Working with Probability Distributions

Andrew Zieffler
January 07, 2021

Preparation

In this set of notes, you will learn about common continuous probability distributions. We will not be using a specific dataset in these notes.

```
# Load libraries
library(tidyverse)
```

Normal (Gaussian) Distribution

The probability density function (PDF) of a normal distribution is mathematically defined as:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$

for $-\infty \leq x \leq \infty$.

To calculate the probability density, we need three pieces of information: (1) the x-value for which we want to determine the probability density, (2) the mean (μ) of the normal distribution x is a member of, and (3) the standard deviation (σ) of the normal distribution x is a member of. Then, we can compute the probability density ($p(x)$) for a particular x value by using the equation.

As an example, consider a normal distribution with a mean of 50, and a standard deviation of 10. The probability density for $x = 65$ can be found using,

$$p(65) = \frac{1}{10\sqrt{2\pi}} \exp\left[-\frac{(65-50)^2}{2 \times 10^2}\right]$$
$$= 0.01295176$$

Using R, we can carry out the computation,

```
# Compute the probability density of x=65 in N(50,10)
(1 / (10 * sqrt(2 * pi))) * exp(-(225) / 200)
```

```
[1] 0.01295176
```

There is also a more direct way to compute this using the `dnorm()` function. This function computes the density of `x` from a normal distribution with a specified `mean` and `sd`.

```
# Compute the probability density of x=65 in N(50,10)
dnorm(x = 65, mean = 50, sd = 10)
```

```
[1] 0.01295176
```

Symbolically, we might write

$$P\Big(x = 65 \mid \mathcal{N}(50, 10)\Big) = 0.01295176$$

which is read, "the probability density of $x = 65$ GIVEN the normal distribution having a mean of 50 and standard deviation of 10 is equal to 0.013."

Note that the probability density for a value is not only a function of *x*, but also depends on the mean and standard deviation of the normal distribution. For example, the probability density of $x = 65$ in the normal distribution having a mean of 30 and standard deviation of 20 is a different value than the probability density we found earlier.

```
# Compute the probability density of x=65 in N(30,20)
dnorm(x = 65, mean = 30, sd = 20)
```

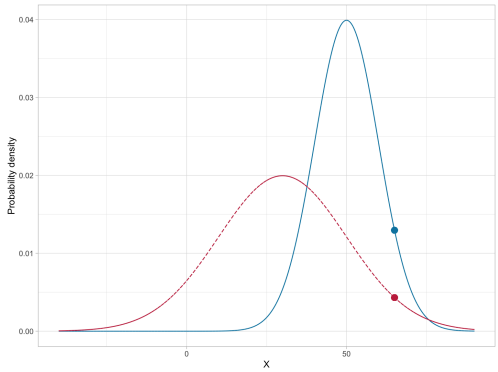
```
[1] 0.004313866
```

Here,

$$P\Big(x = 65 \mid \mathcal{N}(30, 20)\Big) = 0.004313866$$

In general, when we think about the normal distribution, we are thinking about the mapping of each x-value from $-\infty$ to $+\infty$ to its associated probability density. Rather than list each of these mappings out, we can create a plot of these mappings. This plot gives us the familiar "bell shape." Theoretically this plot is the graphical depiction of the PDF.

```
# Create dataset
fig_01 = data.frame(
  X = seq(from = -40, to = 90, by = 0.01)
  ) %>%
  mutate(
    Y1 = dnorm(x = X, mean = 50, sd = 10),
    Y2 = dnorm(x = X, mean = 30, sd = 20)
    )

# Create plot
ggplot(data = fig_01, aes(x = X, y = Y1)) +
  geom_line(color = "#0085af", linetype = "solid") +
  geom_line(aes(y = Y2), linetype = "dashed", color = "#c62f4b") +
  xlab("X") +
  ylab("Probability density") +
  theme_light() +
  geom_point(x = 65, y = 0.01295176,  size = 3, color = "#0085af") +
  geom_point(x = 65, y = 0.004313866, size = 3, color = "#c62f4b")
```



Plot of the probability density function (PDF) for a $\mathcal{N}(50, 10)$ distribution (blue, solid line) and for a $\mathcal{N}(30, 20)$ distribution (red, dashed line). The probability density value for $x = 65$ is also displayed on both PDFs.

