Online CDS 101 *Introduction to Computational and Data Sciences*

Data distributions

Probability mass functions

Download and load the dataset

You can follow along by downloading and loading the dataset by placing the following *setup* code block at the top of a R Markdown file.

~~~
```{r setup, include = FALSE}
# Load required packages
library(tidyverse)
# Load datasets
county <- read_rds(url("http://data.cds101.com/county_complete.rds"))
```
~~~

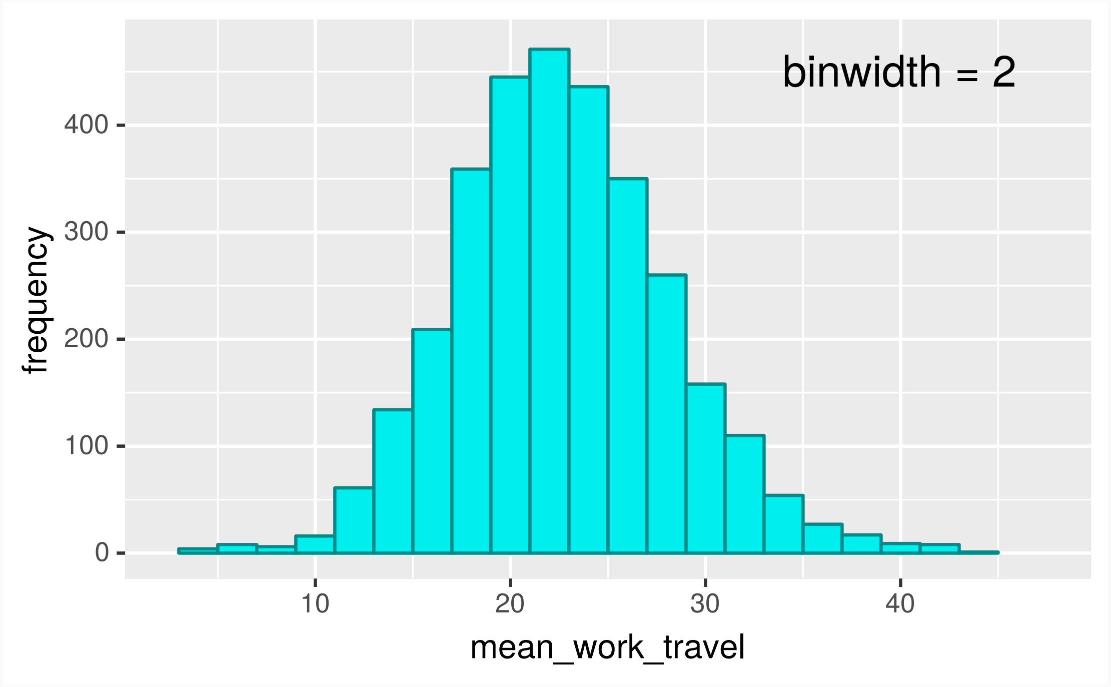
Data distributions

We've already learned that histograms (`geom_histogram( )`) are a convenient way to represent numerical data in a single column (variable)

mean_work_travel
25.1
25.8
23.8
28.3
33.2
28.1
25.1
...

