Graphics with ggplot2

Spring 2023

April 02 2023

Data Visualization provides a powerful way to communicate data-driven findings, motivate analyses, and detect flaw

Visualization in the data science workflow

Data visualization is a key skill for data scientists.

Useful for:

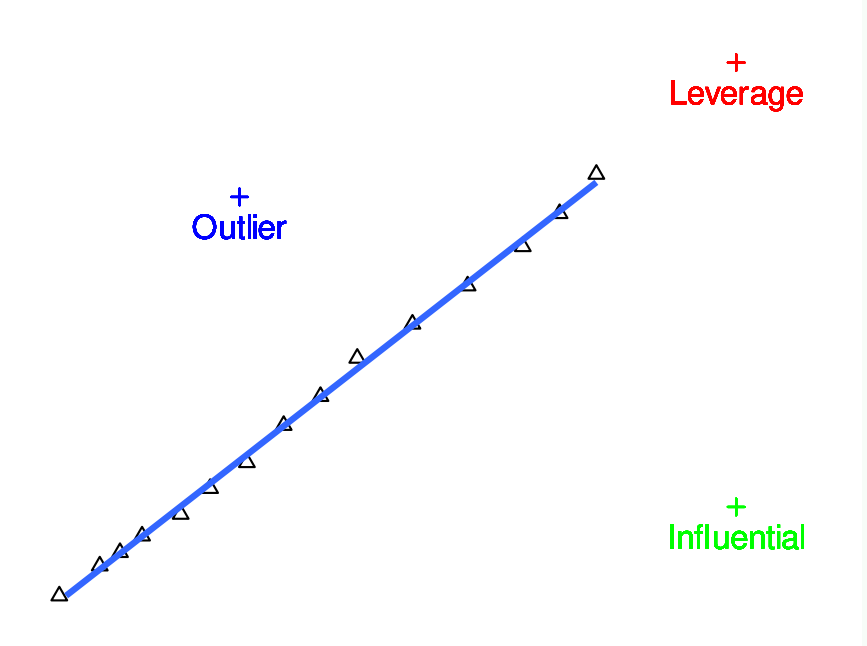
- ***Identification of outliers***
- ***Guidance of recoding operations***
- ***Summarize distributions***
- ***Discover patterns, relationships***
- ***Visualize uncertainty***

Visualization in the data science workflow

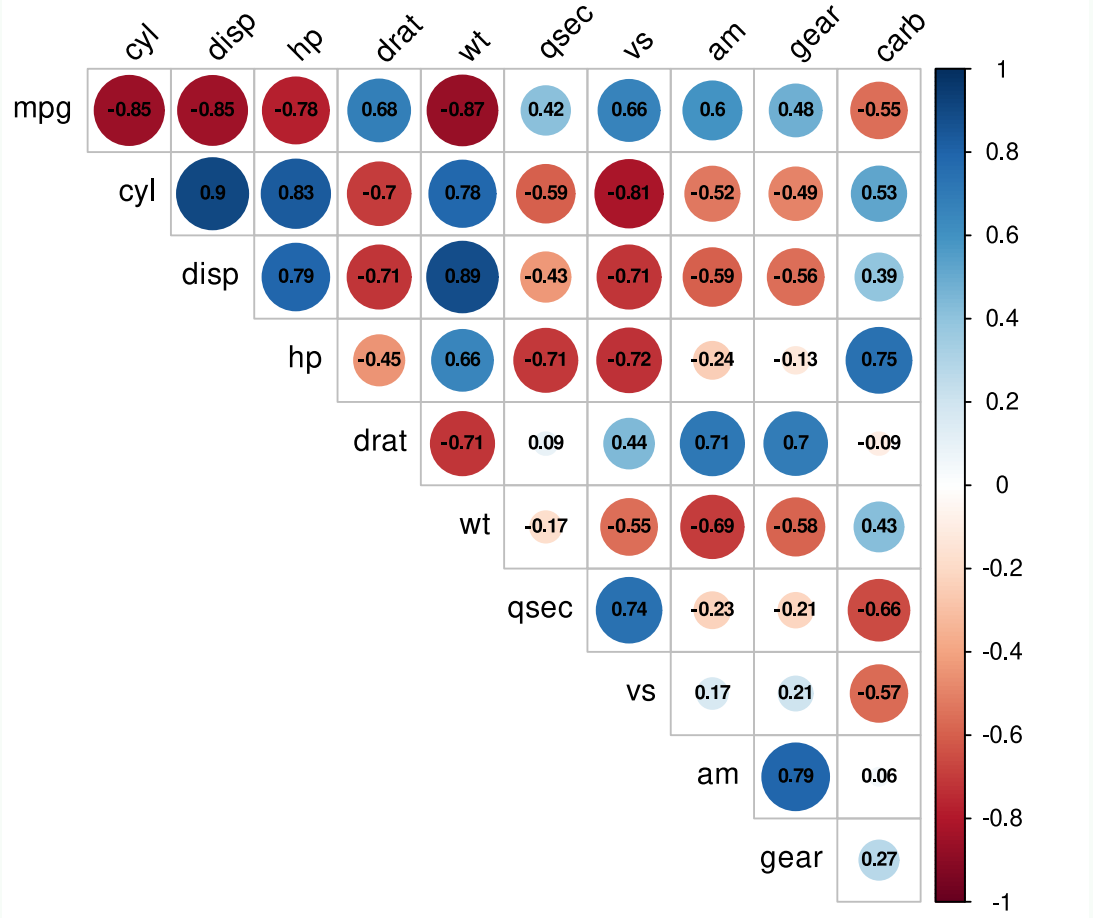
Data visualization is a key skill for data scientists.