Of course, the PDF is different for normal distributions with different means (μ) or standard deviations (σ). This implies that there is not one normal distribution, but rather an infinite number of normal distributions, each with a different mean or standard deviation. (We refer to the normal distribution as a "family" of distributions.)

To completely define the PDF we need to specify the mean and standard deviation we are using to compute the probability densities. Specifying these values is referred to as *parameterizing the distribution*[1].

Other Useful R Functions for Working with Normal Probability Distributions

We use `dnorm()` when we want to compute the probability density associated with a particular x-value in a given normal distribution. There are three other functions that are quite useful for working with the normal probability distribution:

- `pnorm()` : To compute the probability (area under the PDF)
- `qnorm()` : To compute the x value given a particular probability
- `rnorm()` : To draw a random observation from the distribution

Each of these function also requires the arguments `mean=` and `sd=` . Below we will examine how to use each of these additional functions.

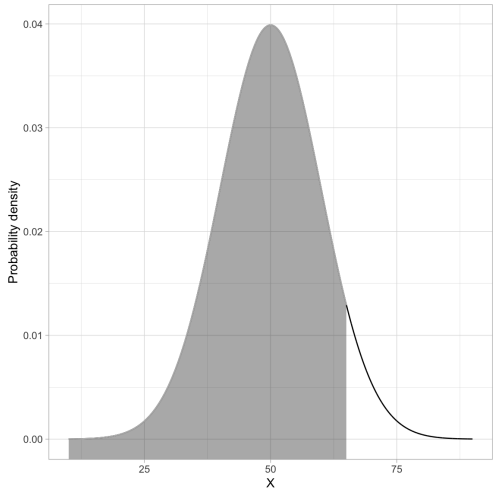
pnorm(): Computing Cumulative Probability Density

The function `pnorm()` computes the area under the PDF curve from $-\infty$ to some x-value. (Sometimes this is referred to as the cumulative probability density of x.) It is important to note that the PDF is defined such that the entire area under the curve is equal to 1. Because of this, we can also think about using area under the curve as an analog to probability in a continuous distribution.

For example, we might ask about the probability of observing an x-value that is less than or equal to 65 given it is from a $\mathcal{N}(50, 10)$ distribution. Symbolically, we want to find:

$$P\Big(x \leq 65 \mid \mathcal{N}(50, 10)\Big)$$

This is akin to finding the proportion of the area under the $\mathcal{N}(50, 10)$ PDF that is to the left of 65. The figure below shows a graphical depiction of the cumulative probability density for $x = 65$.



Plot of the probability density function (PDF) for a $\mathcal{N}(50, 10)$ distribution. The area that is shaded grey (relative to the total area under the PDF) represents the cumulative probability density for $x = 65$.

We can compute the cumulative probability density using the `pnorm()` function. The "p" stand for "probability."

```
# Find P(x<=65 | N(50,10) )
pnorm(q = 65, mean = 50, sd = 10)
```

```
[1] 0.9331928
```

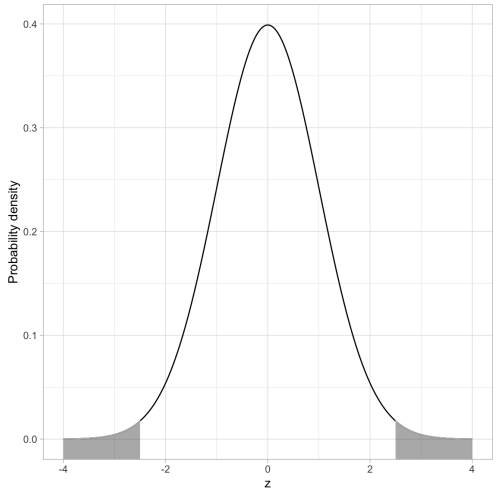
The probability of observing an x-value that is less than or equal to 65 (if it is drawn from a normal distribution with a mean of 50 and standard deviation of 10) is 0.933.