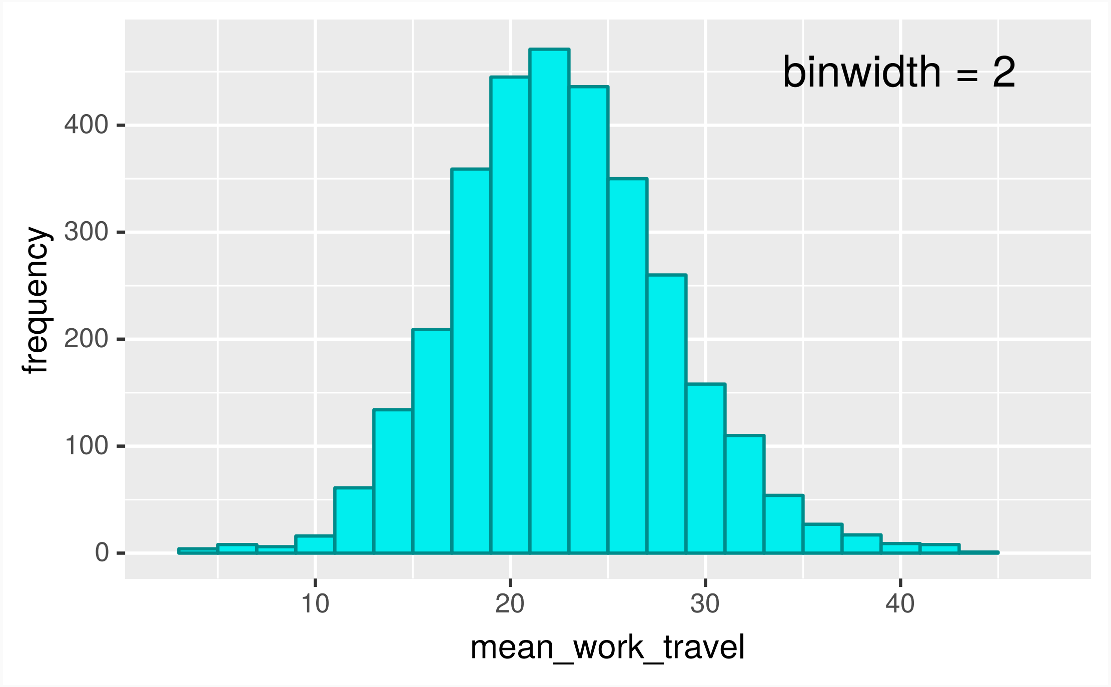
Data distributions

We've already learned that histograms (`geom_histogram()`) are a convenient way to represent numerical data in a single column (variable)



Data distributions

We've already learned that histograms (`geom_histogram()`) are a convenient way to represent numerical data in a single column (variable)

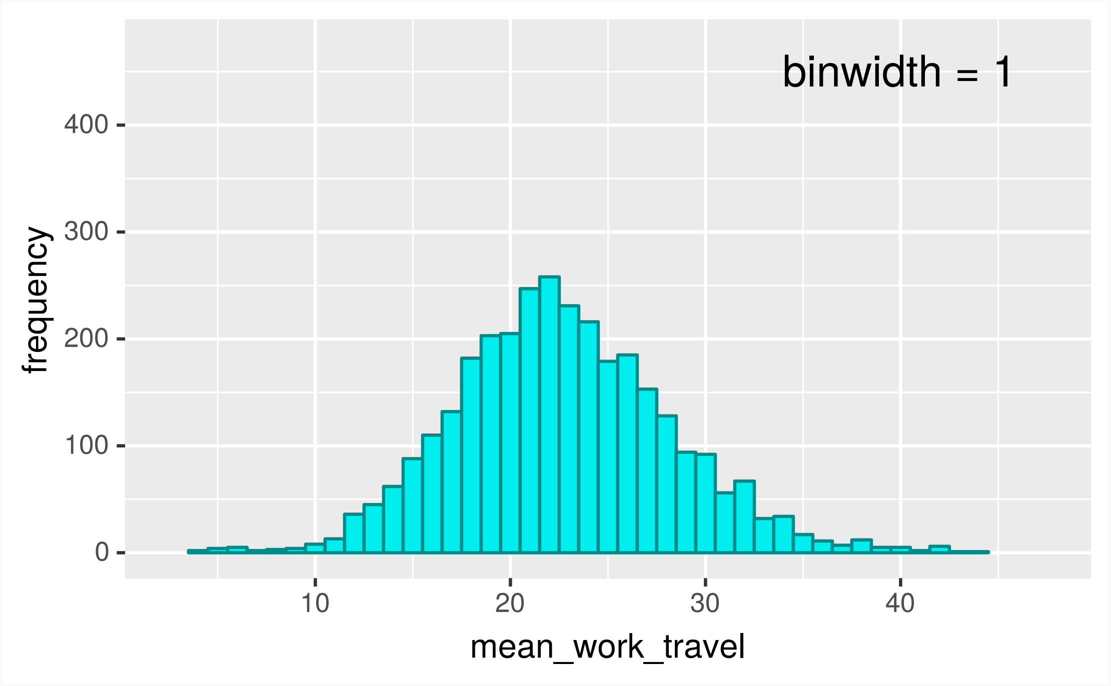


A histogram represents the **frequency** that values show up for a given variable

`binwidth` changes the "buckets" for the data, impacting the frequency heights.

Data distributions

We've already learned that histograms (`geom_histogram()`) are a convenient way to represent numerical data in a single column (variable).



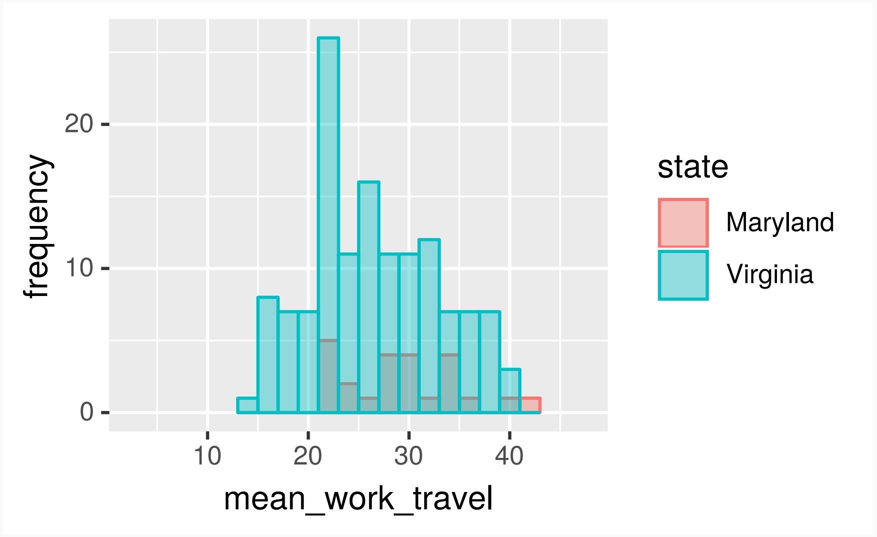
A histogram represents the **frequency** that values show up for a given variable

`binwidth` changes the "buckets" for the data, impacting the frequency heights.