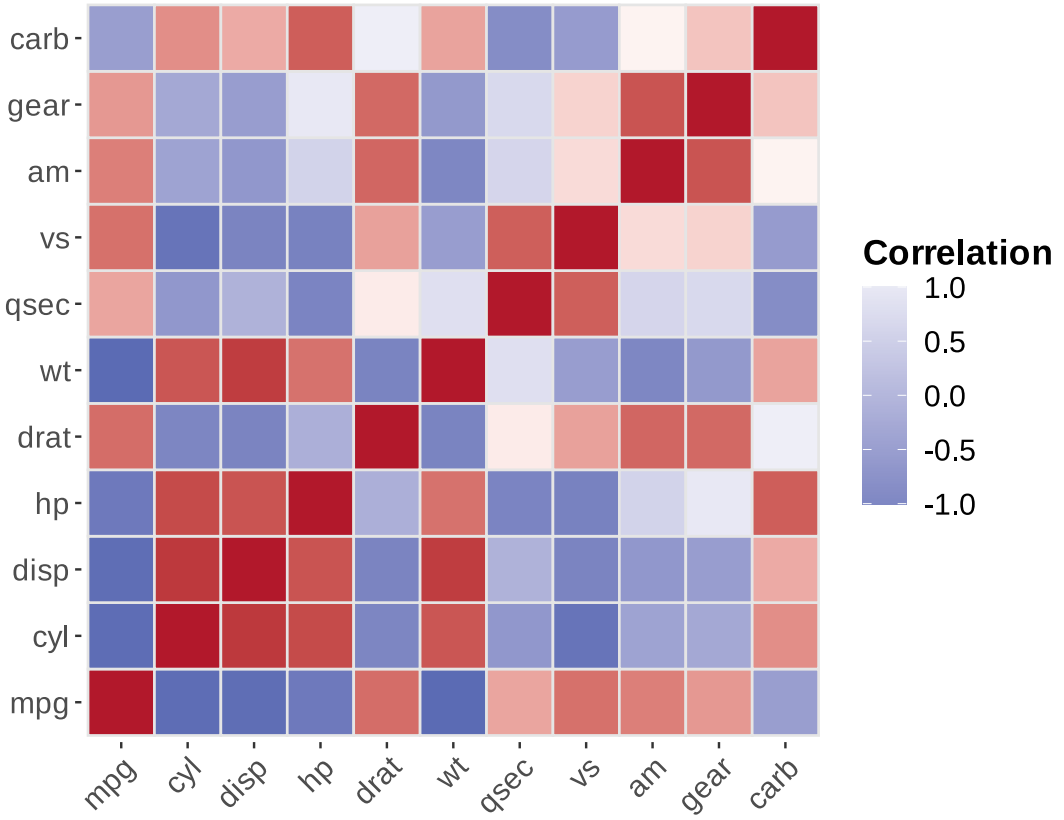
Useful for:

- ***Identification of outliers***
- ***Guidance of recoding operations***
- ***Summarize distributions***
- ***Discover patterns, relationships***
- ***Visualize uncertainty***



Which visualization do you prefer?





Which quantity do I want to visualize?

- Amounts
- Distributions
- Proportions
- Associations
- Trends
- Estimates
- Uncertainty

Which quantity do I want to visualize?

- Amounts
- Distributions
- Proportions
- Associations
- Trends
- Estimates
- Uncertainty

Which question do I want to answer?