In mathematics, the area under a curve is called an *integral*. The grey-shaded area in the previous figure can also be expressed as an integral of the probability density function:

$$\int_{-\infty}^{65} p(x)dx$$

where $p(x)$ is the PDF for the normal distribution.

The most common application for finding the cumulative density is to compute a p-value. The p-value is just the area under the distribution (curve) that is AT LEAST as extreme as some observed value. For example, assume we computed a test statistic of $z = 2.5$, and were evaluating whether this was different from 0 (two-tailed test). Graphically, we want to determine the proportion of the area under the PDF that is shaded grey in the figure below.



Plot of the probability density function (PDF) for the standard normal distribution ($M = 0$, $SD = 1$). The cumulative density representing the p-value for a two-tailed test evaluating whether $\mu = 0$ using an observed mean of 2.5 is also displayed.

If the distribution of the test statistic is normally distributed, we can use `pnorm()` to compute the *p*-value. If we assume the test statistic, *z*, has been scaled to use standardized units, the standard deviation we use in `pnorm()` will be `sd=1` . The mean is based on the value being tested in the null hypothesis. In most null hypotheses, we are testing a difference from 0 (e.g., $H_0 : \mu = 0$, $H_0 : \beta = 0$), so we would use `mean=0` in the `pnorm()` function.

Remember, `pnorm()` computes the proportion of the area under the curve TO THE LEFT of a particular value. Here we will compute the area to the left of -2.5 and then double it to produce the actual *p*-value. (We can double it because the normal distribution is symmetric so the area to the left of -2.5 is the same as the area to the right of $+2.5$.)

```
# Compute the p-value based on z=2.5
2 * pnorm(q = -2.5, mean = 0, sd = 1)
```

```
[1] 0.01241933
```

The probability of observing a statistic at least as extreme as 2.5, assuming the null hypothesis is true, is 0.012. This is evidence against the null hypothesis since the data are inconsistent with the assumed hypothesis.

qnorm(): Computing Quantiles

The `qnorm()` function is essentially the inverse of the `pnorm()` function. The `pnorm()` function computes the cumulative probability GIVEN a particular quantile (x-value). The `qnorm()` function computes the quantile GIVEN a cumulative probability. For example, in the $\mathcal{N}(50, 10)$ distribution, half of the area under the PDF is below the x-value (quantile) of 50.

To use the `qnorm()` function to give the x-value (quantile) that defines the lower 0.5 of the area under the $\mathcal{N}(50, 10)$ PDF, the syntax would be:

```
# Find the quantile that has a cumulative density of 0.5 in the N(50, 10)
        distribution
qnorm(p = 0.5, mean = 50, sd = 10)
```

```
[1] 50
```

rnorm(): Generating Random Observations

The `rnorm()` function can be used to generate random observations drawn from a specified normal distribution. Aside from the `mean=` and `sd=` arguments, we also need to specify the number of observations to generate by including the argument `n=`. For example, to generate 15 observations drawn from a $\mathcal{N}(50, 10)$ distribution we would use the following syntax:

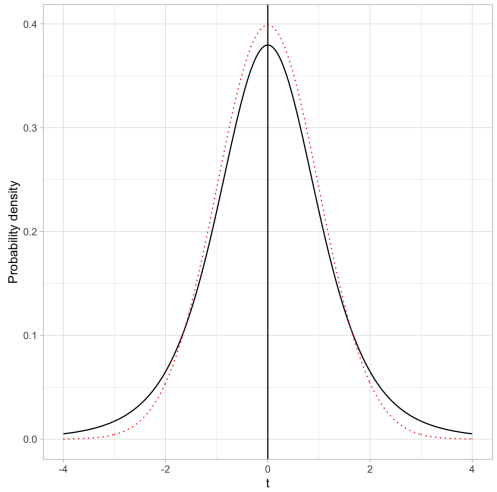
```
# Generate 15 observations from N(50,10)
set.seed(100)
rnorm(n = 15, mean = 50, sd = 10)
```

```
 [1] 44.97808 51.31531 49.21083 58.86785 51.16971 53.18630 44.18209 57.14533
 [9] 41.74741 46.40138 50.89886 50.96274 47.98366 57.39840 51.23380
```