Comparing distributions with unequal observations

So far, we've largely skipped over the question of how to compare distributions with varying numbers of observations.

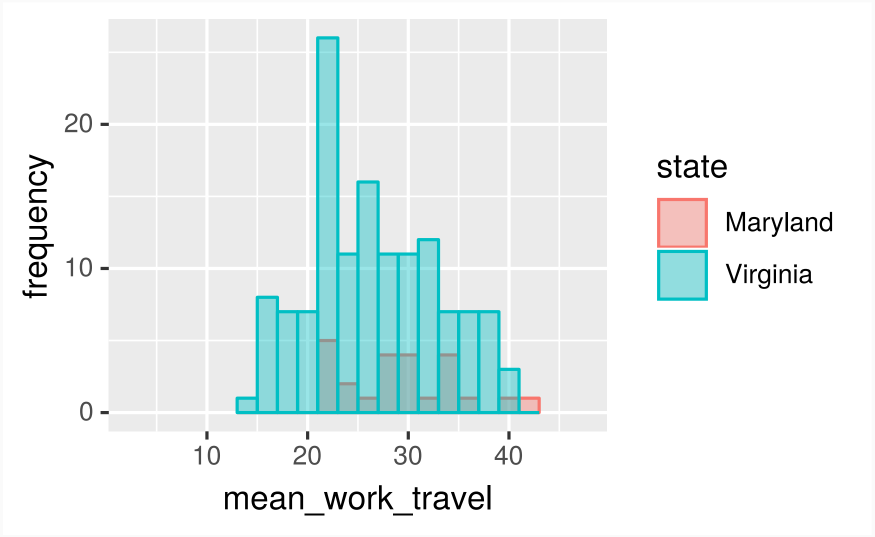
In our current example of average times to travel to work, we can group the data by state and compare Virginia to Maryland.



Comparing distributions with unequal observations

So far, we've largely skipped over the question of how to compare distributions with varying numbers of observations.

In our current example of average times to travel to work, we can group the data by state and compare Virginia to Maryland.

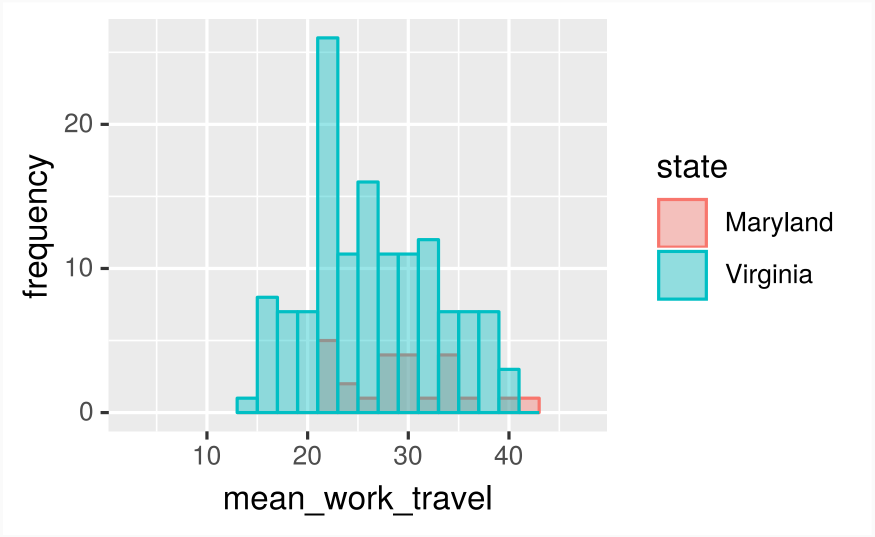


In which state am I more likely to have a 30 minute commute?

Comparing distributions with unequal observations

So far, we've largely skipped over the question of how to compare distributions with varying numbers of observations.

In our current example of average times to travel to work, we can group the data by state and compare Virginia to Maryland.



In the dataset, Virginia has 134 counties compared to Maryland's 24 counties.

