Lab 01: Chi squared, t, and F distributions

Your Name(s) Go(es) Here

Evaluation

This lab will count towards your participation score in this class. I'm not going to grade it for accuracy, I will just check GitHub to see if you have made a good attempt at it. I will post solutions, but it will be helpful to you if you try to work through the lab before checking the solutions.

Places where the line starts with #### 1. (or similar) are places where I'm asking you to do something.

You should definitely feel free to work through this with someone else and just submit one version of the lab. In that case, just put both of your names at the top of the document. You can add your partner as a collaborator on the GitHub repository.

Goals

Our goals are to:

1. Get practice working with R Markdown documents.
2. Get practice with the basic functions for working with random variables and creating plots using ggplot2. Examples of this R functionality are on the first three pages of the "Common Probability Distributions" handout, also linked to on the resources page of the course website.
3. Verify some results about the relationships between χ^2, t, and F distributions through simulation

Loading Packages

I'm loading some R packages with functions that you might use in this assignment.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

Example code to start with

Here is some code (adapted from the "Common Probability Distributions" handout) that:

1. sets a seed for random number generation. This means that every time the following code is run, the same "random" numbers will result
2. generates a sample of size 100 from a Normal(0, 1) distribution and stores it as a variable `x` in a data frame called `sample_data`. Each *row* of this data frame has one of the samples in it.
3. multiplies each of those values of x by 0.5 and adds 2; the results are added to the data frame in a new variable called w
4. creates a histogram of the histogram of the sample data as well as the corresponding normal distribution pdf.
5. evaluates the pdf and cdf of the normal distribution at the value `w = 2`

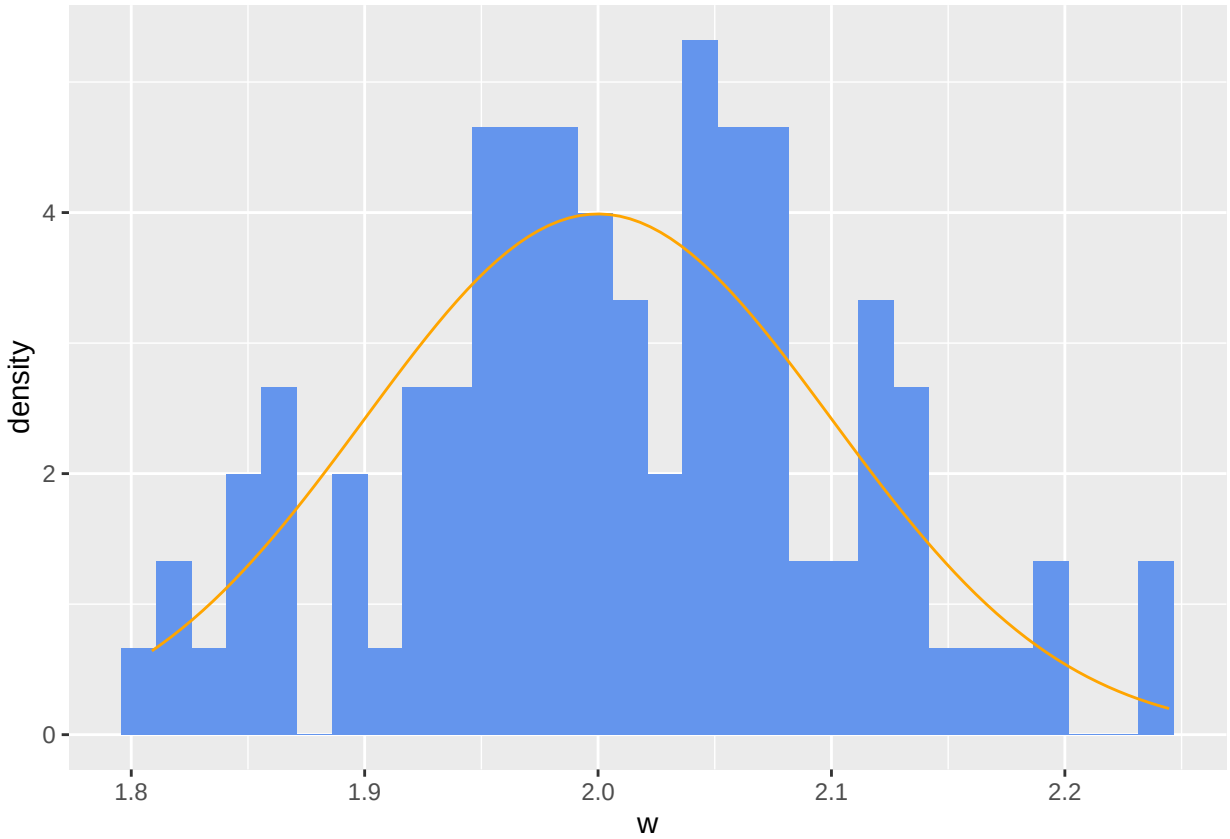
```r
# Step 1: set a seed for random number generation
set.seed(77926)

# Step 2: generate 100 random samples from a Normal(0, 1) distribution, save in a data frame
sample_data <- data.frame(
  x = rnorm(n = 100, mean = 0, sd = 1)
)

# Step 3: add a second variable to the data frame obtained by transforming x.
# We can see the result of this by looking at the data frame with the View() function
sample_data <- sample_data %>%
  mutate(
    w = x * 0.1 + 2
  )
View(sample_data)

# Step 4: make a "density" histogram, overlaid with the pdf of the transformed normal distribution
ggplot(data = sample_data, mapping = aes(x = w)) +
  geom_histogram(mapping = aes(y = ..density..), fill = "cornflowerblue") +
  stat_function(fun = dnorm, args = list(mean = 2, sd = 0.1), color = "orange")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
# Step 5: evaluate the pdf and cdf at a single point, w = 2
dnorm(2, mean = 2, sd = 0.1)
```

```
## [1] 3.989423
```

```r
pnorm(2, mean = 2, sd = 0.1)
```

```
## [1] 0.5
```