The `set.seed()` function sets the state of the random number generator used in R so that the results are reproducible. If you don't use `set.seed()` you will get a different set of observations each time you run `rnorm()`. Here we set the starting seed to 100, but you can set this to any integer you want.

Student's *t*-Distribution

The PDF of Student's t-distribution looks similar to the PDF for a standard normal distribution. In the figure below, Student's t-distribution is depicted with a solid, black line and the standard normal distribution ($M = 0, SD = 1$) is depicted with a dotted, red line.

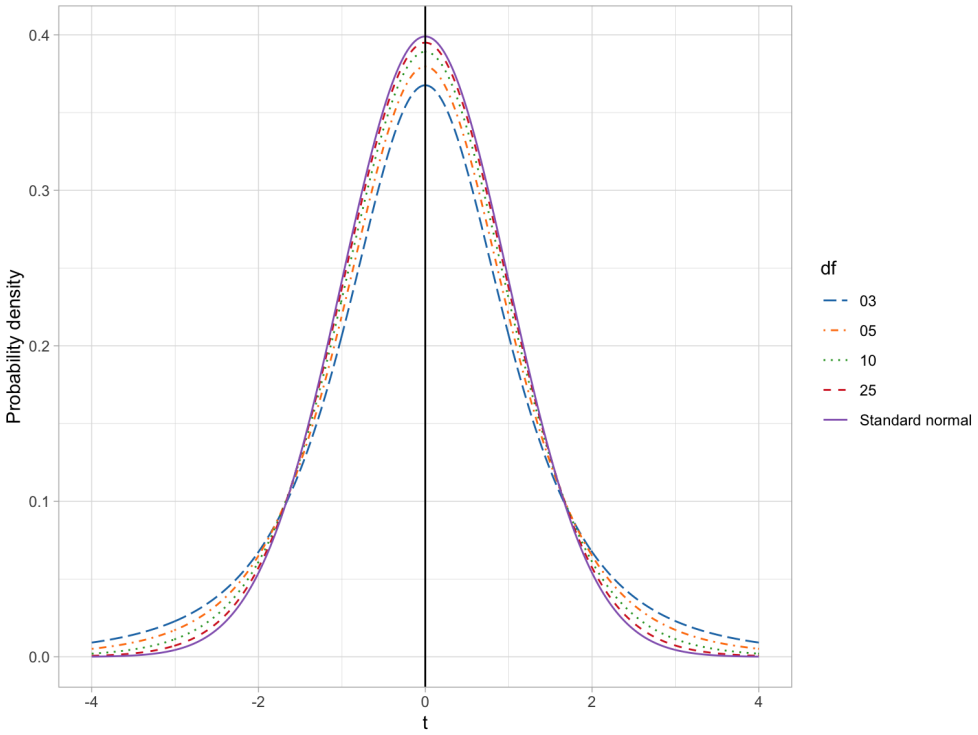


Plot of the probability density function (PDF) for both the standard normal distribution (dotted, red line) and Student's $t(5)$ distribution (solid, black line).

- Both the standard normal distribution and Student's t-distribution are symmetric distributions.
- Both the standard normal distribution and Student's t-distribution have a mean (expected value) of 0.
- The standard deviation for Student's t-distribution is, however, larger than the standard deviation for the standard normal distribution ($SD > 1$). You can see this in the distribution because the tails in Student's t-distribution are fatter (more error) than the standard normal distribution.

In practice, we often use Student's t-distribution rather than the standard normal distribution in our evaluations of sample data. This is because the increased error (i.e., standard deviation) associated with Student's t-distribution better models the additional uncertainty associated with having incomplete information (i.e., a sample rather than the entire population).