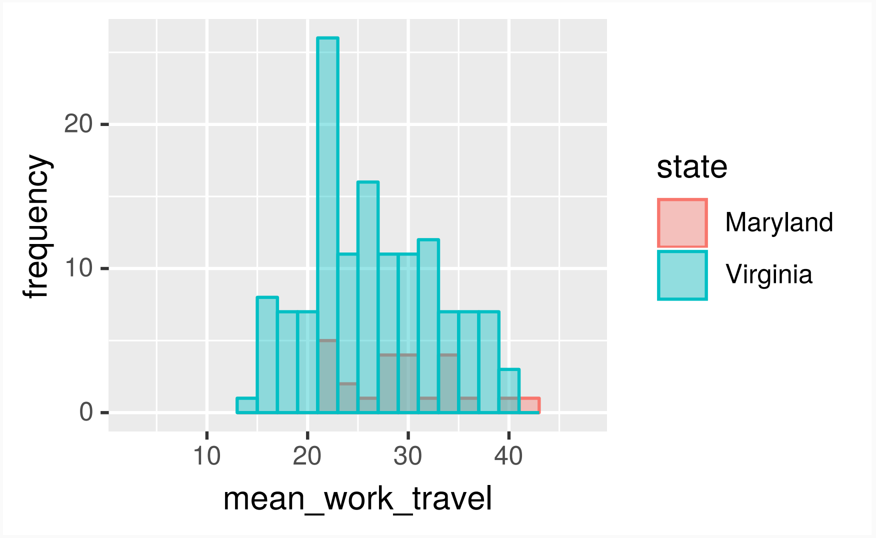
We need to **normalize** the frequency counts.

From frequency to probability

Normalization is straightforward, just divide the frequency count in each "bucket" by the total number of observations in the histogram.

If you group by categories, that you should divide by the number of observations in each group.

To normalize the histograms from the prior example, we need to divide the Virginia frequencies by 134 and the Maryland frequencies by 24.

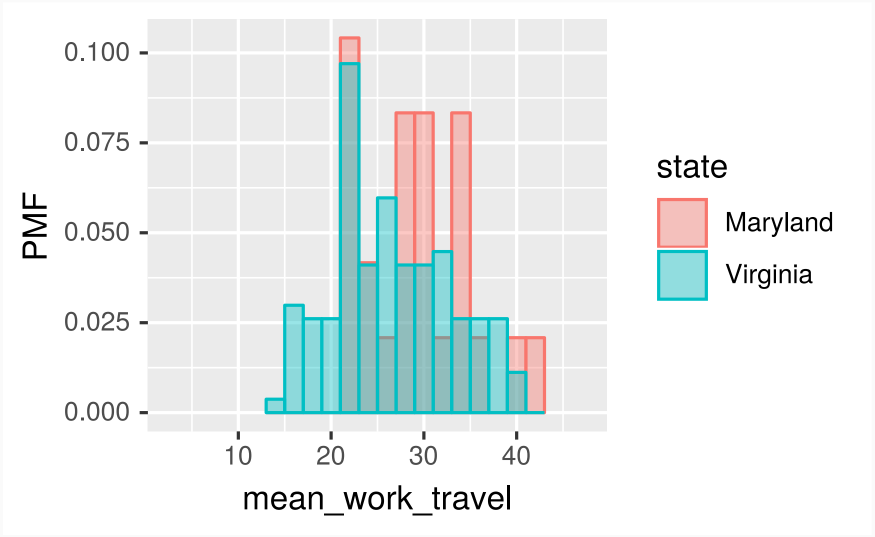


From frequency to probability

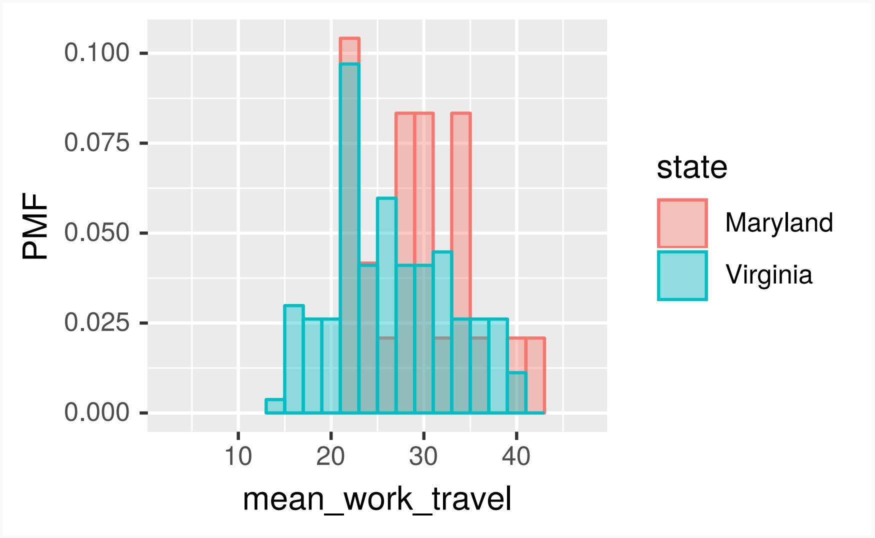
Normalization is straightforward, just divide the frequency count in each "bucket" by the total number of observations in the histogram.