Stuff for you to do

1. Create a data frame with 5 variables in it called `x1`, `x2`, `x3`, `x4`, and `x5`. To start with, draw a sample of size $n = 1$ for each of these, from a Normal(0, 1) distribution.

```
set.seed(77926)

sample_data <- data.frame(
  x1 = rnorm(n = 1000, mean = 0, sd = 1),
  x2 = rnorm(n = 1000, mean = 0, sd = 1),
  x3 = rnorm(n = 1000, mean = 0, sd = 1),
  x4 = rnorm(n = 1000, mean = 0, sd = 1),
  x5 = rnorm(n = 1000, mean = 0, sd = 1)
)
```

2. Create a new variable in your data frame called `u`, calculated as `x1^2 + x2^2 + x3^2`. What is the distribution of the random variable U we have sampled from?

```
sample_data <- sample_data %>%
  mutate(
    u = x1^2 + x2^2 + x3^2
  )
```

U follows a chi squared distribution with 3 degrees of freedom.

3. Create a new variable in your data frame called `v`, calculated as `x4^2 + x5^2`. What is the distribution of the random variable V we have sampled from?

```
sample_data <- sample_data %>%
  mutate(
    v = x4^2 + x5^2
  )
```

V follows a chi squared distribution with 2 degrees of freedom.

4. Create a new variable in your data frame calculated as `x4 / sqrt(u/3)`. What is the distribution of this random variable? Give the variable an appropriate name to reflect the distribution it comes from.

```
sample_data <- sample_data %>%
  mutate(
    t = x4 / sqrt(u/3)
  )
```

This random variable follows a t distribution with 3 degrees of freedom.

5. Create a new variable in your data frame calculated as `(v/2) / (u/3)`. What is the distribution of this random variable? Give the variable an appropriate name to reflect the distribution it comes from.

```
sample_data <- sample_data %>%
  mutate(
    F = (v/2) / (u/3)
  )
```

This random variable follows an F distribution with degrees of freedom 2 and 3.

6. At this point your data frame should have one row and 9 columns. Verify that that's correct. Then, go back to step 1 and change your sample size from $n = 1$ to $n = 1000$, and re-run all of your code chunks above in order. Now instead of having one sample from a Normal(0, 1) distribution for `x1`, you'll have 1000 samples. Verify that your data frame now has 1000 rows and 9 columns.