Student's t-distribution also constitutes a family of distributions; there is not a single t-distribution. The specific shape (and thus probability density) is defined by a parameter called the *degrees of freedom* (df). The plot below shows the standard normal distribution (purple) and four t-distributions with varying df-values. The means and standard deviations for each of these distributions is also provided in a table.



Plot of several t-Distributions with differing degrees of freedom.

Means and standard deviations for four t-Distributions and the standard normal distribution

df	*M*	*SD*
03	0	2.00
05	0	1.50
10	0	1.22
25	0	1.08
z	0	1.00

If we compare the means and standard deviations for these distributions, we find that the mean for all the *t*-distributions is 0, same as the standard normal distribution. All *t*-distributions are unimodal and symmetric around zero. The standard deviation for every *t*-distribution is higher than the standard deviation for the standard normal distribution. Mathematically, the variance for the *t*-distribution is:

$$\sigma^2(t) = \frac{\nu}{\nu - 2}$$

where ν (the Greek letter nu) is the degrees of freedom. (Note that $\nu \geq 2$.) Examining this formula, we find that Student t-distributions with higher df values have less variation. When $\nu = +\infty$, the variance approaches 1, which is the same as the standard normal distribution.

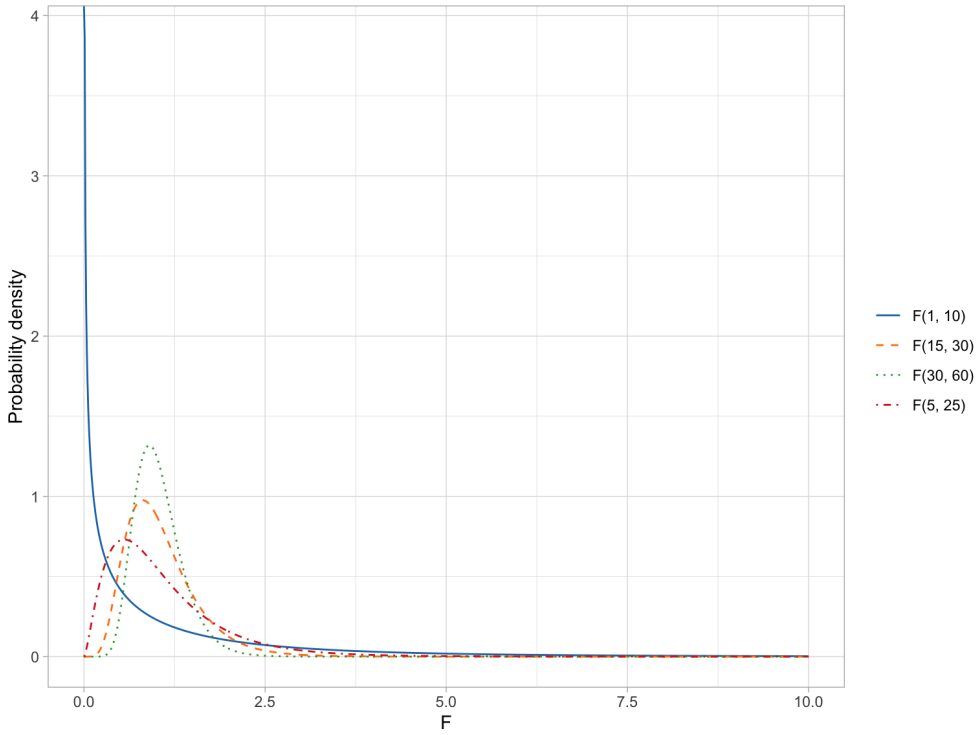
There are four primary functions for working with Student's t-distribution:

- `dt()` : To compute the probability density (point on the curve)
- `pt()` : To compute the cumulative density (area under the PDF)
- `qt()` : To compute the quantile value given a particular probability
- `rt()` : To draw a random observation from the distribution

Each of these requires the argument `df=` .

