Psych 100A Spring 2019: Week 8 Slides

Amanda Montoya

May 21, 2019

```r
NLSdata <- read.csv("http://bit.ly/NLSdata", header = TRUE)
```

Learning Outcomes Today

- Apply the concept of a confidence interval to b_1
- Decribe pros and cons of different methods of computing a confidence interval
- Describe similarities across methods of computing confidence intervals

Does living with your romantic partner predict sleep?

Remember WAY WAY back! Week 4 / Week 5 when we were thinking about the two group model.

I asked if you if you think that living with your romantic partner predicts how much people sleep. We were able to explore the model quite a bit, but now we can learn even more.

Cohab looked like a pretty okay predictor of sleep (not great not terrible).

Could the effect we saw have been due to sampling variability? How big is the effect of cohabitation on sleep **in the population**?

"It's not that I don't love you – I'm just not ready for us to start living together."

Remembering our Model

When there are two groups, we can use a general linear model to fit the data, which will predict the group mean for individuals in each group (i.e., $\bar{Y}_{cohab}$ or $\bar{Y}_{nocohab}$ depending on which group someone is in)

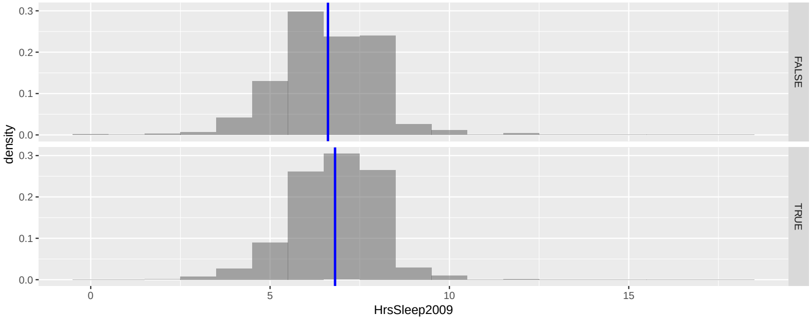
```
SleepCohabStats <- favstats(HrsSleep2009~Cohab2009, data = NLSdata)
SleepCohabStats
```

```
##   Cohab2009 min Q1 median Q3 max     mean       sd    n missing
## 1     FALSE   0  6      7  8  12 6.613682 1.361868 1491       0
## 2      TRUE   0  6      7  8  18 6.811928 1.267820 2733       0
```

Explanatory Variable: Two groups

When there are two groups, we can use the mean from each group as the prediction

```
##   Cohab2009 min Q1 median Q3 max     mean       sd    n missing
## 1     FALSE   0  6      7  8  12 6.613682 1.361868 1491       0
## 2      TRUE   0  6      7  8  18 6.811928 1.267820 2733       0
```



Fitting a linear model

We can fit a linear model using Cohab2009 as a predictor using R. The coefficients correspond to our general linear model notation

$$\hat{Y} = b_0 + b_1 X_i$$

b_0: Intercept b_1: Coefficient for X_i

$$\hat{Y} = 6.614 + 0.198 X_i$$

```r
Cohab.model <- lm(HrsSleep2009~Cohab2009, data = NLSdata)
Cohab.model
```

```
##
## Call:
## lm(formula = HrsSleep2009 ~ Cohab2009, data = NLSdata)
##
## Coefficients:
##   (Intercept)  Cohab2009TRUE
##        6.6137         0.1982
```

Predicting Y

X_i is an indicator which says whether or not someone lives with their partner.

When someone does not live with their partner: $X_i = 0$

$$\hat{Y} = b_0 + b_1 X_i$$

$$\hat{Y} = 6.614 + 0.198 \times 0 = 6.614$$

The predicted Y for someone who does not live with their partner is 6.614.

When someone does live with their partner: $X_i = 1$

$$\hat{Y} = b_0 + b_1 X_i$$

$$\hat{Y} = 6.614 + 0.198 \times 1 = 6.614 + 0.198 = 6.812$$

The predicted Y for someone who does live with their partner is 6.6812.

In general, b_1 will always be the change in predicted Y_i with a one unit increase in X_i.