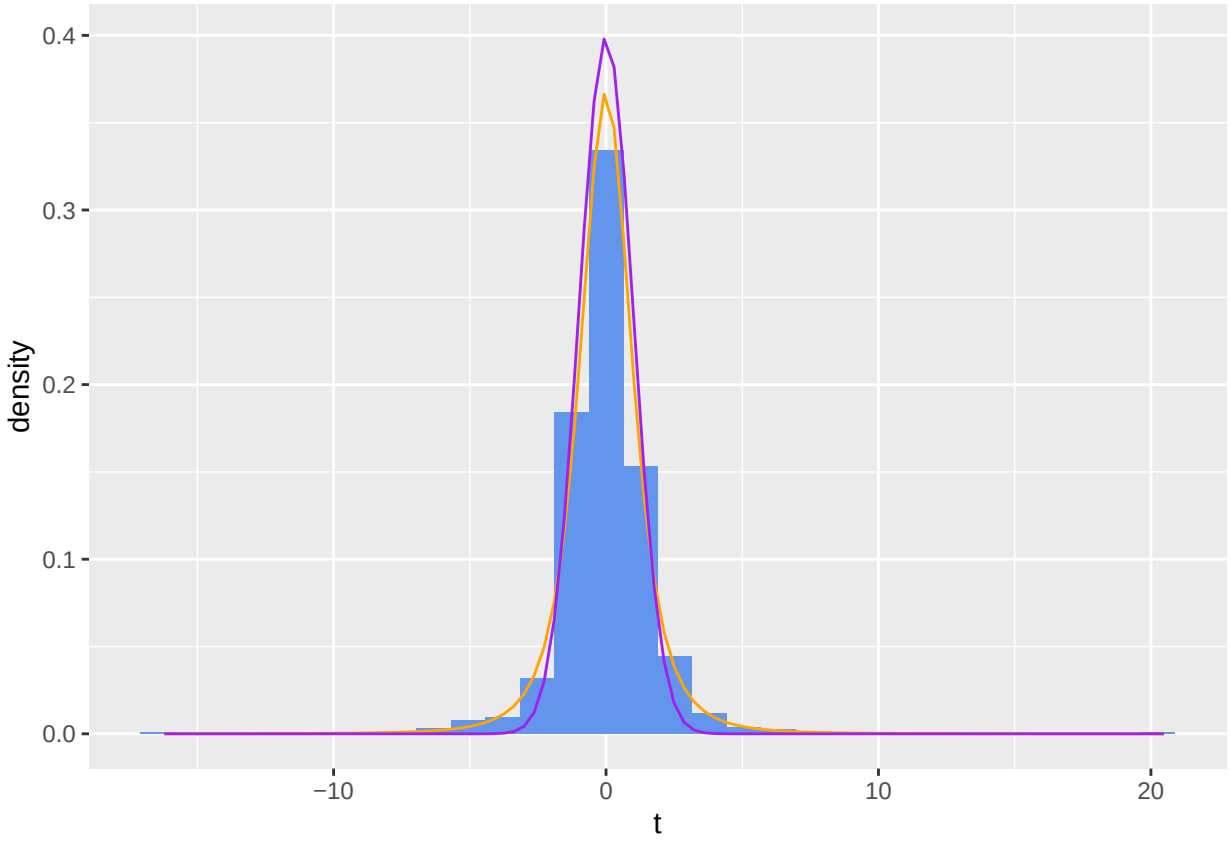
```r
View(sample_data) # results display in RStudio, but not when knitting
dim(sample_data)
```

```
## [1] 1000    9
```

7. Create a density histogram of the 1000 sample values for the variable you created in step 4. In a second layer, overlay a plot of the pdf for the distribution you chose. In a third layer, overlay a plot of the pdf of a Normal(0, 1) distribution using a different color.

```r
ggplot(data = sample_data, mapping = aes(x = t)) +
  geom_histogram(mapping = aes(y = ..density..), fill = "cornflowerblue") +
  stat_function(fun = dt, args = list(df = 3), color = "orange") +
  stat_function(fun = dnorm, args = list(mean = 0, sd = 1), color = "purple")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



8. Using the pt function, calculate the probability that a random variable that follows a t distribution with 3 degrees of freedom will be larger than 2. Repeat this calculation, but for a Normal(0, 1) distribution.

