda = 0.2") +
    annotate(geom="text", x= 6, y = .45, label = paste0("Black line = Sampled Means\n",
        "Red = Normal Distribution with mean of Sampled Mean and \n",
        "Standard Deviation of Sampled Mean's Standard Deviation"), hjust = 0,
        size = 2.5)
```



Code to Generate Table 1

```
p <- c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 0.9, 0.95, 0.99)
quants <- data.table(p = p, Realized = quantile(result$mn, p),
                     Theoretical = qnorm(p, mean = mean(result$mn), sd = sd(result$mn)))
quants[, diff := round(100 * abs(Realized - Theoretical)/Theoretical, 1)]
grid.arrange(tableGrob(quants[, .(p = paste0(p*100, "%"),
                                  Realized = round(Realized,3),
                                  Theoretical = round(Theoretical,3),
                                  `Absolute Percent Of\nTheoretical Difference` = diff)],
                       rows = NULL, theme = ttheme_minimal()))
```