# Confidence Intervals

*Dr. John R. Vokey*

*2018-12-03*

## What are Confidence Intervals?

It seems to me that despite the unceasing cry for confidence intervals instead of significance tests most people understand confidence intervals even less well than they do significance tests. What does a 95% confidence interval actually mean? The most common (mis)interpretation is that there is a .95 probability that the parameter (typically mu) is within the interval. But that is just wrong, even (or especially) if one is frequentist in the iterpretation of probabilities: the parameter is either in the interval or it is not, no probability about it. That is why Neyman, the inventor of confidence intervals, called the scale an index of "confidence" not probability: according to Neyman, you are 95% *confident* that the parameter is within the interval (whatever that means!), granting the truth of the distributional assumptions (e.g., normal for t-tests).

The most common frequentist interpretation is as follows. If the sampling is repeated many, many times, 95% of those times the computed intervals will contain the parameter, mu, as shown in demo0 with numtests set to 100000:

```
numtests <- 100000
cnt <- 0
i1 <- 0
i2 <- 0
mus <- 0
means <- 0
for (i in 1:numtests){
  n <- 1000
  mu <- 0
  sd <- 1
  df <- n-1
  x<-rnorm(n,mu,sd)
  m <-mean(x)
  se<-sd(x)/sqrt(n)
  t=qt(.025,df,lower.tail=FALSE)
  int1 <- m-t*se
  int2 <- m+t*se
  if (mu < int1 | mu > int2) cnt <- cnt+1
  i1[i] <- int1
  i2[i] <- int2
  mus[i] <- mu
  means[i] <- m
}
print(cnt/numtests)
```