If you group by categories, that you should divide by the number of observations in each group.

To normalize the histograms from the prior example, we need to divide the Virginia frequencies by 134 and the Maryland frequencies by 24.



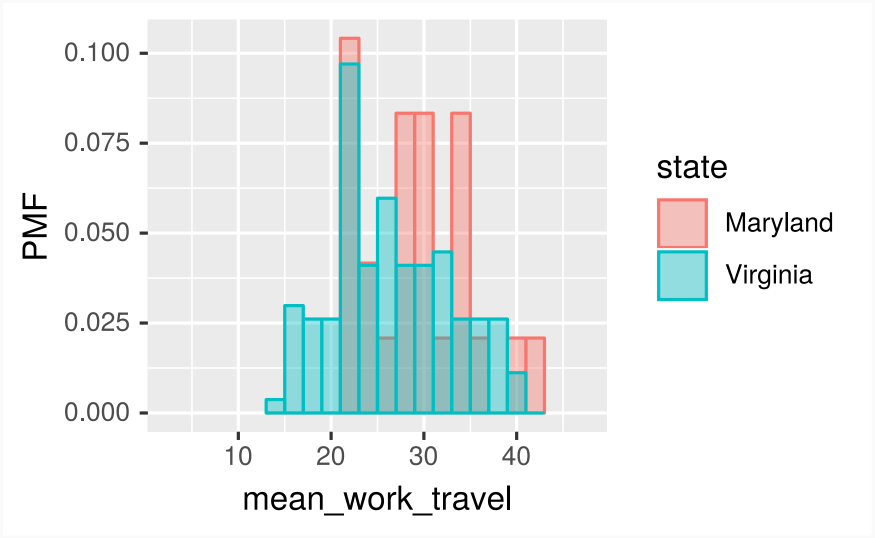
Probability mass function (PMF)



Just like a histogram, except that the bar heights reflect **probabilities** instead of **frequency counts**.

Allows for a meaningful comparison of distributions with different numbers of observations.

Probability mass function (PMF)

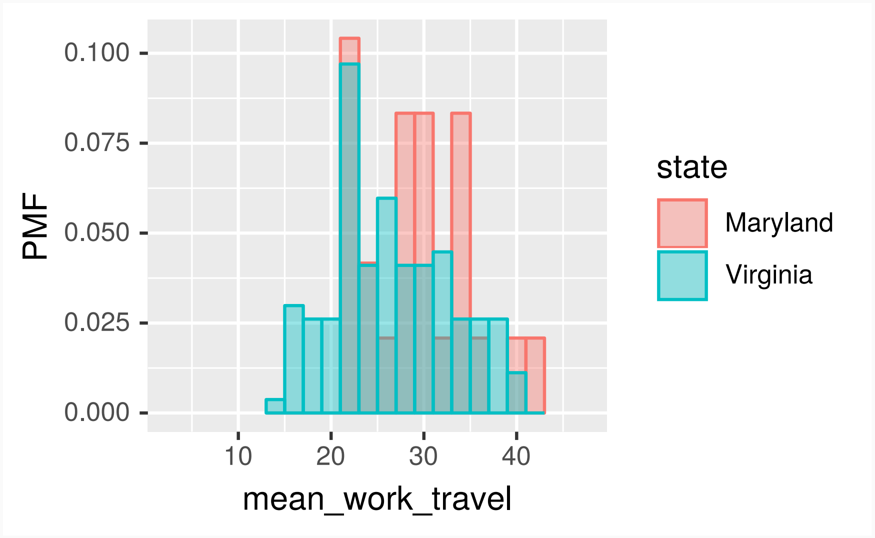


Just like a histogram, except that the bar heights reflect **probabilities** instead of **frequency counts**.

Allows for a meaningful comparison of distributions with different numbers of observations.

In which state am I more likely to have a 30 minute commute?

Probability mass function (PMF)



Just like a histogram, except that the bar heights reflect **probabilities** instead of **frequency counts**.

Allows for a meaningful comparison of distributions with different numbers of observations.