How do we interpret b_1 in this situation?

Examining explained variability

```r
supernova(Cohab.model)
```

```
## Analysis of Variance Table (Type III SS)
## Model: HrsSleep2009 ~ Cohab2009
##
##                                SS   df     MS      F    PRE     p
## ----- ----------------- -------- ---- ------ ------ ------ -----
## Model (error reduced) |   37.914    1 37.914 22.373 0.0053 .0000
## Error (from model)    | 7154.812 4222  1.695
## ----- ----------------- -------- ---- ------ ------ ------ -----
## Total (empty model)   | 7192.726 4223  1.703
```

Cohabitation explains about .5% of variability in sleep. Explained variability isn't very high.

Variance explained per degree of freedom seems very large (remember that F > 2 is pretty big).

Cohabitation seems to ok (but simple) model of hours of sleep.

Calculating a CI for b_1

We've learned a few different ways to compute a confidence interval.

We'll try each of them and discuss as we go.

Ultimately, each of the methods lead to very similar answers, so most of the time it doesn't matter which one you use.

Figure 2:

Why so many ways?

Each method for creating a confidence interval helps you understand what a confidence interval is, how it's calculated, and why it's useful **in a different way**.

- Simulation
- Normal Distribution
- T-distribution
- Resampling/Bootstrapping

Many students learn the process, but don't understand the "why" or the "how".

Sometimes the results differ, and we may want to understand why.

Additionally, in more advanced methods we can't rely on the mathematical models, and we need resampling or simulation approaches.

Learning many approaches makes you more flexible.

The Goal

Confidence intervals tell us a range of population values for which our data seems likely.

What population value are we estimating with b_1?

- μ
- b_0
- μ_1
- β_1

Additionally, we can be 95% confident that the confidence intervals captures the population value.

What does that mean again?

The Goal

β_1 is the difference in hours of sleep between those who do and don't live together, **if we could survey the entire population of the United States.**

Confidence intervals tell us a range of population values for which our data seems likely.

Cohab Specific: We found a sample difference in Hours Sleep by cohabitation group. For what population group differences (i.e., β_1) would finding a sample difference like ours be likely?

Additionally, we can be 95% confident that the confidence intervals captures the population value.

Cohab specific: We can be 95% confident that the population difference in sleep is contained in the range *Lowerbound - UpperBound*.

Simulation: How

Steps for simulation:

- IF. . . .my population is normally distributed characteristics similar to my sample characteristics
- Sample many many samples from the population (M)
- For each sample calculate the estimate of interest
- Arrange estimates
- Select the ($M \times 0.025$)th and ($M \times 0.975$)th elements of the arranged estimates to create confidence interval

Simulation: Example

- IF. . . .my population is normally distributed with a mean equal to my sample mean and standard deviation equal to my sample standard deviation

```r
#What proportion of people in my data live with their partner?
tally(~Cohab2009, data = NLSdata, format = "proportion")
```

```
## Cohab2009
##     TRUE    FALSE
## 0.647017 0.352983
```

```r
#Create cohabitation in the population
pop<-data.frame(Cohab2009=c(rep(TRUE,times=0.647017*1000000),
                                   rep(FALSE,times=0.352983*1000000)))
#Check that population cohab looks like sample cohab
tally(~Cohab2009, data = pop, format = "proportion")
```

```
## Cohab2009
##     TRUE    FALSE
## 0.647017 0.352983
```

Simulation: Example

- IF. . . .my population is normally distributed with a mean equal to my sample mean and standard deviation equal to my sample standard deviation

Making Hours of Sleep: the the GLM we use an equation to describe the relationship between Hours of Sleep and Cohabitation

$$Y_i = b_0 + b_1 X_i + e_i$$

Based on our sample we have some estimates of these things:

$$HoursSleep_i = 6.6137 + 0.1982 * X_i + e_i$$

Typically we assume that e_i $N(0, \sigma^2)$ and we estimated as 1.695.

```
#Y_i                  =   b0   +  b1    X_i                    +  e_i
pop$HrsSleep2009<-6.6137+0.1982*pop$Cohab2009+rnorm(1000000,0,sqrt(1.69
```