- **"Is the distribution normal (or uniform or...)?" → Histogram, density plot, Q-Q plot**
- **"Are univariate distributions across subgroups different?" → Boxplots**
- **"How do differences in amounts between groups compare?" → Barplot**
- **"What is the relationship between x and y ?" → Scatterplot, contour plot, hex bins**
- **"What are the correlations in a set of variables?" → Correlogram, small multiples**
- **"Are the data clustered by subgroup?" → Scatterplot with color**
- **"How uncertain are estimates? → Error bars, confidence bands**

ggplot2 — Overview

- *A powerful package for visualizing data*
- *Used widely by academics and industries alike*

Some useful resources

- The package documentation
- The book by its creator Hadley Wickham
- The reference page
- The extensions, maintained by the `ggplot2` community

Our building blocks 🧱

Essentials

- **Data**: the data frame, or data frames, we will use to plot
- **Aesthetics**: the variables we will be working with
- **Geometric objects**: the type of visualization

Additional elements

- **Theme adjustments**: linewidth, text, colors etc
- **Facets**
- **Coordinate system**
- **Statistical transformations**
- **Position adjustments**
- **Scales**

Data

In ggplot2, we always specify a data frame with :

```
ggplot(name_of_your_df)
```

Data

In ggplot2, we always specify a data frame with :

```
ggplot(name_of_your_df)
```

Aesthetics

Specify the variables in the data frame we will be using and what role they play. Use the function aes() within the ggplot() function after the data frame.

```
ggplot(name_of_your_df, aes(x = your_x_axis_variable, y = your_y_axis_variable))
```

Data

In ggplot2, we always specify a data frame with :

```
ggplot(name_of_your_df)
```

Aesthetics

Specify the variables in the data frame we will be using and what role they play. Use the function aes() within the ggplot() function after the data frame.

```
ggplot(name_of_your_df, aes(x = your_x_axis_variable, y = your_y_axis_variable))
```

Beyond your axis, you can add more aesthetics representing further dimensions of the data in the two dimensional graphic plane, such as: shape, linewidth, color, fill, alpha to name a few.

Geometric objects

The third layer required to create our plot (which determines the specific kind of visualization, such as a bar plot or scatter plot) involves adding a geometric object.

To do this, we should append a plus **(+)** at the end of the initial line and specify the desired geometric object type, like `geom_point()` for a scatter plot or `geom_bar()` for bar plots.

```
ggplot(name_of_your_df, aes(x = your_x_axis_variable, y = your_y_axis_variable)) +
  geom_point()
```

Theme and Axes

At this stage, our plot might require a few finishing touches. We might want to adjust the axis names or remove the default gray background. To accomplish this, we should add another layer, preceded by a plus sign **(+)**.

To modify the axis names, we can use the `labs()` function. Additionally, we can apply some of the pre-defined themes, such as `theme_minimal()`.

```
ggplot(name_of_your_df, aes(x = your_x_axis_variable, y = your_y_axis_variable)) +
  geom_point() +
  theme_minimal() +
  labs(x = "Your x label",
       y = "Your y label")
```

Common ggplot2 options

```
ggplot(data) +    # data
  <geometry_funs>(aes(<variables>)) + # aesthetic variable mapping
  <label_funs> +  # add context
  <facet_funs> +  # add facets (optional)
  <coordinate_funs> +  # play with coords (optional)
  <scale_funs> + # play with scales (optional)
  <theme_funs> # play with axes, colors, etc (optional)
```