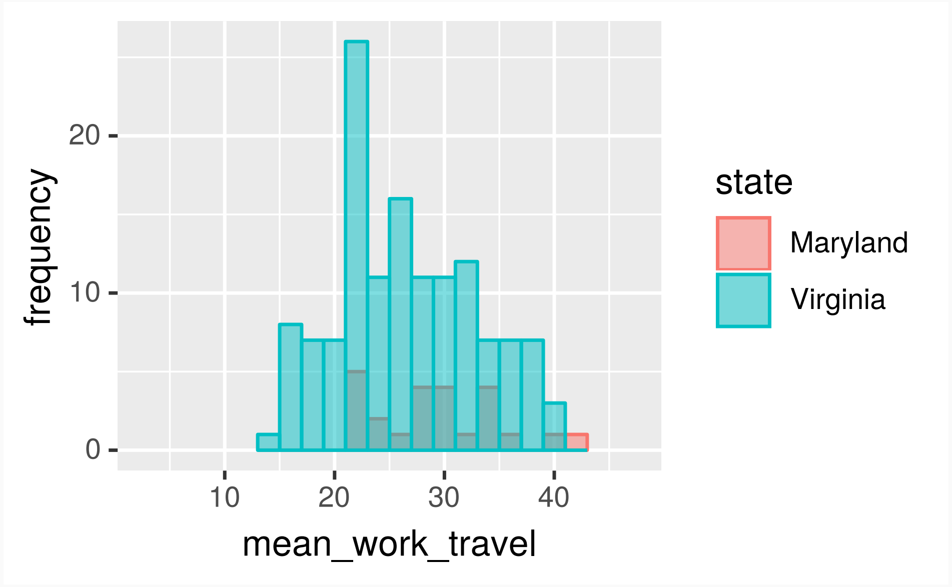
In which state am I more likely to have a 30 minute commute?

Maryland

Creating PMFs in R

With `ggplot2`, it's straightforward to convert a histogram into a PMF.

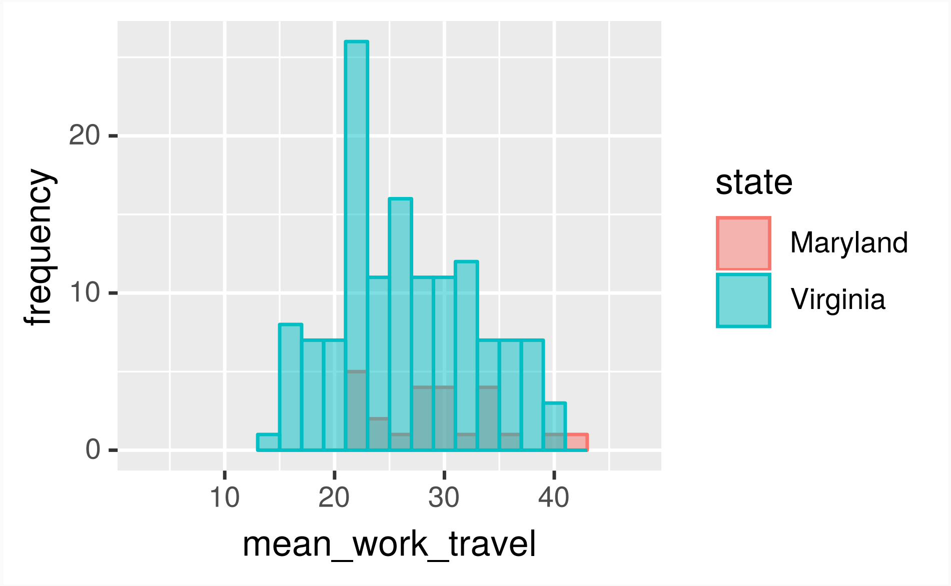
```r
county %>%
  filter(state == "Virginia" | state == "Maryland") %>%
  ggplot() +
  geom_histogram(
    mapping = aes(x = mean_work_travel, fill = state),
    position = "identity",
    alpha = 0.5
  )
```



Creating PMFs in R

With `ggplot2`, it's straightforward to convert a histogram into a PMF.

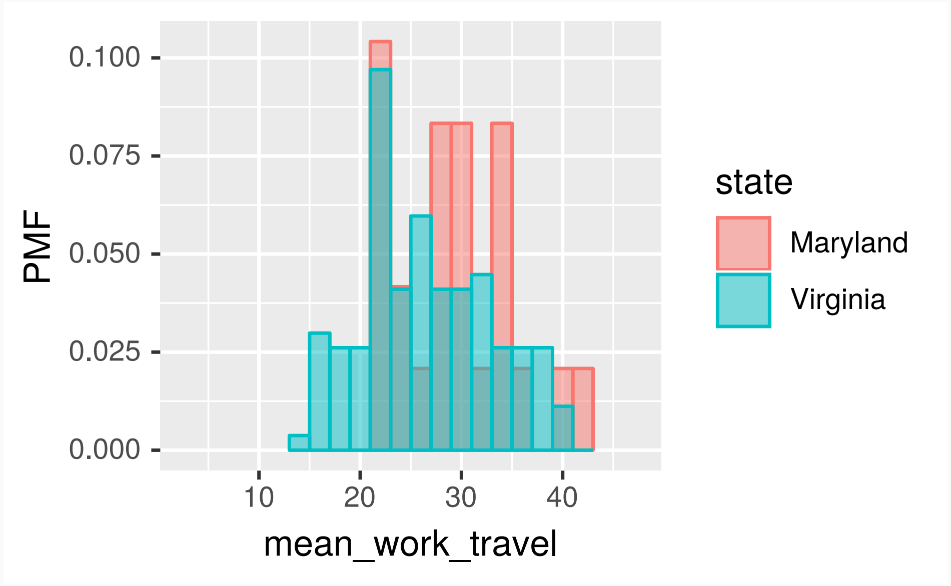
```
county %>%
  filter(state == "Virginia" | state == "Maryland") %>%
  ggplot() +
  geom_histogram(
    mapping = aes(x = mean_work_travel, fill = state),
    position = "identity",
    alpha = 0.5
  )
```