Checking our work

```r
favstats(HrsSleep2009~Cohab2009, data = pop)
```

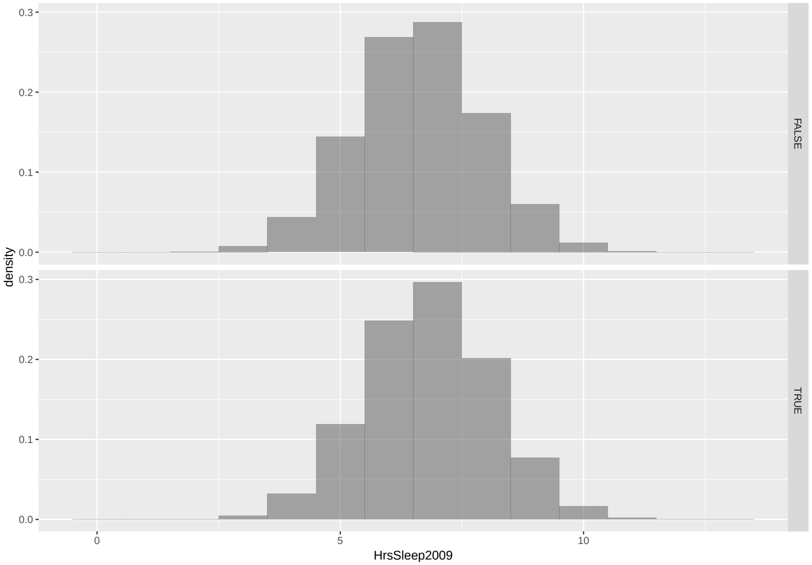
```
##   Cohab2009       min       Q1   median       Q3      max     mean
## 1     FALSE 0.0130996 5.734662 6.613303 7.489076 12.19904 6.614165
## 2      TRUE 0.6824249 5.935721 6.810965 7.687947 12.86679 6.810597
##         sd      n missing
## 1 1.302060 352983       0
## 2 1.298919 647017       0
```

```r
lm(HrsSleep2009~Cohab2009, data = pop)
```

```
## 
## Call:
## lm(formula = HrsSleep2009 ~ Cohab2009, data = pop)
## 
## Coefficients:
##   (Intercept)  Cohab2009TRUE
##        6.6142         0.1964
```

Checking our work

```
gf_dhistogram(~HrsSleep2009, data = pop, binwidth = 1)%>%
gf_facet_grid(pop$Cohab2009 ~ .)
```

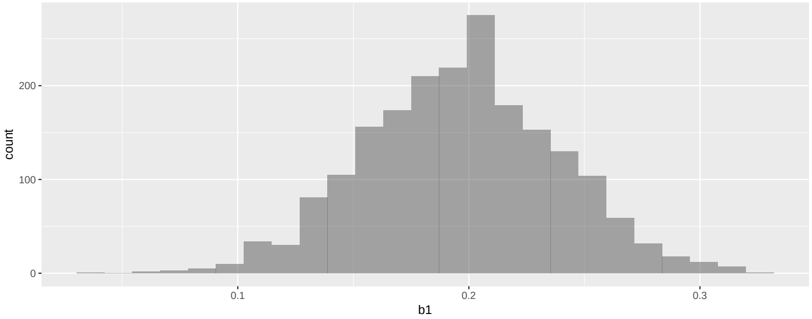


Simulation: Example

- Sample many many samples from the population (M)
- For each sample calculate the estimate of interest

Why am I using sample(pop, 4224)?

```
#           many many, calculate estimates        sample from pop
Sim.SDob1<-do(2000)*b1(HrsSleep2009~Cohab2009,data=sample(pop,4224))
gf_histogram(~b1, data = Sim.SDob1)
```



Simulation: Example

- Arrange estimates
- Select the $M \times 0.025$th and $M \times 0.975$th elements of the arranged estimates to create confidence interval

```
Sim.SDob1 <- arrange(Sim.SDob1, desc(b1))
sim.ul <- Sim.SDob1$b1[2000*0.025]
sim.ul
```

```
## [1] 0.2787108
```

```
sim.ll <- Sim.SDob1$b1[2000*0.975]
sim.ll
```

```
## [1] 0.1133958
```

The range of means for which our b_1 estimate 0.1982462 seems likely (using a 95% limit) is $\beta_1 =$ (0.1133958 - 0.2787108).