```r
1 - pt(2, df = 3)
```

```
## [1] 0.06966298
```

```r
1 - pnorm(2, mean = 0, sd = 1)
```

```
## [1] 0.02275013
```

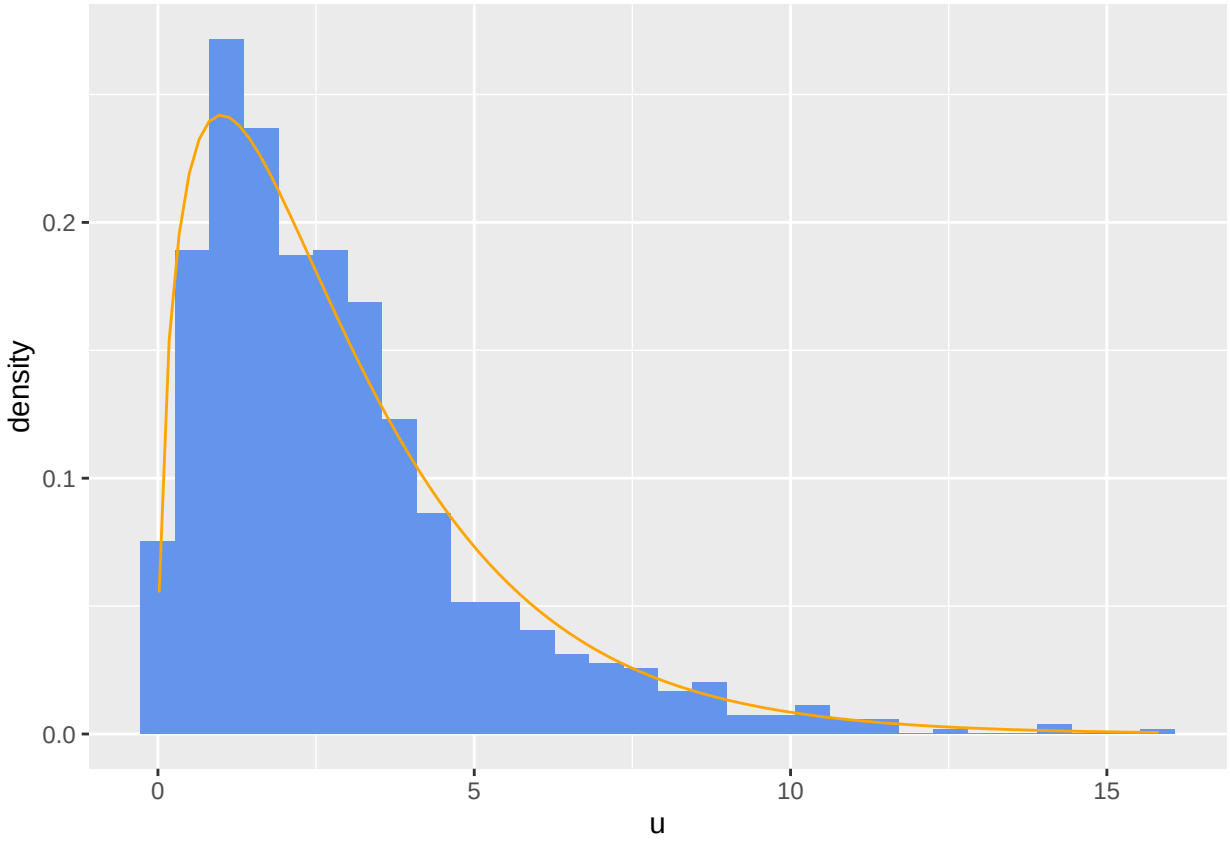
9. Based on your histogram from part 7 and your result from part 8, sum up the similarities and differences between a t distribution and a Normal(0, 1) distribution in a sentence or two.

Both are roughly bell shaped distributions centered at 0. The t distribution has wider tails; there's a higher probability of getting a relatively large or relatively small number from a t distribution than from a normal distribution.

10. (If time) Create density histograms of the 1000 sample values for the variables you created in steps 2 and 5, along with plots of the pdfs for those random variables.