Creating PMFs in R

With `ggplot2`, it's straightforward to convert a histogram into a PMF.

```r
county %>%
  filter(state == "Virginia" | state == "Maryland") %>%
  ggplot() +
  geom_histogram(
    mapping = aes(x = mean_work_travel, y = ..density.., fill = state),
    position = "identity",
    alpha = 0.5
  )
```



Obtaining PMF values

Obtaining PMF values

1. Compute them manually

Obtaining PMF values

1. Compute them manually
2. Extract them from your `ggplot2` visualization

Obtaining PMF values

1. Compute them manually

2. Extract them from your `ggplot2` visualization

Obtaining PMF values

1. Compute them manually
2. Extract them from your `ggplot2` visualization

Assign the figure to a variable

```r
va_md_pmf_figure <- county %>%
  filter(state == "Virginia" | state == "Maryland") %>%
  ggplot() +
  geom_histogram(
    mapping = aes(x = mean_work_travel, y = ..density.., fill = state),
    binwidth = 2,
    center = 0
  )
```

Use `ggplot_build()` with `purrr::pluck()` and `as_data_frame()` as follows:

```r
va_md_pmf_data <- va_md_pmf_figure %>%
  ggplot_build() %>%
  purrr::pluck("data", 1) %>%
  as_data_frame()
```

Obtaining PMF values

```r
va_md_pmf_data %>%
  glimpse()
```

```
## Observations: 30
## Variables: 17
## $ fill     <chr> "#00BFC4", "#F8766D", "#00BFC4", "#F8766D", "#00BFC4"...
## $ y        <dbl> 0.003731343, 0.003731343, 0.029850746, 0.029850746, 0...
## $ count    <dbl> 1, 0, 8, 0, 7, 0, 7, 0, 26, 5, 11, 2, 16, 1, 11, 4, 1...
## $ x        <dbl> 14, 14, 16, 16, 18, 18, 20, 20, 22, 22, 24, 24, 26, 2...
## $ xmin     <dbl> 13, 13, 15, 15, 17, 17, 19, 19, 21, 21, 23, 23, 25, 2...
## $ xmax     <dbl> 15, 15, 17, 17, 19, 19, 21, 21, 23, 23, 25, 25, 27, 2...
## $ density  <dbl> 0.003731343, 0.000000000, 0.029850746, 0.000000000, 0...
## $ ncount   <dbl> 0.03846154, 0.00000000, 0.30769231, 0.00000000, 0.269...
## $ ndensity <dbl> 0.03846154, 0.00000000, 0.30769231, 0.00000000, 0.269...
## $ PANEL    <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ group    <int> 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1, 2, 1,...
## $ ymin     <dbl> 0.000000000, 0.003731343, 0.000000000, 0.029850746, 0...
## $ ymax     <dbl> 0.003731343, 0.003731343, 0.029850746, 0.029850746, 0...
## $ colour   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
## $ size     <dbl> 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5...
## $ linetype <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,...
## $ alpha    <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, N...
```

Obtaining PMF values

To get the Maryland PMF data:

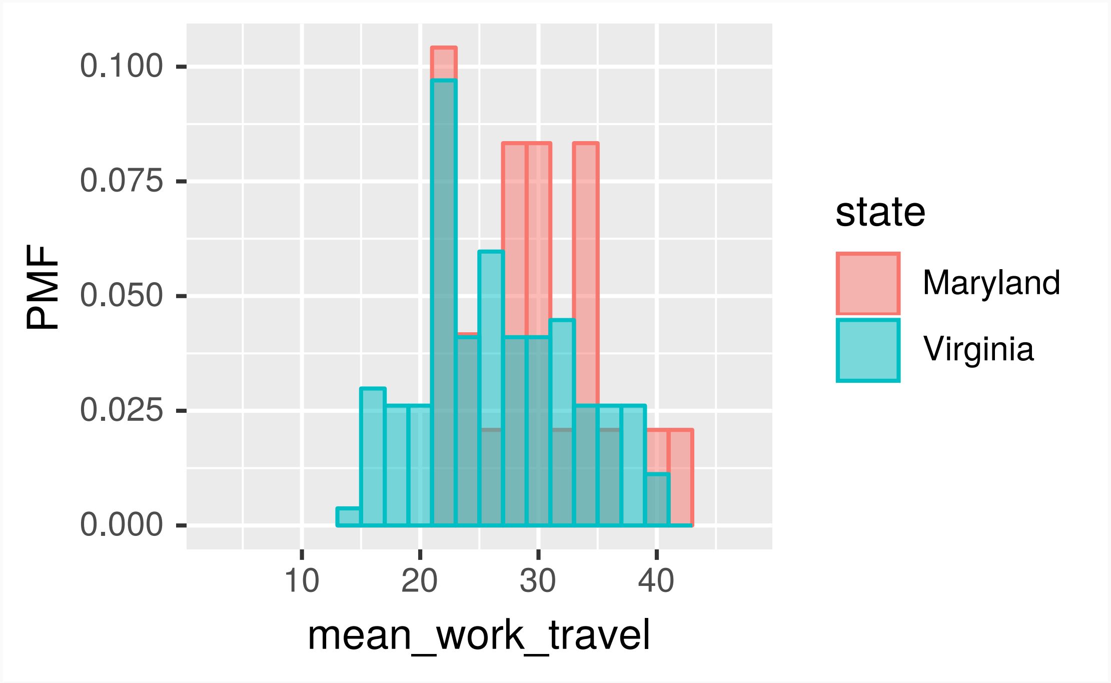
```r
md_pmf_data <- va_md_pmf_data %>%
  filter(group == 1) %>%
  select(x, density)
```