Simulation: Pros and Cons

Pros:

- Have complete control over elements of the population (could change anything we want)
- Performs well (gets the right answer a lot of the time)

Cons:

- Creating a population can be pretty intense
- Rely on assumption about the shape of the distribution of the errors
- Interval not guaranteed to be symmetric
- Answer will be a little different every time when we rerun code

Bootstrapping: How

If we're just going to use our sample to make the population, why can't we just treat our sample like the population, sampling from our sample with replacement.

Steps for simulation:

- Sample many many samples from the my sample with replacement (M)
- For each sample calculate the estimate of interest
- Arrange estimates
- Select the ($M \times 0.025$)th and ($M \times 0.975$)th elements of the arranged estimates to create confidence interval

Bootstrapping

- Sample many many samples from the my sample with replacement (M)
- For each sample calculate the estimate of interest

```
bootdat <- resample(NLSdata, 4224)
favstats(HrsSleep2009~Cohab2009, data = bootdat)
```

```
##   Cohab2009 min Q1 median Q3 max     mean       sd    n missing
## 1     FALSE   2  6      7  8  12 6.612356 1.333019 1473       0
## 2      TRUE   0  6      7  8  18 6.829153 1.335352 2751       0
```

```
b1(HrsSleep2009 ~ Cohab2009, data = bootdat)
```

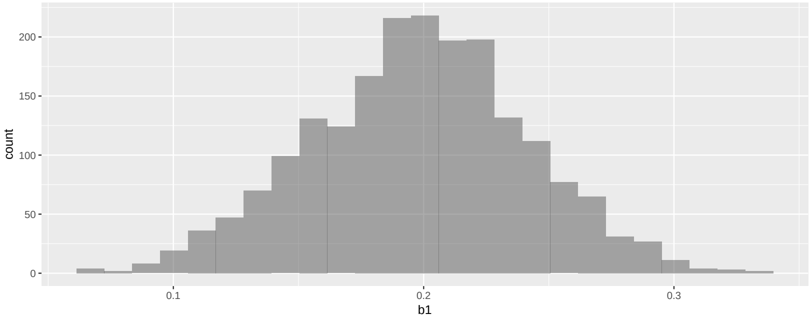
```
## [1] 0.2167973
```

```
boot.SDob1 <- do(2000)*b1(HrsSleep2009 ~ Cohab2009, data = resample(NLS
```

Bootstrapping:How

Visualizing the sampling distribution

```r
gf_histogram(~b1, data = boot.SDob1)
```



Bootstrapping:How

- Arrange estimates
- Select the ($M \times 0.025$)th and ($M \times 0.975$)th elements of the arranged estimates to create confidence interval

```
boot.SDob1 <- arrange(boot.SDob1, desc(b1))
boot.ul <- boot.SDob1$b1[2000*0.025]
boot.ul
```

```
## [1] 0.2831777
```

```
boot.ll <- boot.SDob1$b1[2000*0.975]
boot.ll
```

```
## [1] 0.1125432
```

The range of means for which our b_1 estimate 0.1982462 seems likely (using a 95% limit) is $\beta_1 =$ 0.1125432 - 0.2831777.

Bootstrapping: Pros and Cons

Pros:

- Using the sample as representation of population makes simulation easier
- No assumptions about the shape of distribution of the errors
- Performs well (gets the right answer a lot of the time)

Cons:

- Interval not guaranteed to be symmetric
- Answer will be a little different every time when we rerun code

Normal Distribution: How

- Use a normal distribution with mean $= b_1$ and sd $= s_{b_1}/\sqrt{n}$ from sample to approximate sampling distribution
- Calculate points which denote the top and bottom 2.5% (xpnorm)
- Combine points at confidence interval.