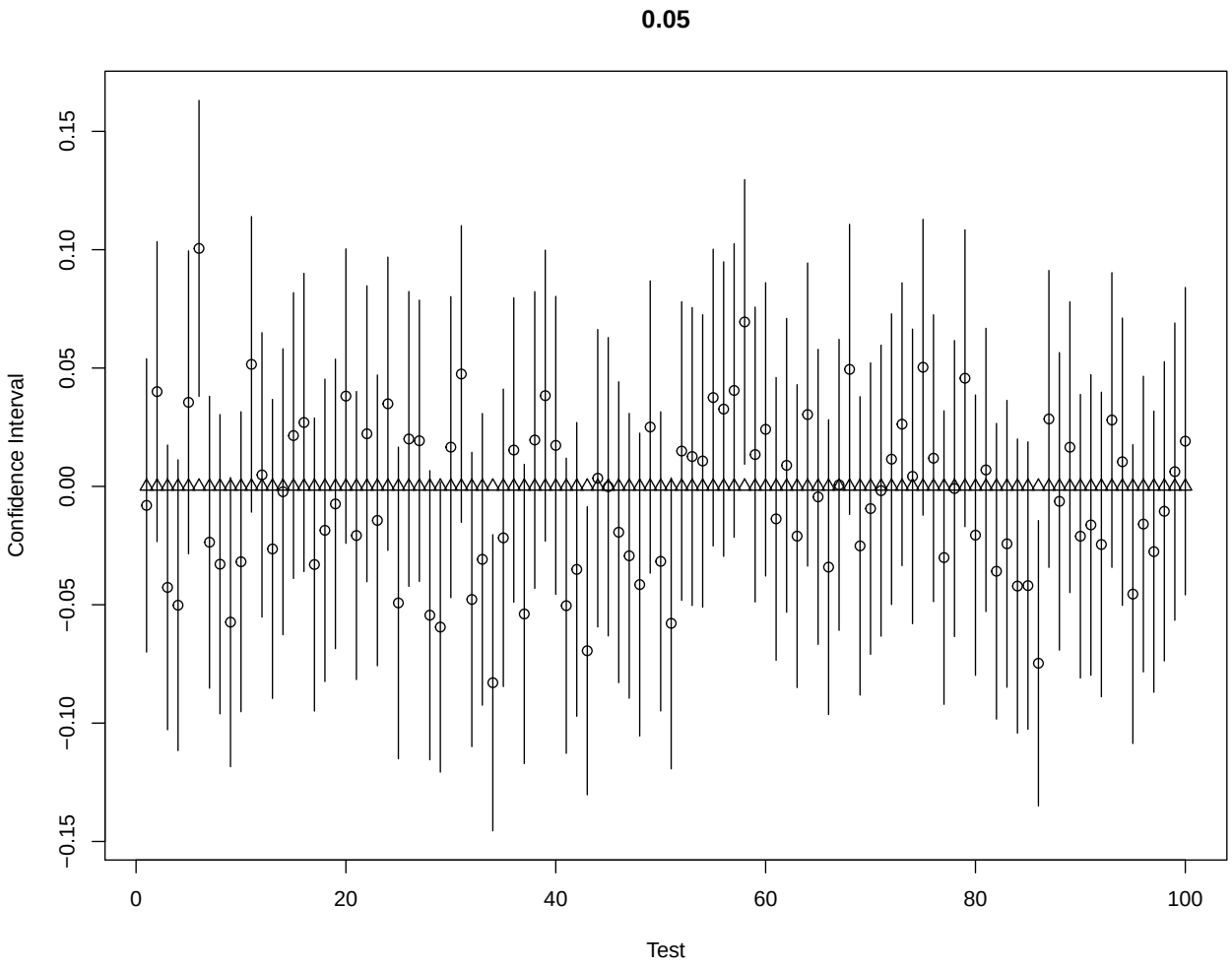
```
## [1] 0.04971
```

(For the remaining demos, numtests will be set 100 so we can plot the results.) The most common demonstration of this interpretation is as illustrated by the following r-code, which repeats for 100 tests sampling 1000 scores from a normal distribution with mu = 0 and sigma = 1, and then plots the 100 confidence intervals along with their sample means (circles) and mu (triangles), returing the p-value for the number that fall outside of the interval (which should be circa .05). Look at the plot and count the number that fail to include mu (they can be difficult to see).

```r
numtests <- 100
cnt <- 0
i1 <- 0
i2 <- 0
mus <- 0
means <- 0
for (i in 1:numtests){
  n <- 1000
  mu <- 0
  sd <- 1
  df <- n-1
  x<-rnorm(n,mu,sd)
  m <-mean(x)
  se<-sd(x)/sqrt(n)
  t=qt(.025,df,lower.tail=FALSE)
  int1 <- m-t*se
  int2 <- m+t*se
  if (mu < int1 | mu > int2) cnt <- cnt+1
  i1[i] <- int1
  i2[i] <- int2
  mus[i] <- mu
  means[i] <- m
}
print(cnt/numtests)
```

```
## [1] 0.05
```

```r
plot(means, ylim=c(min(i1), max(i2)),main=paste(cnt/numtests),
     ylab="Confidence Interval",
     xlab="Test")
segments(x0=1:numtests,y0=i1,x1=1:numtests,y1=i2)
points(mus,pch=2)
```

