Sample Mean Statistic

Steve Swann

03 September 2015

Introduction

We obtain a random sample from a population.

1. What can we infer from this sample about the population?
2. Can we estimate the mean of the population?
3. How big does the sample need to be?
4. Can the probability density of the population be of any type, or must it be normally distributed?
5. What happens if, though bad luck, the sample we draw does not represent the population very well?
6. Can we use numeric techniques to estimate the mean of a population?

Fortunately we can simulate these scenarios in R and empirically model what actually occurs.

Drawing a Single Sample

We have a population P of numeric values and draw a sample s from the population P.

From the sample s we compute the sample mean and the sample standard deviation.

```r
# population P normal distribution of size n=100,000
P = rnorm(n=100000, mean=25, sd=5)

# draw sample s of size=size.of.sample from the population P
size.of.sample = 1000
s = sample(P, size=size.of.sample, replace=FALSE)
sample.mean = mean(s)
sample.sd = sd(s)
print (paste('sample.mean =', round(sample.mean,2)))
```

```
## [1] "sample.mean = 25.12"
```

```r
print (paste('sample.sd =', round(sample.sd,2)))
```

```
## [1] "sample.sd = 5.23"
```

Does *sample.mean* always approximate the population mean. If we make random draws from the population P how will the value of *sample.mean* vary. We can use R to simulate multiple random draws and then plot the results.

Drawing Multiple Samples From The Population

When we draw multiple samples we obtain a mean for each sample taken. In this context we need to think of the sample mean as a statistic.

```r
no.of.random.samples = 1000
# sample.means will contain each sample.mean
sample.means = numeric()
i = 1
while (i <= no.of.random.samples) {
  s = sample(P, size=size.of.sample, replace=FALSE)
  sample.mean = mean(s)
  sample.means = c(sample.means, sample.mean)
  i = i + 1
}
```

We now have 1,000 sample means and can make a density plot of *sample.mean*. Superimposed on the density plot we place a theoretical normal distribution curve for the sample means in red. This curve is given by the function:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