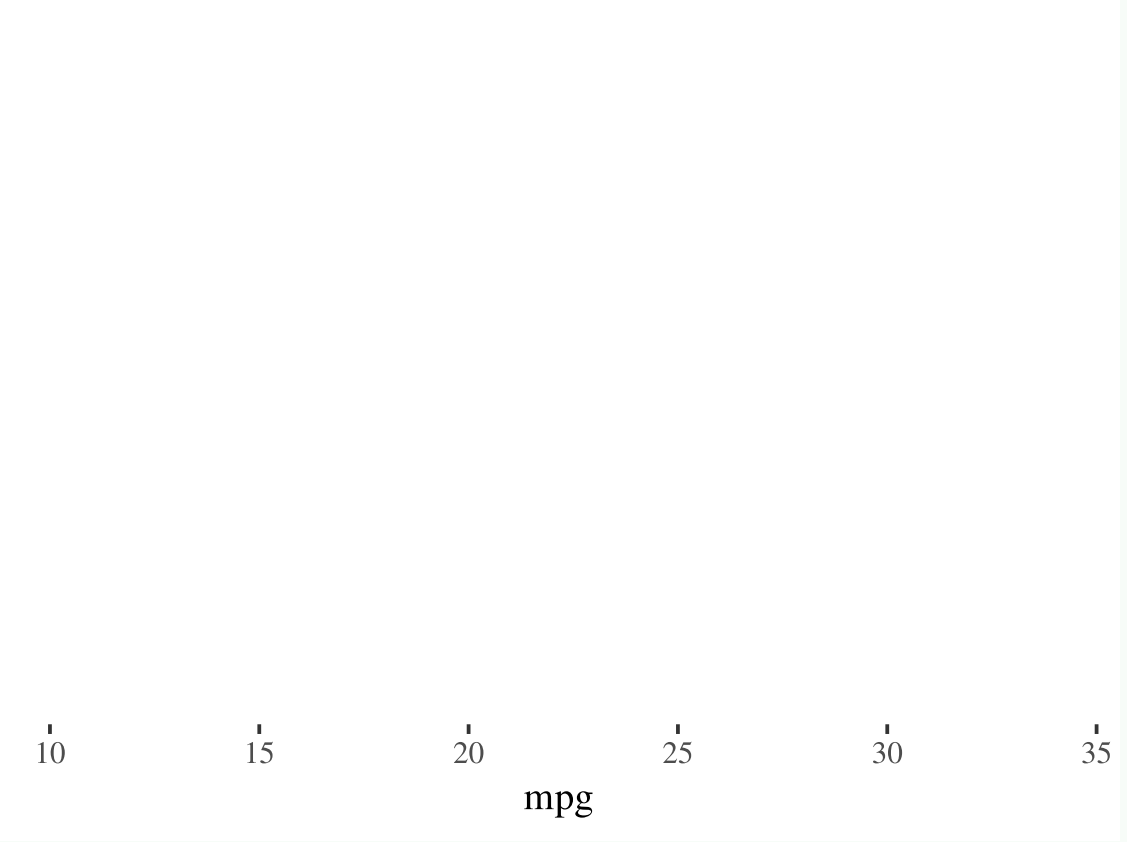
See the ggplot2 cheatsheets for more details

Histogram

```
# Histogram of mpg (miles per gallon) in the mtcar
```

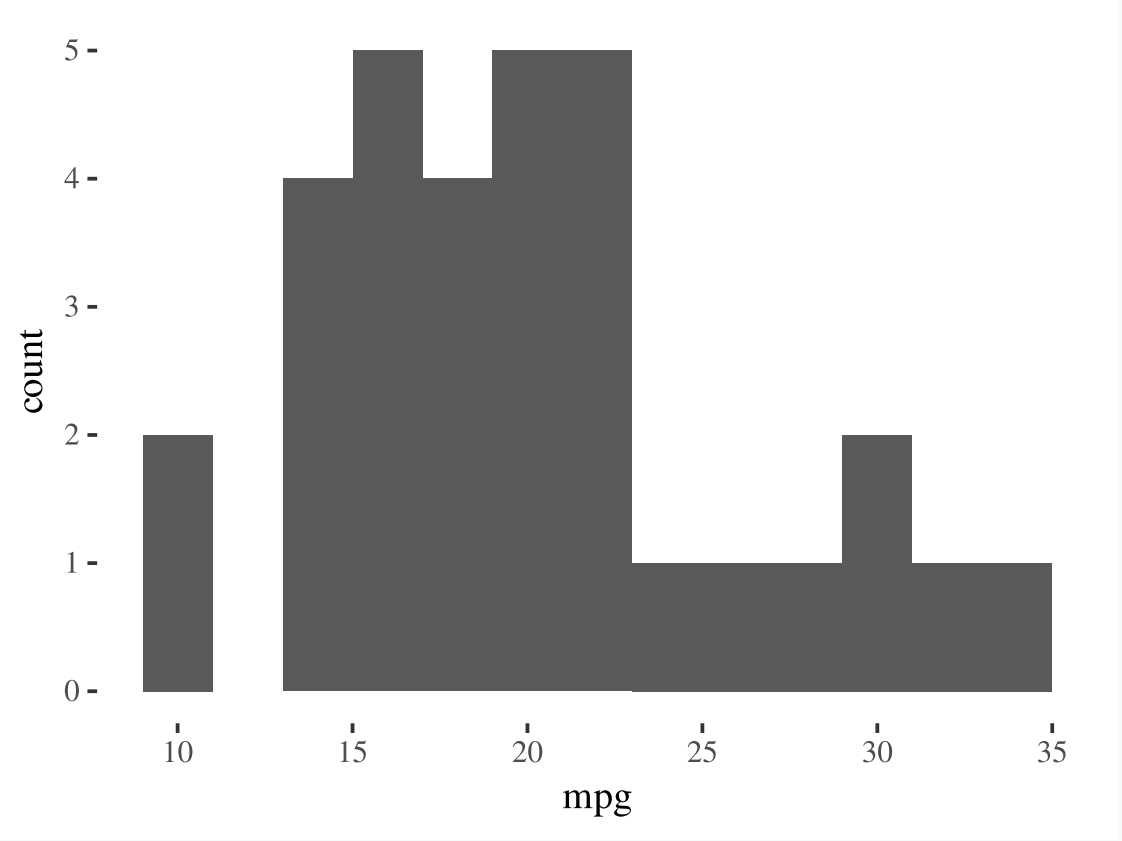
Histogram

```
# Histogram of mpg (miles per gallon) in the mtcar
ggplot(mtcars, aes(x = mpg))
```