Normal Distribution: How (Thinking about the Mean)

- Use a normal distribution with mean $= \bar{Y}$ and sd $= s/\sqrt{n}$ from sample to approximate sampling distribution

Notice that we skip making a population distribution and go straight to the sampling distribution

We can trust that the sampling distribution is normal because of the **Central Limit Theorem**.

What was that again?

Which of the following is NOT a characteristic of the Central Limit Theorem as applied to a sampling distribution of the mean?

- Sampling distribution will become more normal in shape as sample size gets larger
- The mean of the sampling distribution ($\mu_{\bar{Y}}$ is equal to the mean of the population (μ)
- The standard deviation of the sampling distribution $\sigma_{\bar{Y}}$ is equal to standard deviation of the population divided by the square root of N $\sigma/\sqrt{n}$.
- Regardless of the sample size, the sampling distribution is always guaranteed to be normally distributed.

Normal Distribution: How

- Use a normal distribution with mean $= \bar{Y}$ and sd $= s/\sqrt{n}$ from sample to approximate sampling distribution

How does this work for b_1? - mean $= b_1$ from sample - sd $=$??

For the sampling distribution of the mean, s is used because it's the average error from the model. The standard deviation for b_1 is similar, but adjust for the additional complexity of the model ($df = n - 2$).

$$s_{b_1} = \sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n-2}}$$

Normal Distribution: How

- Use a normal distribution with mean $= b_1$ and sd $= s_{b_1}$ from sample to approximate sampling distribution

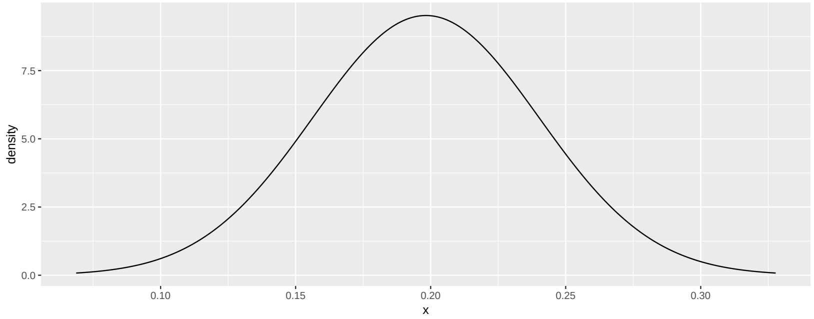
```
#Use standard error from lm summary to get s_{b_1}
summary(Cohab.model)
```

```
##
## Call:
## lm(formula = HrsSleep2009 ~ Cohab2009, data = NLSdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.8119 -0.8119  0.1881  1.1881 11.1881
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    6.61368    0.03371  196.17  < 2e-16 ***
## Cohab2009TRUE  0.19825    0.04191    4.73 2.32e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.302 on 4222 degrees of freedom
## Multiple R-squared:  0.005271,   Adjusted R-squared:  0.005036
```

Normal Distribution: How

- Use a normal distribution with mean $= b_1$ and sd $= s_{b_1}$ from sample to approximate sampling distribution

```r
gf_dist("norm", params = list(0.19825, 0.04191))
```



Normal Distribution: How

- Calculate points which denote the top and bottom 2.5% (qnorm)

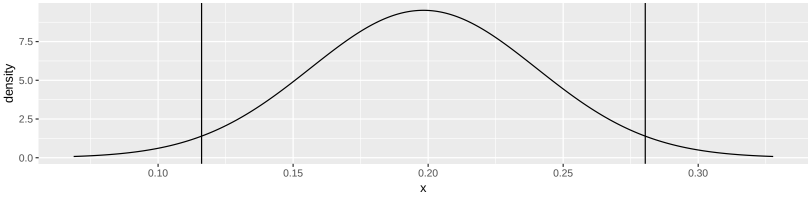
```
qnorm(0.025, 0.19825, 0.04191)
```

```
## [1] 0.1161079
```

```
qnorm(0.975, 0.19825, 0.04191)
```

```
## [1] 0.2803921
```

```
gf_dist("norm", params = list(0.19825, 0.04191))%>%
  gf_vline(xintercept = qnorm(0.025, 0.19825, 0.04191))%>%
  gf_vline(xintercept = qnorm(0.975, 0.19825, 0.04191))
```



Normal Distribution: How

- Combine points at confidence interval.