The F-distribution

The F-distribution, like the t-distribution, constitutes a family of distributions. They are positively skewed and generally have a lower-limit of 0. To parameterize an F-distribution we need two parameters, namely ν_1 and ν_2. These are both degrees of freedom. The exact shape of the F-distribution s governed by the two degrees of freedom parameters. The figure below shows several F-distributions with different degrees of freedom.



Plot of several F-Distributions with differing degrees of freedom.

The expected value (mean) and standard deviation of the F-distribution is:

$$E(F) = \frac{\nu_2}{\nu_2 - 2}$$

$$\sigma^2(F) = \frac{2\nu_2^2(\nu_1 + \nu_2 - 2)}{\nu_1(\nu_2 - 2)^2(\nu_2 - 4)}$$

where $\nu_2 > 2$ for the mean and $\nu_2 > 4$ for the variance.

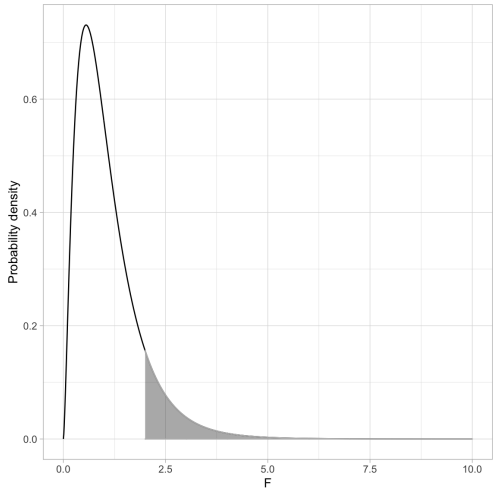
From these formulas we can see that as $\nu_2 \to +\infty$ the mean of the F-distribution approaches 1. We can also see that the variation in the F-distribution is a function of both parameters and the variance decreases as either parameter gets larger.

The means and standard deviations for our four example F-distributions are given in the table below.

Means and standard deviations for four F-distributions.

df_1	df_2	M	SD
1	10	1.250000	2.1650635
15	30	1.071429	0.5031299
30	60	1.034483	0.3348305
5	25	1.086957	0.7938008

Because there is no negative side of the distribution, when we use the F-distribution to compute a p-value, we only compute the cumulative density GREATER THAN OR EQUAL TO the value of the F-statistic. For example, the figure below shows the $F(5, 25)$-distribution and the shaded area corresponds to the p-value for an observed F-statistic of 2.



Plot of the probability density function (PDF) for the $F(5, 25)$-distribution. The cumulative density representing the p-value associated with an F-statistic of 2 is shaded in grey.

Here we can use the `pf()` function to compute the *p*-value. Remember, `pf()` computes the proportion of the area under the curve TO THE LEFT of a particular value. Here we will need to compute the area to the RIGHT of +2.

```
# Compute the p-value based on F(5,25)=2
1 - pf(q = 2, df1 = 5, df2 = 25)
```

```
[1] 0.1134803
```

The probability of observing an *F*-statistic at least as extreme as 2, assuming the null hypothesis is true, is 0.113. This is not evidence against the null hypothesis since the data are consistent with the assumed hypothesis.

Creating a PDF and Adding Shading in a ggplot

One method to create the PDF for a distribution using `ggplot()` is to create a dataset that includes a sequence of *X*-values for which you want to show the PDF and compute the probability density for each of those values. Then you can use `geom_line()` to connect those probability densities.