where

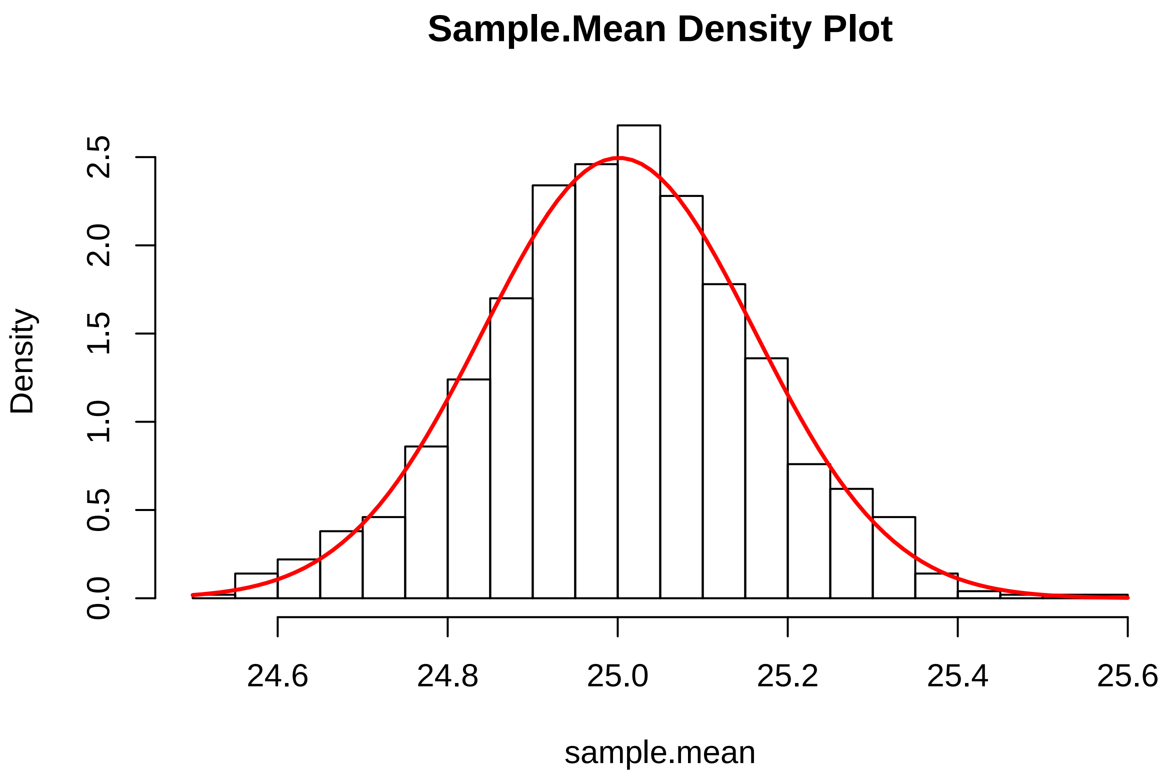
$$\mu = mean(sample.means)$$

and

$$sd = sd(sample.means)$$

.

```r
# normal distribution function
normDist = function(x,mean,sd) {
  return ((1/(sd*sqrt(2*pi)))*exp(-((x -mean)^2)/(2*sd^2)))
}
hist(sample.means, breaks=16, main='Sample.Mean Density Plot',xlab ='sample.mean', freq=FALSE)
mean.sample.means = mean(sample.means)
sd.sample.means = sd(sample.means)
curve(normDist(x,mean=mean.sample.means,sd=sd.sample.means), col="red", lwd=2, add=TRUE)
```



```
## [1] "mean.sample.means = 25.001"
```

```
## [1] "sd.sample.means = 0.16"
```

As can be seen the red curve reasonably matches the empirical density of the *sample.means*. The standard deviation of the *sample.means* is called the **standard error**.

From the theoretical normal distribution for the sample mean (eq 1) we can predict the lower and upper bound for the *sample.mean*. We call this the **confidence interval**.

Suppose we select a confidence interval of 60% and simulate this in R.

```
conf.int = 0.6
lower.bound = qnorm((1 - conf.int)/2, mean = mean.sample.means, sd = sd.sample.means, lower.tail = TRUE)
upper.bound = mean.sample.means + (mean.sample.means - lower.bound)
```

```
## [1] "lower.bound = 24.867"
```

```
## [1] "upper.bound = 25.136"
```

If we use these lower and upper bounds we can expect to find approximately 40% of each sample.mean in sample.means to fall outside these bounds. Let's check.

```
below.lower.bound.count = sum(sample.means <= lower.bound)
above.upper.bound.count = sum(sample.means >= upper.bound)
sample.means.outside.ci.count = below.lower.bound.count + above.upper.bound.count
sample.means.outside.ci.percent = 100 * sample.means.outside.ci.count/length(sample.means)
```

```
## [1] "sample.means.outside.ci.percent = 38.7"
```

Depending on the problem we generally want the **confidence interval** to be better than 60%. Typical values are 90%, 95% and 99%. The bigger we make the **confidence interval** the wider the upper and lower bound for the *sample.mean*.

To show this we can generate a 99% confidence interval.

```
conf.int = 0.99
lower.bound = qnorm((1 - conf.int)/2, mean = mean.sample.means, sd = sd.sample.means, lower.tail = TRUE
upper.bound = mean.sample.means + (mean.sample.means - lower.bound)
```

```
## [1] "lower.bound = 24.59"
```

```
## [1] "upper.bound = 25.413"
```

Infering the population mean from a sigle random sample

In a real world situation we generally cannot draw multiple samples, certainly not a 1,000 times as per the simulation. But we can make use of the **Central Limit Theorem**. If we draw a sample of "reasonable size" from **any** real world population, and compute the sample.mean and sample.sd we can infer the following:

1. Standard error of the sample mean statistic

$$se_{sample.mean} = \frac{sd_{sample}}{\sqrt{size.of.sample}}$$

2. Best estimate of population mean

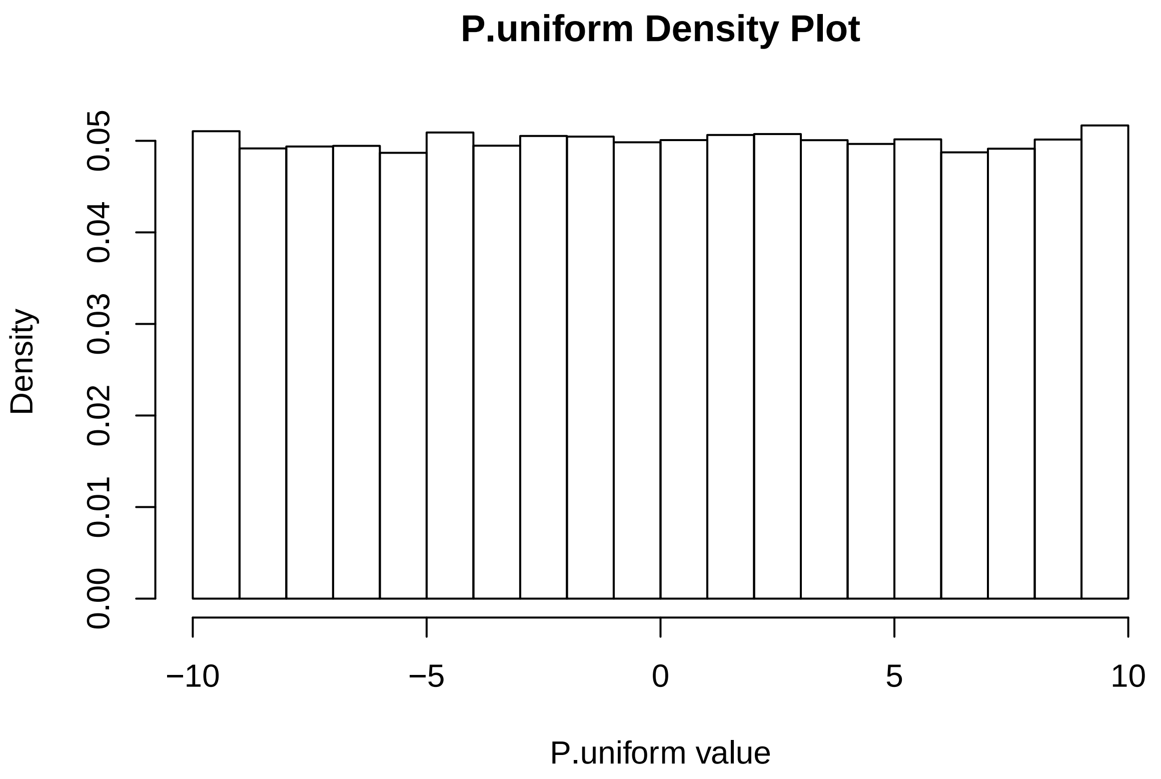
$$population.mean_{best.point.estimate} = mean_{sample}$$

3. Sample mean statistic is normally distributed

$$sample.mean \approx NormalDistribution(mean = mean_{sample}, sd = se_{sample.mean})$$

We use a different example to illustrate the power of the CLT. Suppose the population P is a uniform distribution between -10 and +10.

```
P.uniform = runif(100000, min=-10, max=10)
hist(P.uniform, breaks=16, main='P.uniform Density Plot',xlab ='P.uniform value', freq=FALSE)
```



```
mean.P.uniform = mean(P.uniform)
sd.P.uniform = sd(P.uniform)
```

Notice P.uniform is definately not a normal distribution. We also compute the exact values for the population mean and standard deviation.

```
## [1] "mean.P.uniform = 0.012"
```

```
## [1] "sd.P.uniform = 5.776"
```

We can estimate *se* and *sample.mean* of a sample drawn from P.uniform.

```
s = sample(P.uniform, size=size.of.sample, replace=FALSE)
mean.sample = mean(s) # var name !!!!
sd.sample = sd(s)
# from CLT, se = standard error
se = sd(s)/sqrt(size.of.sample)
```

```
## [1] "mean.sample = 0.143"
```

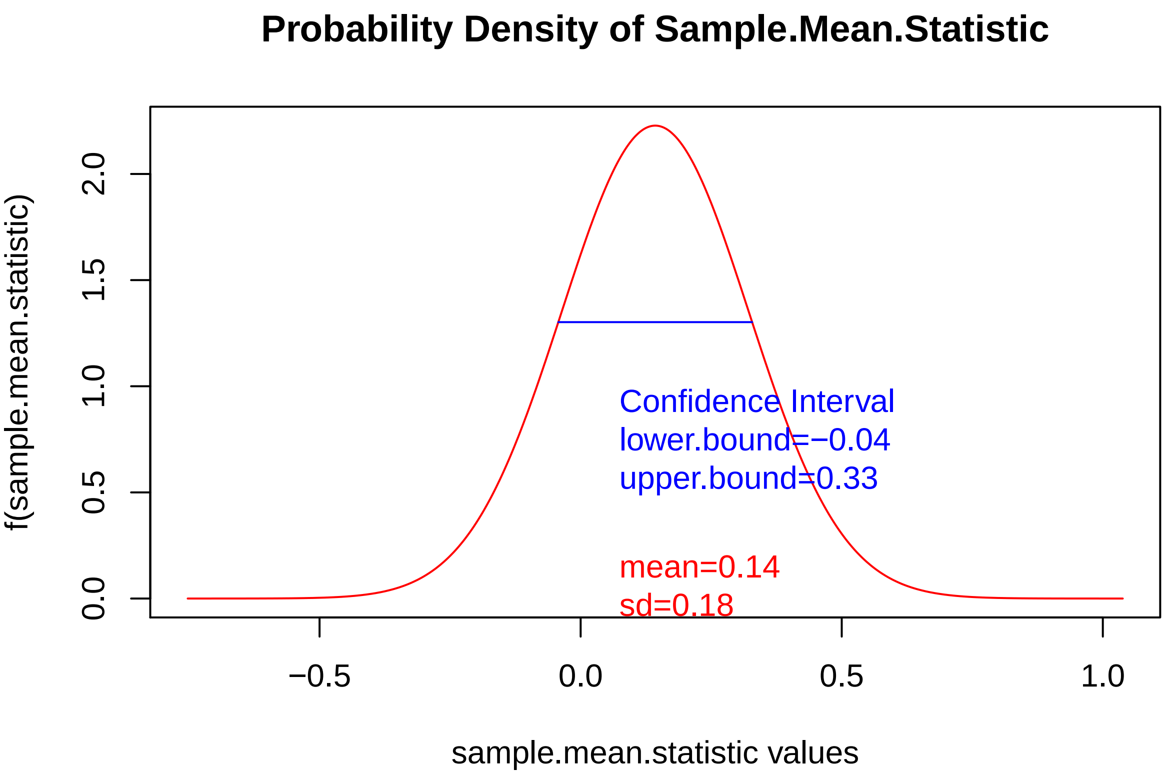
```
## [1] "sd.sample = 5.663"
```

```
## [1] "se.population.mean.statistic = 0.179"
```