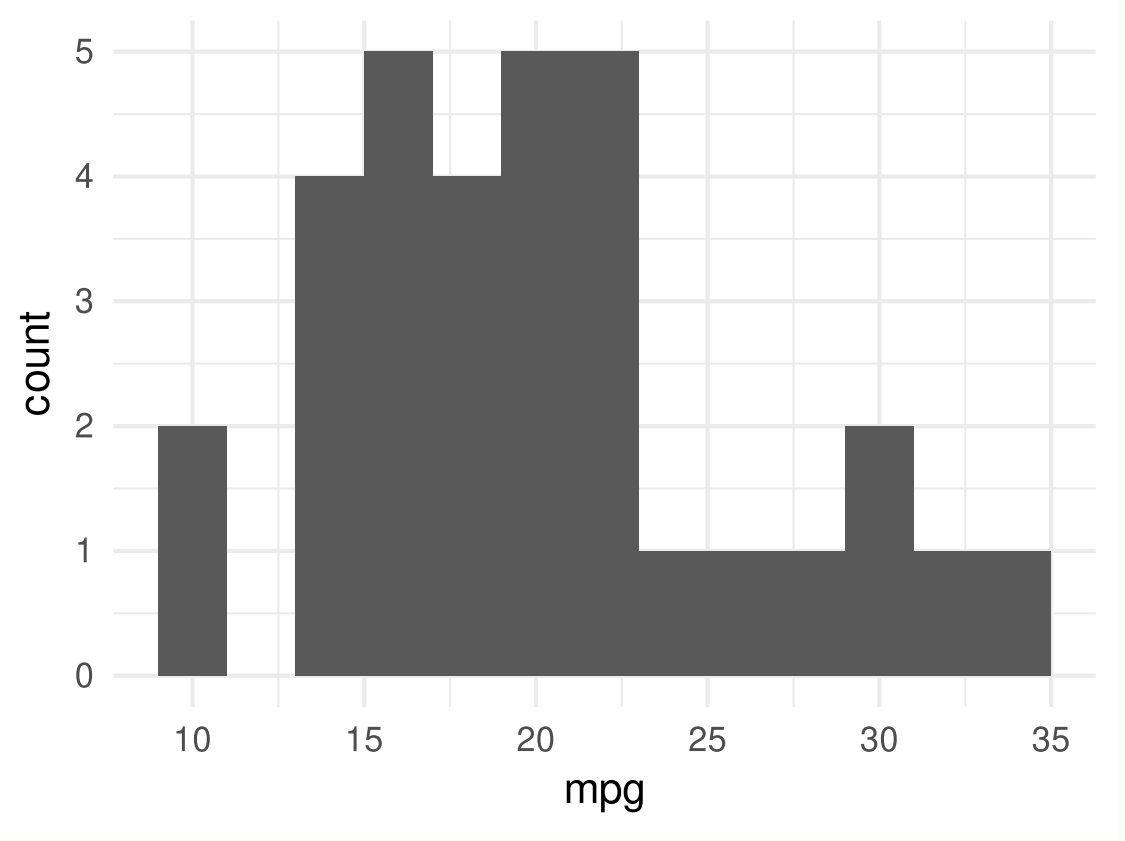
Histogram

```
# Histogram of mpg (miles per gallon) in the mtcar
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(binwidth = 2)
```