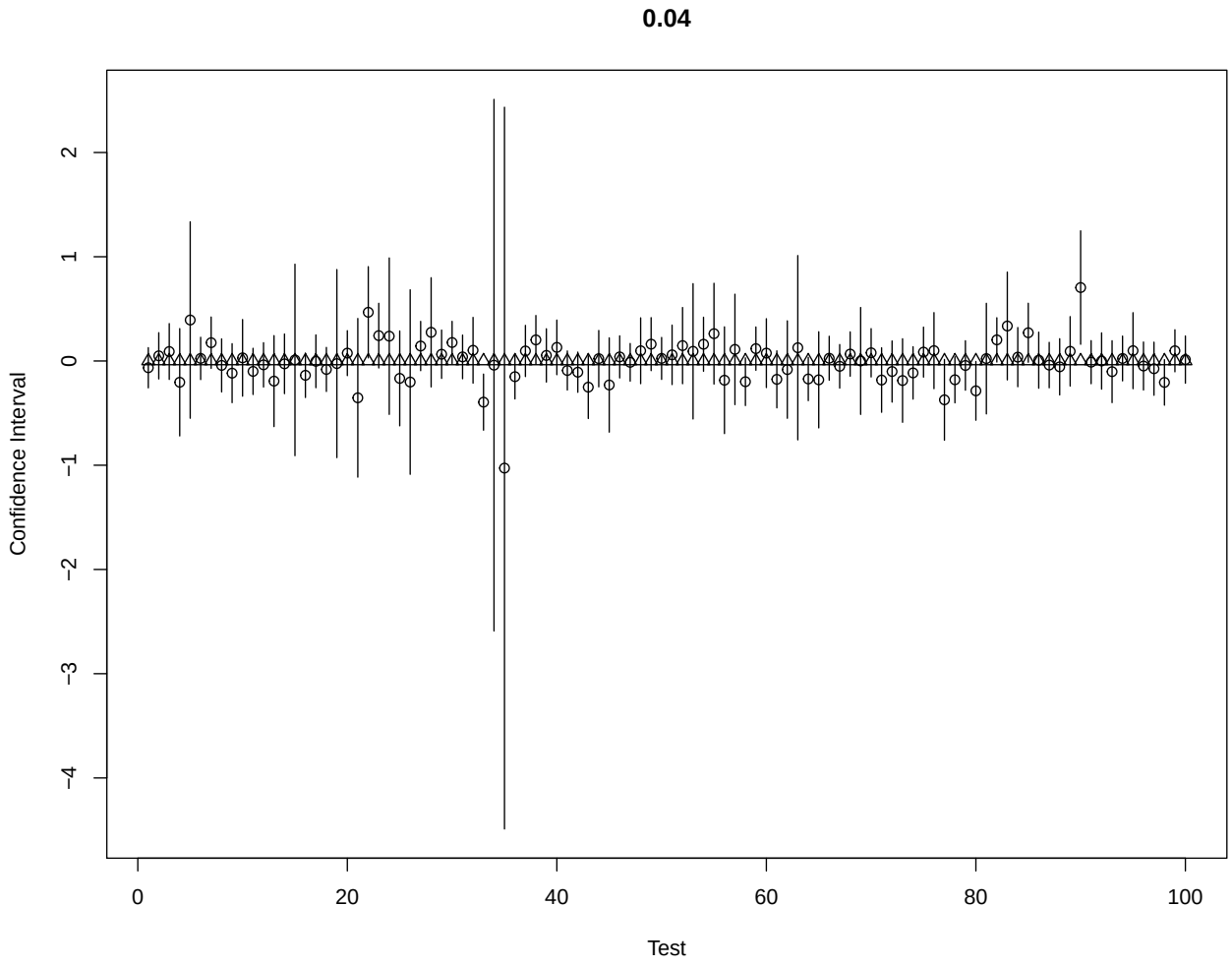



But that is sampling the *same* number of scores from from the *same* normal distribution each time. How important is that? In the next demo, we vary the sample size:

```
numtests <- 100
cnt <- 0
i1 <- 0
i2 <- 0
mus <- 0
means <- 0
for (i in 1:numtests){
  n<-runif(1,min=2,max=100)
  mu <- 0
  sd <- 1
  df <- n-1
  x<-rnorm(n,mu,sd)
  m <-mean(x)
  se<-sd(x)/sqrt(n)
  t=qt(.025,df,lower.tail=FALSE)
  int1 <- m-t*se
  int2 <- m+t*se
  if (mu < int1 | mu > int2) cnt <- cnt+1
  i1[i] <- int1
  i2[i] <- int2
```

```
  mus[i] <- mu
  means[i] <- m
}
print(cnt/numtests)
```

```
## [1] 0.04
```

