**Simulation studies for confidence intervals with the t-distribution**

Earlier, we did a computer lab which explored how confidence intervals for the mean perform when we change different parameters. One assumption that we made is that we knew what the standard deviation of the distribution was when we created the confidence interval. Typically, this knowledge is unavailable to us when we create a confidence interval. Instead, we have only the sample standard deviation to work with. How does this change our results?

Let’s begin by repeating the simulation from the earlier lab. This time, instead of using the true standard deviation to create our confidence interval, we’ll use the sample standard deviation. Thus, our confidence interval is:



**Your turn:**

1. Why is this confidence interval incorrect?

2. Under what conditions would this confidence interval be approximately correct? Why?

Let’s redo the simulation from before, drawing data from a normal distribution with mean equal to 10 and standard deviation equal to 5. The only change in the code is that the upper and lower bounds are defined using the sample standard deviation.

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)\*sd(x)/sqrt(n) # Find the bounds.

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

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

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

}

**Your turn:**

Run the code above, changing the value of the sample size. Predict the number of times you successfully trap the mean in each case and compare it to the actual number of times you trap the mean. What do you observe?

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

Comment:

This time, we’ll properly use the t-distribution to create our confidence interval. Recall that a  confidence interval for the mean when the standard deviation is unknown is:



To make this change to our simulation code, we only need to change the code that looks up the z-score to make it look up the t-score. Remember that to find the proper t-score, we need two pieces of information: the level of significance and the degrees of freedom. Once we have those two pieces, we can tell R to find the correct value using the command **qt**. Since the degrees of freedom are equal to one less than the sample size, the correct code is:

qt(1 – alpha/2, n-1)

The full code is:

mu = 10

sigma = 5

n = 100

alpha = 0.05

successes = 0

for(i in 1:1000){

x = rnorm(n, mu, sigma)

upper = mean(x) + qt(1-alpha/2, n-1)\*sd(x)/sqrt(n)

lower = mean(x) - qt(1-alpha/2, n-1)\*sd(x)/sqrt(n)

if(upper > mu && lower < mu){

successes = successes + 1}

}

**Your turn**

We’ll repeat the studies we did for the confidence intervals using z this time with t. What happens this time?

1. 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: (Be sure to compare these results to the ones on the previous page.)

2. 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:

3. 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:

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:

5. **Challenge!** As before, we’re going to change the distribution from which the data is drawn from normal to exponential with mean 1. Make the following three changes to the original code.

mu = 1

sigma = 1

x = rexp(n, 1/mu)

Now explore the effect of sample size. 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. What requirement of the t-distribution is being broken in this case?

6. **Extra challenge!** Instead of using the exponential distribution, use the Poisson distribution to generate the data. (This distribution is used to count the number of events that occur over a period of time.) This distribution has the property that its mean and *variance* are always the same.

First, use the following code to generate some random data from a Poisson distribution with mean 10.

mu = 10

sigma = sqrt(10) # Note that this line isn’t necessary for the code.

n=100

x = rpois(n, mu)

Create a histogram of the data. Comment on the shape of the histogram.

Next, use these lines of code to change the original simulation to generate data from a Poisson distribution instead of a normal distribution and repeat the exercise in problem 5.

|  |  |  |
| --- | --- | --- |
| 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. What requirement of the t-distribution is being broken in this case? Does it seem to matter as much in this case? Why?