Histogram

```
# Histogram of mpg (miles per gallon) in the mtcar
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(binwidth = 2) +
  theme_minimal()
```

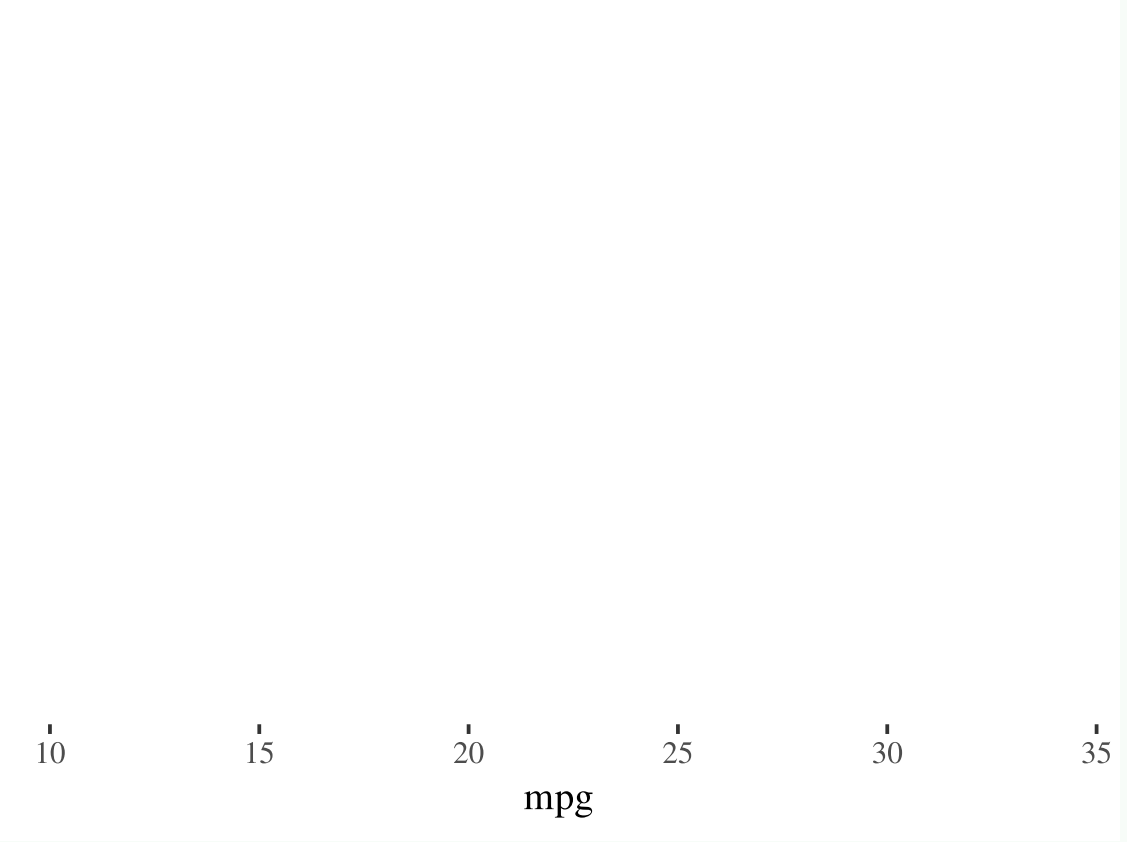


Density Plot

```
# Density plot of mpg (miles per gallon)  # Densit
```