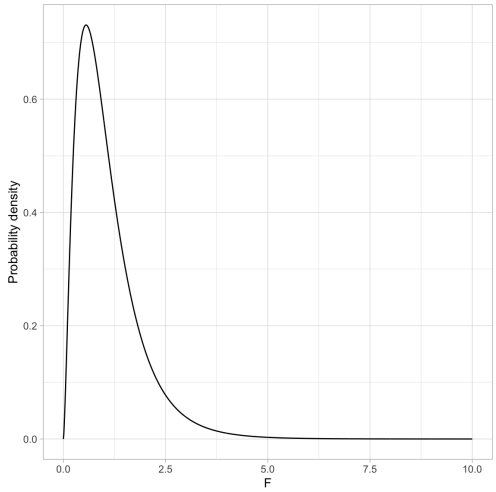
For example, say we want to create the PDF of the $F(15, 100)$-distribution. Here I will define this for F-values from 0 to 10. (These are the x-values in my plot.) Then I need to compute the probability densities for each of those values using `pf()`.

```
# Create F-value and compute probability densities
fig_01 = data.frame(
  X = seq(from = 0, to = 10, by = 0.01)
  ) %>%
  mutate(
    Y = df(x = X, df1 = 5, df2 = 25)
    )

# View data
head(fig_01)
```

```
     X           Y
1 0.00 0.000000000
2 0.01 0.008319698
3 0.02 0.022838243
4 0.03 0.040722606
5 0.04 0.060856280
6 0.05 0.082557740
```

Then we can plot the Y versus the X values and connect them using `geom_line()`.

```
ggplot(data = fig_01, aes(x = X, y = Y)) +
  geom_line() +
  xlab("F") +
  ylab("Probability density") +
  theme_light()
```



Plot of the probability density function (PDF) for the $F(5, 25)$-distribution.

To add shading under the curve we need to create a new dataset that only includes the X and Y values in the shaded region. For example to shade the area under the PDF where $F > 2$, we need to create a new dataset where the X values are greater than 2. Below I do this using `filter()` and store the data in an object called `shade_01`.

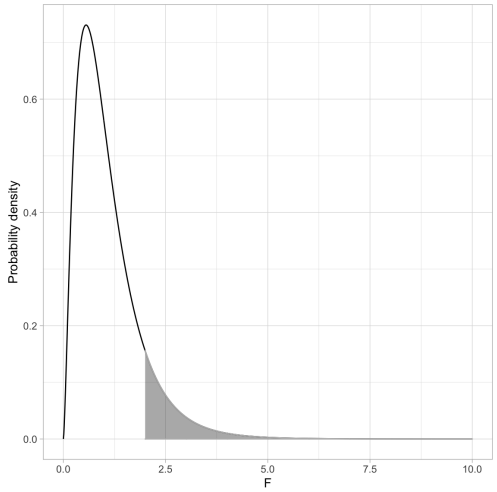
```
# Filter data included in the shaded region
shade_01 = fig_01 %>%
  filter(X >= 2)

# View data
head(shade_01)
```

```
     X         Y
1 2.00 0.1558647
2 2.01 0.1537083
3 2.02 0.1515807
4 2.03 0.1494816
5 2.04 0.1474106
6 2.05 0.1453676
```

We re-draw the PDF and then use `geom_ribbon()` to add shading. This layer requires us to define the area we want shaded. Here we want to shade from $Y = 0$ to $Y =$ the probability density for each of the X values in the shading data. To carry this out we need to define `x=`, `ymin=` and `ymax=`.

Since the X values are in a column called `x` and the probability densities are in a column called `Y` in the shaded dataa frame, we can call `x=X` and `ymax=Y` in the `aes()` of `geom_ribbon()`. The `ymin=` value of 0 is not a column in the data frame, so it is specified OUTSIDE the `aes()` function. We can then also set characteristics like color of the shading (`color=`) and transparency level (`alpha=`). Finally, to ensure that `geom_ribbon()` is shading only the region we want, we set `data=shade_01`.

```
# Create plot
ggplot(data = fig_01, aes(x = X, y = Y)) +
  geom_line() +
  xlab("F") +
  ylab("Probability density") +
  theme_light() +
  geom_ribbon(data = shade_01, ymin = 0, aes(x = X, ymax = Y),
              color = "#bbbbbb", alpha = 0.4)
```



Plot of the probability density function (PDF) for the $F(5, 25)$-distribution. The cumulative density representing the p-value associated with an F-statistic of 2 is shaded in grey.

---

1. Remember, the mean and standard deviations in the population are called "parameters."↩