x	density
14	0
16	0
18	0
20	0
22	0.104166666666667
24	0.0416666666666667
26	0.0208333333333333
…	…

Obtaining PMF values

To get the Maryland PMF data:

```
md_pmf_data <- va_md_pmf_data %>%
  filter(group == 1) %>%
  select(x, density)
```



Obtaining PMF values

To get the Maryland PMF data:

```r
md_pmf_data <- va_md_pmf_data %>%
  filter(group == 1) %>%
  select(x, density)
```

x	**density**
14	0
16	0
18	0
20	0
22	0.104166666666667
24	0.0416666666666667
26	0.0208333333333333
…	…

Obtaining PMF values

To get the Virginia PMF data:

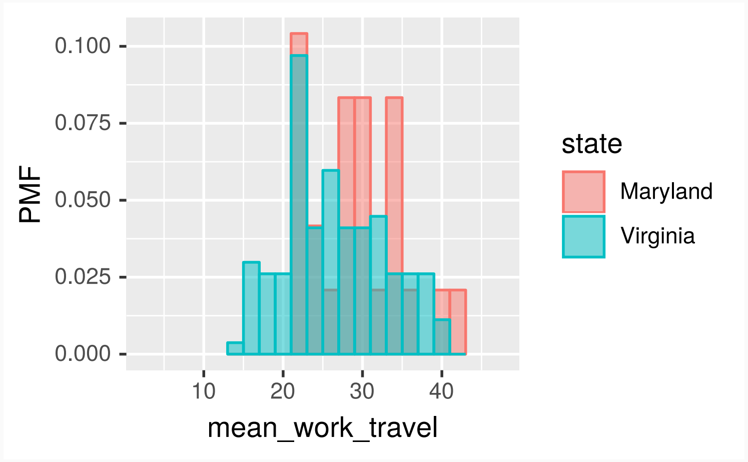
```r
va_pmf_data <- va_md_pmf_data %>%
  filter(group == 2) %>%
  select(x, density)
```

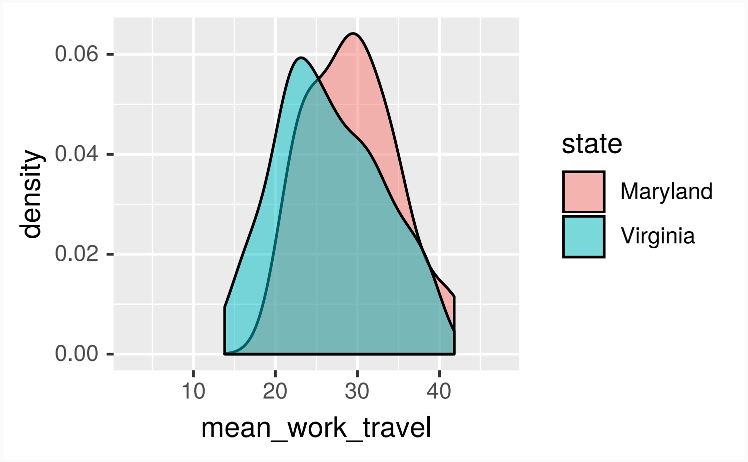
x	density
14	0.00373134328358209
16	0.0298507462686567
18	0.0261194029850746
20	0.0261194029850746
22	0.0970149253731343
24	0.041044776119403
26	0.0597014925373134
…	…

Density plots as an alternative

As an alternative to the probability mass function, we can also use the density plots provided in ggplot2. Unlike the histograms, they are automatically normalized.

```
county %>%
  filter(state == "Virginia" | state == "Maryland") %>%
  ggplot() +
  geom_density(
    mapping = aes(x = mean_work_travel, fill = state),
    alpha = 0.5
  )
```





Credits

License