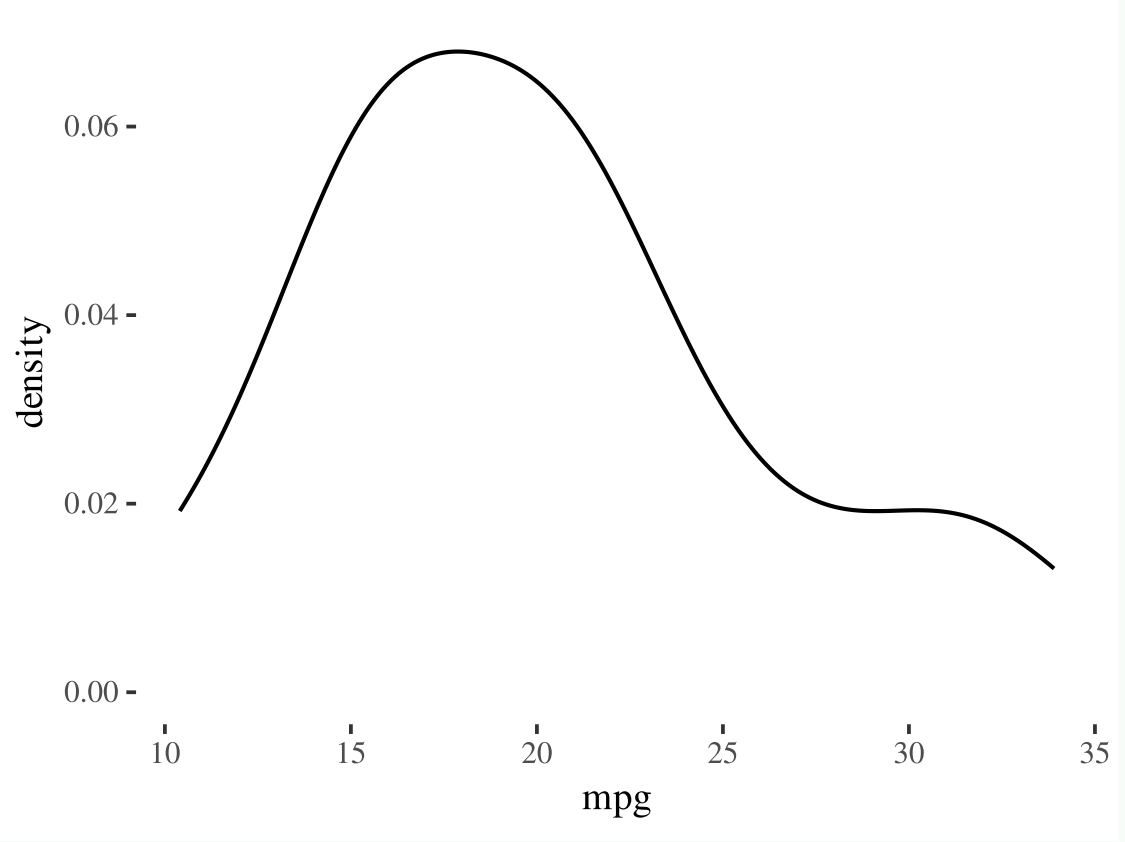
Density Plot

```
# Density plot of mpg (miles per gallon)  # Densit
ggplot(mtcars, aes(x = mpg))
```



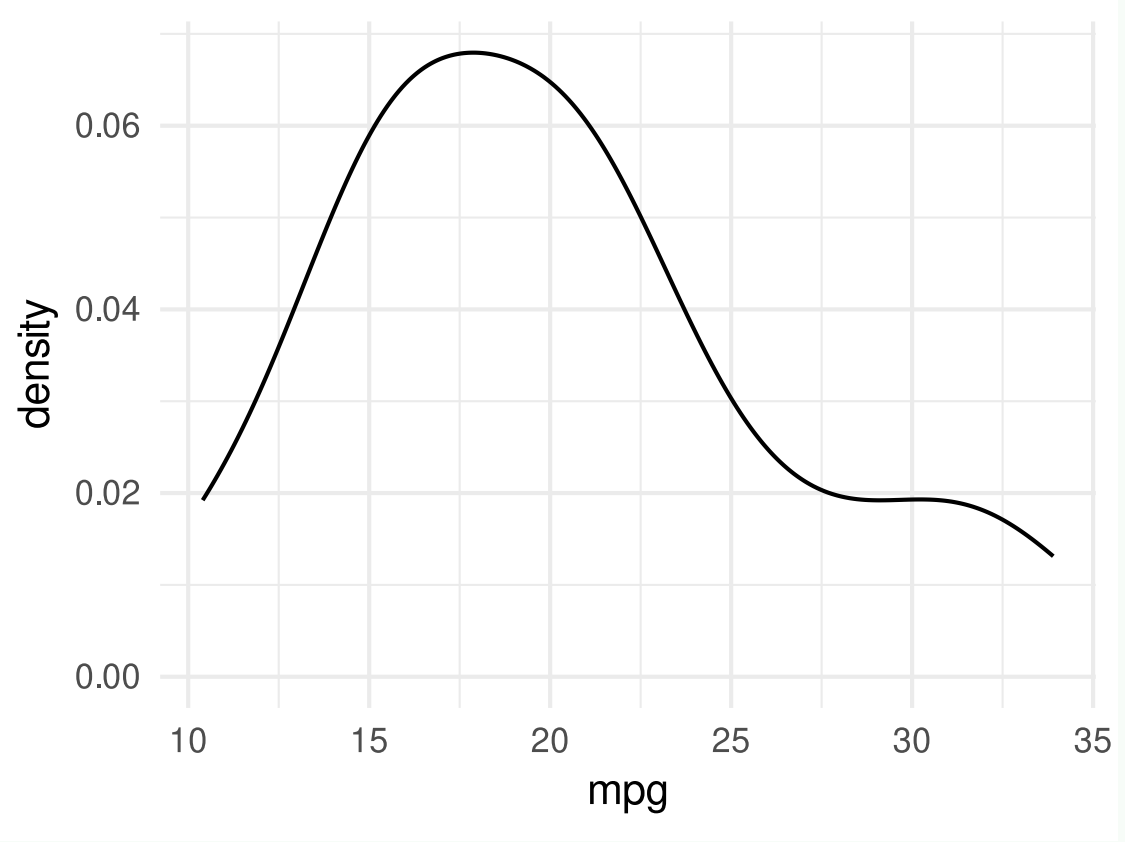
Density Plot

```
# Density plot of mpg (miles per gallon)  # Densit
ggplot(mtcars, aes(x = mpg)) +
  geom_density()
```