Now we have sufficient information to plot the theoretical distribution of the sample mean statistic. We can also add a *confidence interval* of arbitary value, in this case we pick 70%, and is shown as the blue line.

```r
lb = mean.sample - 5*se
ub = mean.sample + 5*se
x = seq(lb, ub, length.out=1000)
y = lapply(x, normDist, mean=mean.sample, sd=se)
y = unlist(y)
plot(x,y, col="red", type='l',
     main='Probability Density of Sample.Mean.Statistic',
     xlab='sample.mean.statistic values',
     ylab='f(sample.mean.statistic)'
     )
legend(x=mean.sample - 0.2*(mean.sample - lb), y=0.3*max(y)/2,
       legend=c(paste('mean=',round(mean.sample,2),sep=''), paste('sd=',round(se,2),sep='')),
       text.col = 'red', bty='n')

conf.int = 0.70 # arbitrarily chosen
lower.bound = qnorm((1 - conf.int)/2, mean = mean.sample, sd = se, lower.tail = TRUE)
upper.bound = mean.sample + (mean.sample - lower.bound)
y.lower.bound = normDist(lower.bound, mean.sample, se)
y.upper.bound = normDist(upper.bound, mean.sample, se)
lines(c(lower.bound, upper.bound), c(y.lower.bound, y.upper.bound), col='blue')
legend(x=mean.sample - 0.2*(mean.sample - lb), y=1.0*max(y)/2,
       legend=c('Confidence Interval', paste('lower.bound=',round(lower.bound,2),sep=''),
       paste('upper.bound=',round(upper.bound,2),sep='')),
       text.col = 'blue', bty='n')
```



How do we interpret the **confidence interval**. We expect the population mean to be on the interval lower.bound to upper.bound "most of the time".

If we draw multiple random samples from the population using a 70% CI, we can expect the population mean to be on the interval approximately 70 percent of the time. We can simulate this and check the results.

```
no.of.random.samples = 1000
# sample.means will contain each sample.mean
sample.means = numeric()
population.mean.on.CI.count = 0
i = 1
while (i <= no.of.random.samples) {
  s = sample(P.uniform, size=size.of.sample, replace=FALSE)
  sample.mean = mean(s)
  sd.sample = sd(s)
  se = sd(s)/sqrt(size.of.sample)
  lower.bound = qnorm((1 - conf.int)/2, mean = sample.mean, sd = se, lower.tail = TRUE)
  upper.bound = sample.mean + (sample.mean - lower.bound)
  if ((mean.P.uniform >= lower.bound) & (mean.P.uniform <= upper.bound)) {
    population.mean.on.CI.count = population.mean.on.CI.count + 1
  }
  i = i + 1
}
print (paste('population.mean.on.CI.count:',round(100*population.mean.on.CI.count/no.of.random.samples,
```

```
## [1] "population.mean.on.CI.count: 71 %"
```

Conclusions

1. Can we estimate the mean of the population?

We can estimate the mean of the population. But we cannot provide an absolute answer. We can provide a likely interval for the population mean.

2. How big does the sample need to be?

We have not answered that question yet and will try to answer it in a seperate document. For the purpose of this document we have used the idea of "reasonable sample size".

3. Can the probability density of the population be of any type, or must it be normally distributed?

Yes, any population distribution is fine, thanks to the Central Limit Theorem.

4. What happens if, though bad luck, the random sample we draw does not represent the population very well?

We will review the implications of this in a seperate document.

5. Can we use numeric techniques to estimate the mean of a population?

We will examine this in a later document.

Notation

Commonly used notation for population and sample parameters.

Population mean μ and population standard deviation σ.

Sample mean $\bar{X}$ and sample standard deviation s

Citations

Sample Means [http://www.stat.yale.edu/Courses/1997-98/101/sampmn.htm]