Our observed slope 0.1982462 is considered likely for population means within the range of 0.1161079 to 0.2803921.

We are 95% confident that the difference in hours of sleep between cohabitating and noncohabitating partners is between 0.1161079 and 0.2803921.

Notice that zero is not contained in this interval. So we're very confident that there is some difference between cohabitating individuals, but how much is still inexact.

Calculating Margin of Error

For a confidence interval, if we can find the "Margin of Error" (MOE) we can then just calculate a confidence interval as:

$$b_1 \pm MOE = b_1 \pm s_{b_1} \times z_{critical}$$

We can backwards engineer the MOE:

```r
normMOE <- b1(Cohab.model) -  qnorm(0.025, 0.19825, 0.04191)
normMOE
```

```
## [1] 0.08213828
```

Or we can calculating it by just not accounting for the mean in the normal distribution (consider a normal dsitribution centered around 0) but with the estimated SD

```r
qnorm(0.975, 0, 0.04191)
```

```
## [1] 0.08214209
```

Normal Distribution: How

All steps in one!

```
confint.default(Cohab.model)
```

```
##                   2.5 %    97.5 %
## (Intercept)   6.5476053 6.6797589
## Cohab2009TRUE 0.1160992 0.2803931
```

Work with your neighbors, how would you interpret the (Intercept) confidence interval?

Normal Distribution: Pros and Cons

Pros:

- VERY EASY! (thanks to R)
- Interval guaranteed to be symmetric
- Answer is always the same

Cons:

- Central Limit Theory doesn't always work for small samples (no exact definition of how small is too small)
- Doesn't take into account that the standard error is an estimate (not a population value)

T-distribution: How

- Use a t-distribution with mean $= 0$ and $df = n - 2$ from sample to approximate sampling distribution (centered around 0)
- Calculate points which denote the top and bottom 2.5% (qt), to get $t_{critical}$
- Combine estimate and MOE to get a confidence interval

T-statistic

t-distribution takes into account the fact that the standard deviation of the sampling distribution σ_{b_1} is not known, but rather estimated.

t-distribution is not a distribution of means, but rather a distribution of t-values (this part is ignored in the book). Anything where we take an estimate and divide by it's estimated standard error is called a t-statistic

$$t = b_1/s_{b_1}$$

```r
summary(Cohab.model)
```

```
##
## Call:
## lm(formula = HrsSleep2009 ~ Cohab2009, data = NLSdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.8119 -0.8119  0.1881  1.1881 11.1881
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    6.61368    0.03371  196.17  < 2e-16 ***
## Cohab2009TRUE  0.19825    0.04191    4.73 2.32e-06 ***
```

T-distribution

Since the estimate of the standard error depends on the number of degrees of freedom, the t-distribution also depends on the degrees of freedom.

Fewer degrees of freedom (smaller sample size) means more variability in distribution

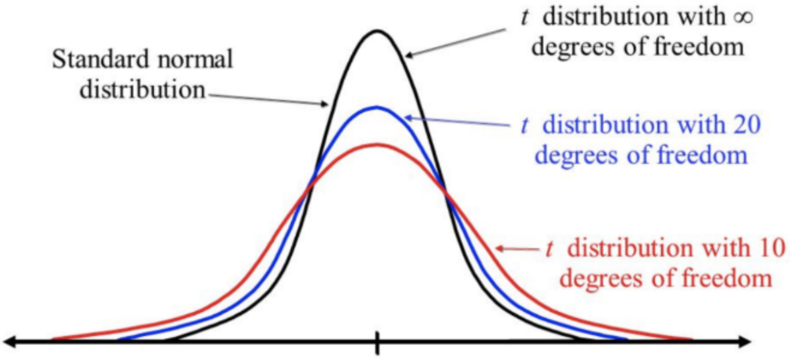
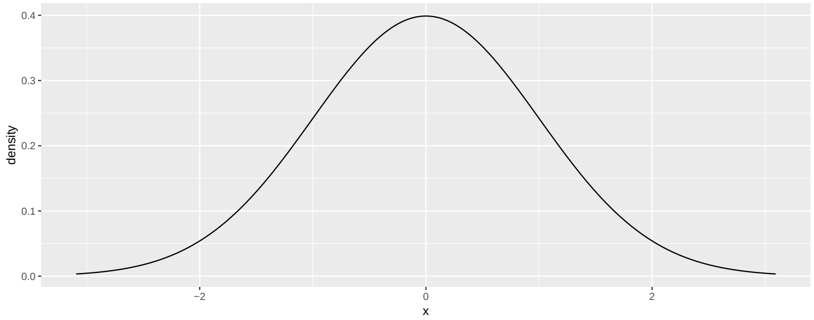