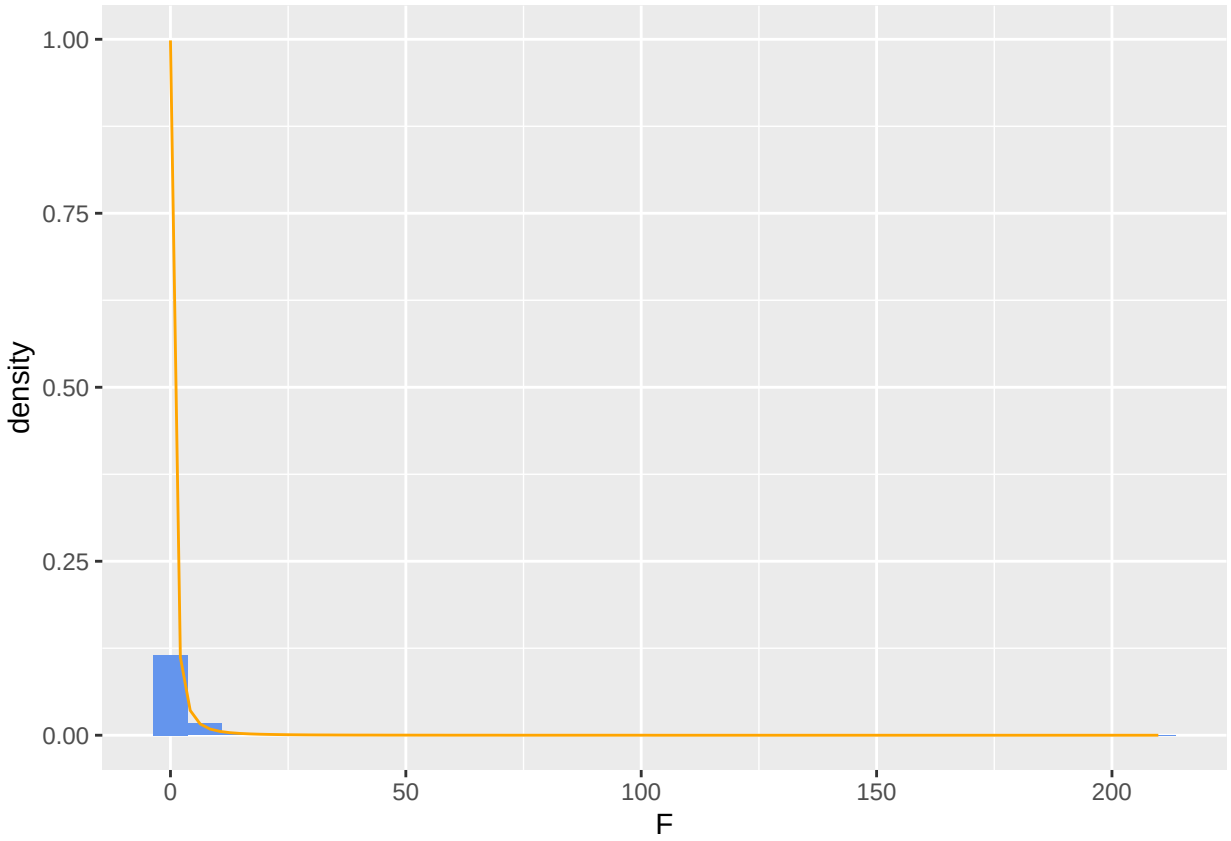
```
ggplot(data = sample_data, mapping = aes(x = u)) +
  geom_histogram(mapping = aes(y = ..density..), fill = "cornflowerblue") +
  stat_function(fun = dchisq, args = list(df = 3), color = "orange")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
ggplot(data = sample_data, mapping = aes(x = F)) +
  geom_histogram(mapping = aes(y = ..density..), fill = "cornflowerblue") +
  stat_function(fun = df, args = list(df1 = 2, df2 = 3), color = "orange")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


1.00
0.75
0.50
0.25
0.00
density
0
50
100
150
200
F