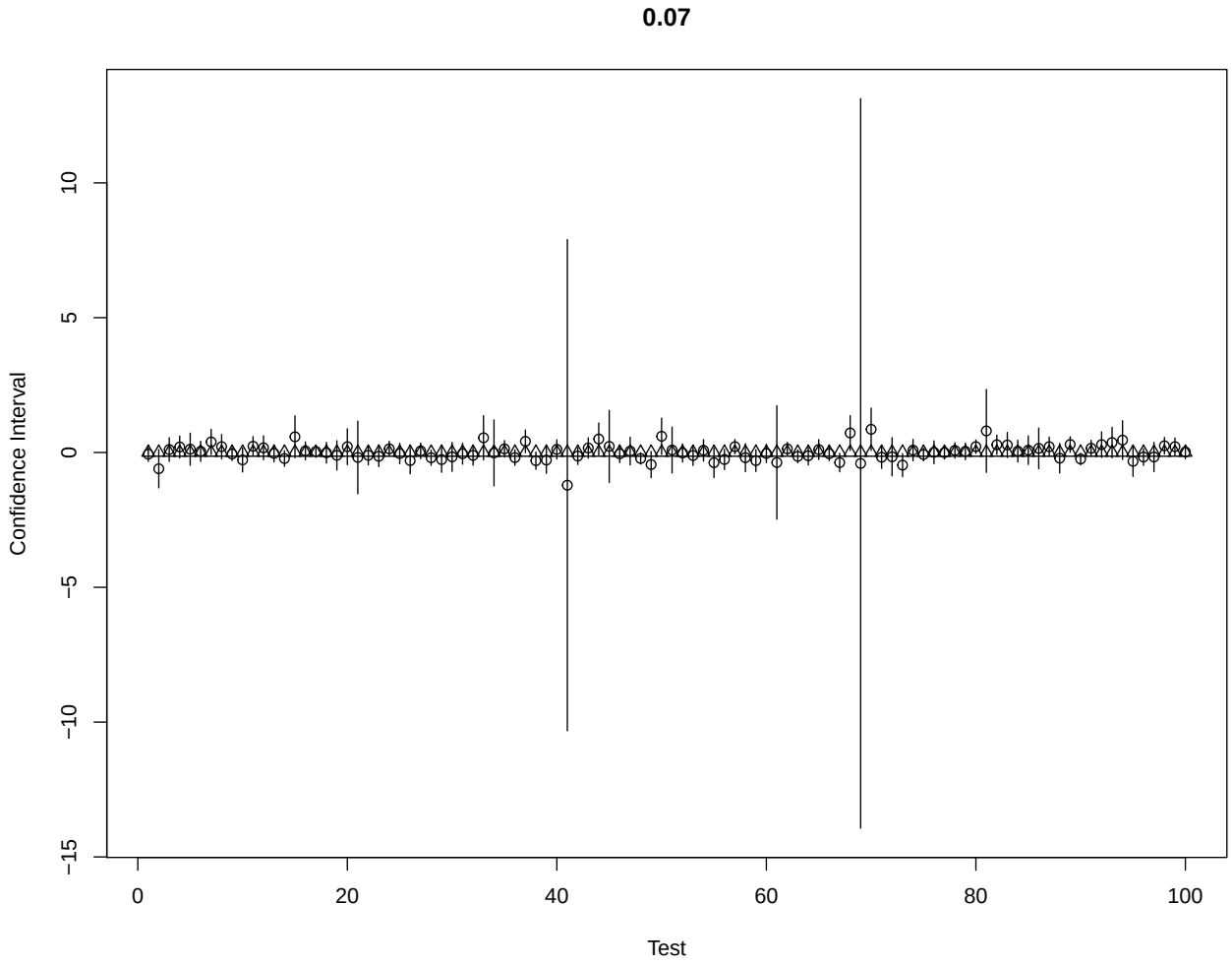



Doesn't matter one bit. So, the confidence interval logic does not require identical sample sizes. How about the distribution itself? Can the sigma vary from one test to the next?

```
numtests <- 100
cnt <- 0
i1 <- 0
i2 <- 0
mus <- 0
means <- 0
for (i in 1:numtests){
  n<-runif(1,min=2,max=100)
  mu <- 0
  sd <- runif(1,min=1,max=2) # keep the range small so the plot is not noisy
  df <- n-1
  x<-rnorm(n,mu,sd)
  m <-mean(x)
  se<-sd(x)/sqrt(n)
```

```
  t=qt(.025,df,lower.tail=FALSE)
  int1 <- m-t*se
  int2 <- m+t*se
  if (mu < int1 | mu > int2) cnt <- cnt+1
  i1[i] <- int1
  i2[i] <- int2
  mus[i] <- mu
  means[i] <- m
}
print(cnt/numtests)
```

```
## [1] 0.07
```

