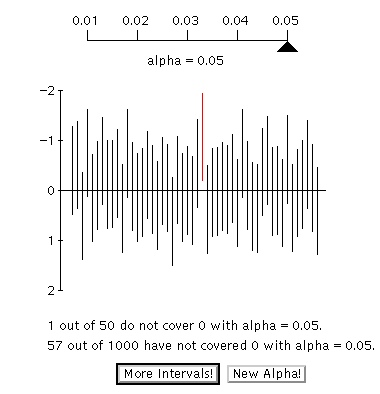
**Using simulation studies to examine confidence intervals**

Recall from class that confidence intervals give us a technique that supposedly allows us to “trap” the parameter of interest  of the time. You may recall a figure similar to the following one which illustrates this concept:



This figure shows several confidence intervals that are created when random data is generated from a distribution with mean 0. As you can see, most of the intervals contain the value 0, but not all.

Of course, it’s one thing to see a picture of confidence intervals “supposedly” in action. It’s another thing to create them yourself. In this lab, we’ll generate random data and observe what happens to the confidence intervals when we start to mess around with the settings.

**Creating the basic z confidence interval for means**

Let’s start out supposing that our data comes from a normal distribution with some mean and some standard deviation. Let’s also suppose for now that we know what the standard deviation is (an unrealistic assumption in the real world). If these facts are true, we know that a  confidence interval for the mean is:



Let’s see if we can recreate this confidence interval in R. First, let’s suppose that the mean of our data is 10 and the standard deviation is 5. Then, we’ll generate n=100 data points from a normal distribution with this mean and standard deviation.

mu = 10

sigma = 5

n = 100

x = rnorm(n, mu, sigma)

Now, let’s create the upper and lower bounds of our confidence interval. To find the appropriate z-scores, we’ll use the qnorm command in R. This achieves the same effect as looking up the inverse value in our z-tables. For now, we’ll start with the standard 95% confidence interval, so our value for alpha will be 0.05.

alpha = 0.05

upper = mean(x) + qnorm(1-alpha/2, 0, 1)\*sigma/sqrt(n)

lower = mean(x) - qnorm(1-alpha/2, 0, 1)\*sigma/sqrt(n)

Now we can check. Is the upper bound greater than the true mean? Is the lower bound less than the true mean? If so, we’ve succeeded in trapping our parameter of interest.

Of course, only running this code once won’t tell us if our confidence interval works 95% of the time. We’ll have to run this code many times, recording our successes (or failures) each time. To simplify this process, let’s use a for loop. Every time the loop runs, we’ll check if the true mean (mu) is between the upper and lower bounds of our confidence interval. If it is, we’ll give ourselves another point.

mu = 10 # Add the initial settings.

sigma = 5

n = 100

alpha = 0.05

successes = 0

for(i in 1:1000){ # Indicate how many times the loop will run.

x = rnorm(n, mu, sigma) # Generate new data each time.

upper = mean(x) + qnorm(1-alpha/2, 0, 1)\*sigma/sqrt(n) # Find the bounds.

lower = mean(x) - qnorm(1-alpha/2, 0, 1)\*sigma/sqrt(n)

if(upper > mu && lower < mu){ # Check to see if we’ve trapped the mean.

successes = successes + 1} # If so, score another success.

}

At the end, the variable “successes” will tell us how many of the 1000 trials successfully trapped our mean.

**Your turn**

1. Run the code above. How many times did you successfully trap the mean? How many times did you expect you would successfully trap the mean?

2. Run the same code three more times. Record the number of successes from each run. Are the numbers exactly the same? Do you expect them to be the same? Why or why not?

**Changing the parameters – what happens?**

For the remainder of the lab, you will change one or more of the settings in the original code, determining what effect these parameters have on the behavior of confidence intervals. Each time predict how many times your confidence interval will succeed, record the number of actual successes, and then write a short comment about the results. After each run, return the settings to their original values before continuing on to the next part of the lab.

**Your turn**

1. Change the mean of the actual distribution to the following:

|  |  |  |
| --- | --- | --- |
| New value for mean | Predicted number of successes | Actual number of successes |
| mu = 0 |  |  |
| mu = 100 |  |  |
| mu = 1000 |  |  |
| mu = -10 |  |  |

Comment:

2. Change the standard deviation of the actual distribution to the following:

|  |  |  |
| --- | --- | --- |
| New value for stand. dev. | Predicted number of successes | Actual number of successes |
| sigma = 1 |  |  |
| sigma = 10 |  |  |
| sigma = 100 |  |  |
| sigma = 1000 |  |  |

Comment:

3. Change the sample size to the following:

|  |  |  |
| --- | --- | --- |
| New value for sample size | Predicted number of successes | Actual number of successes |
| n = 50 |  |  |
| n = 30 |  |  |
| n = 20 |  |  |
| n = 10 |  |  |

Comment:

4. Change the level of significance to the following:

|  |  |  |
| --- | --- | --- |
| New value for alpha | Predicted number of successes | Actual number of successes |
| alpha = 0.20 |  |  |
| alpha = 0.10 |  |  |
| alpha = 0.01 |  |  |
| alpha = 0.001 |  |  |

Comment:

Also, what issue is raised with the value of alpha equal to 0.001? How would you change the simulation to examine if the confidence interval is performing as advertised?

5. **Challenge!** For the final simulation study, we’re going to change the distribution from which our data is drawn. Instead of using the normal distribution, we’ll use the exponential distribution – a distribution which is often used to describe the amount of time between events. For our simulation, we’ll use an exponential distribution with mean 1. One interesting property of the exponential distribution is that the mean is always equal to the standard deviation. Thus, we only need to specify one parameter value when we generate our random data. For the exponential distribution, this parameter is defined as 1/mean. Thus, we’ll need to make a few changes to the code.

First, use the following code:

mu = 1

sigma = 1

n = 100

x = rexp(n, 1/mu)

Create a histogram of x. Compare this histogram to the normal distribution.

Now, substitute the three lines of code regarding mu, sigma, and x into the original code. We’ll now repeat our study on the effect of sample size using the exponential distribution as opposed to the normal distribution.

Change the sample size to the following:

|  |  |  |
| --- | --- | --- |
| New value for sample size | Predicted number of successes | Actual number of successes |
| n = 50 |  |  |
| n = 30 |  |  |
| n = 20 |  |  |
| n = 10 |  |  |

Comment on the results. Why does the confidence interval appear to work as advertised in some cases and not in others? What theorem justifies these results?