Figure 3: t-distribution

T-distribution: How

- Use a t-distribution with mean $= 0$ and $df = n - 2$ from sample to approximate sampling distribution (centered around 0)

```
gf_dist("t", params = (4224-2))
```



T-distribution: How

- Calculate points which denote the top and bottom 2.5% (qt), to get $t_{critical}$

$$b_1/s_{b_1} \pm t_{critical}$$

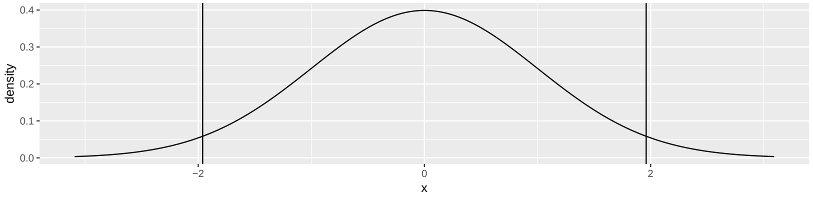
```
qt(0.025, 4224-2)
```

```
## [1] -1.960526
```

```
qt(0.975, 4224-2)
```

```
## [1] 1.960526
```

```
gf_dist("t", params = (4224-2))%>%
  gf_vline(xintercept = qt(0.025, 4224-2))%>%
  gf_vline(xintercept = qt(0.975, 4224-2))
```



T-distribution: How

- Combine estimate and MOE to get a confidence interval

$$b_1/s_{b_1} \pm t_{critical}$$

$$b_1 \pm s_{b_1} \times t_{critical}$$

```
b1(Cohab.model) + 0.04191*qt(0.975, 4224-2)
```

```
## [1] 0.2804118
```

```
b1(Cohab.model) - 0.04191*qt(0.975, 4224-2)
```

```
## [1] 0.1160805
```

```
confint(Cohab.model)
```

```
##                   2.5 %    97.5 %
## (Intercept)   6.5475863 6.6797779
## Cohab2009TRUE 0.1160757 0.2804167
```

T-distribution: Pros and Cons

Pros:

- VERY EASY! (thanks to R)
- Interval guaranteed to be symmetric
- Answer is always the same
- Takes into account that the standard error is an estimate (not a population value)

Cons:

- Central Limit Theory doesn't always work for small samples (no exact definition of how small is too small)

All intervals together

```
#simulation
c(sim.ll, sim.ul)
```

```
## [1] 0.1133958 0.2787108
```

```
#bootstrap
c(boot.ll, boot.ul)
```

```
## [1] 0.1125432 0.2831777
```

```
#normal distr
confint.default(Cohab.model)
```

```
##                   2.5 %    97.5 %
## (Intercept)   6.5476053 6.6797589
## Cohab2009TRUE 0.1160992 0.2803931
```

```
#t-dist
confint(Cohab.model)
```

```
##                   2.5 %    97.5 %
## (Intercept)   6.5475863 6.6797779
## Cohab2009TRUE 0.1160757 0.2804167
```

Making Conclusions

Regardless of the method, we would conclude that $\beta_1 = 0$ is not among the population means for which our data are likely.

This means, that we can safely conclude our data came from a positive β_1 (i.e., We believe that there is a difference between sleep for cohabitating people (cohab $>$ no cohab))