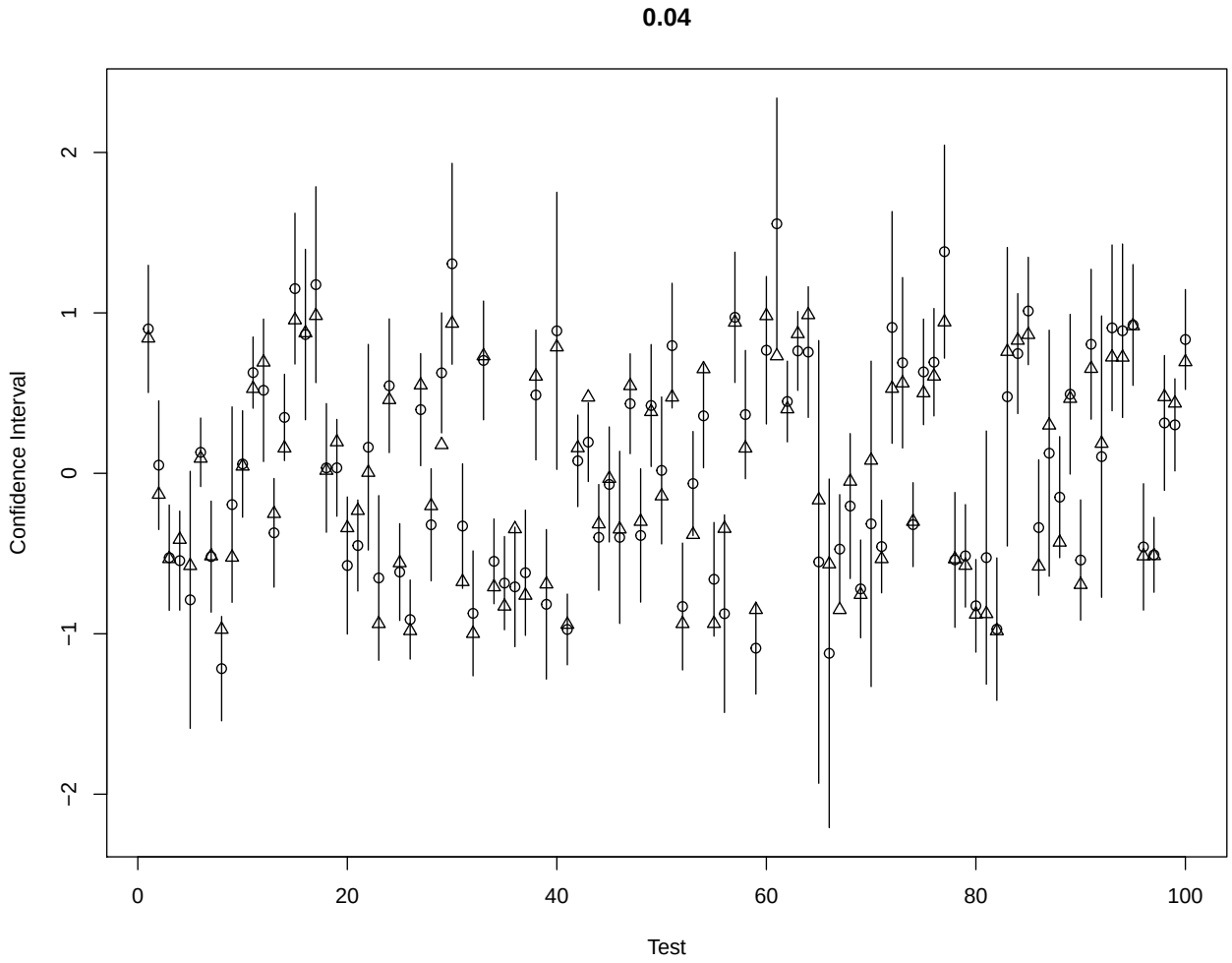



Yup. Still works. How about mu? Does the parameter, at least, need to remain constant?

```
numtests <- 100
cnt <- 0
i1 <- 0
i2 <- 0
mus <- 0
means <- 0
for (i in 1:numtests){
  n<-runif(1,min=10,max=100)
  mu <- runif(1,min=-1,max=1) # keep the range small so the plot is not noisy
```

```
  sd <- runif(1,min=1,max=2) # keep the range small so the plot is not noisy
  df <- n-1
  x<-rnorm(n,mu,sd)
  m <-mean(x)
  se<-sd(x)/sqrt(n)
  t=qt(.025,df,lower.tail=FALSE)
  int1 <- m-t*se
  int2 <- m+t*se
  if (mu < int1 | mu > int2) cnt <- cnt+1
  i1[i] <- int1
  i2[i] <- int2
  mus[i] <- mu
  means[i] <- m
}
print(cnt/numtests)
```

```
## [1] 0.04
```




Nope. So, it is independent of sample size, independent of sigma, and even independent of mu. So, in what sense are confidence intervals even about parameter estimation? They seem to more a statement about the confidence *procedure*, and here a probability statement *can* be made: In the long run, 95% of time engaging in this procedure will capture the parameter of the sampled distribution, 5% not, even if the distribution changes for each sample. And who ever asked for that? It appears that confidence intervals are, as one wag

once said, “a confidence trick”.

You will find the demos to be more precise if the variable numtests is set to a much higher value (say, 100000), but the plots become unreadable. Similarly the random ranges for n, mu, and sd can be made much larger without affecting the results, but again the plots become unreadable.