Density Plot

```
# Density plot of mpg (miles per gallon)  # Densit
ggplot(mtcars, aes(x = mpg)) +
  geom_density() +
  theme_minimal()
```

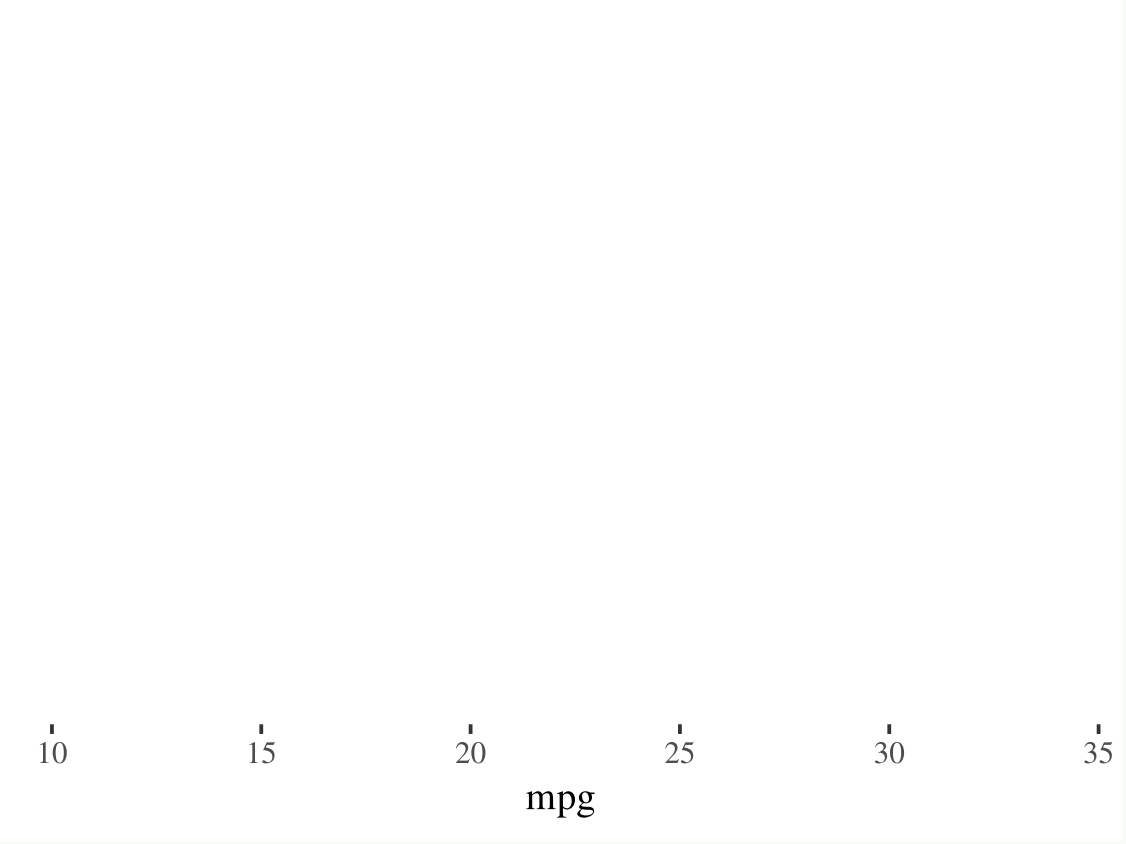


Histogram with density overlay

```
# Density plot of mpg (miles per gallon)  # Densit
```

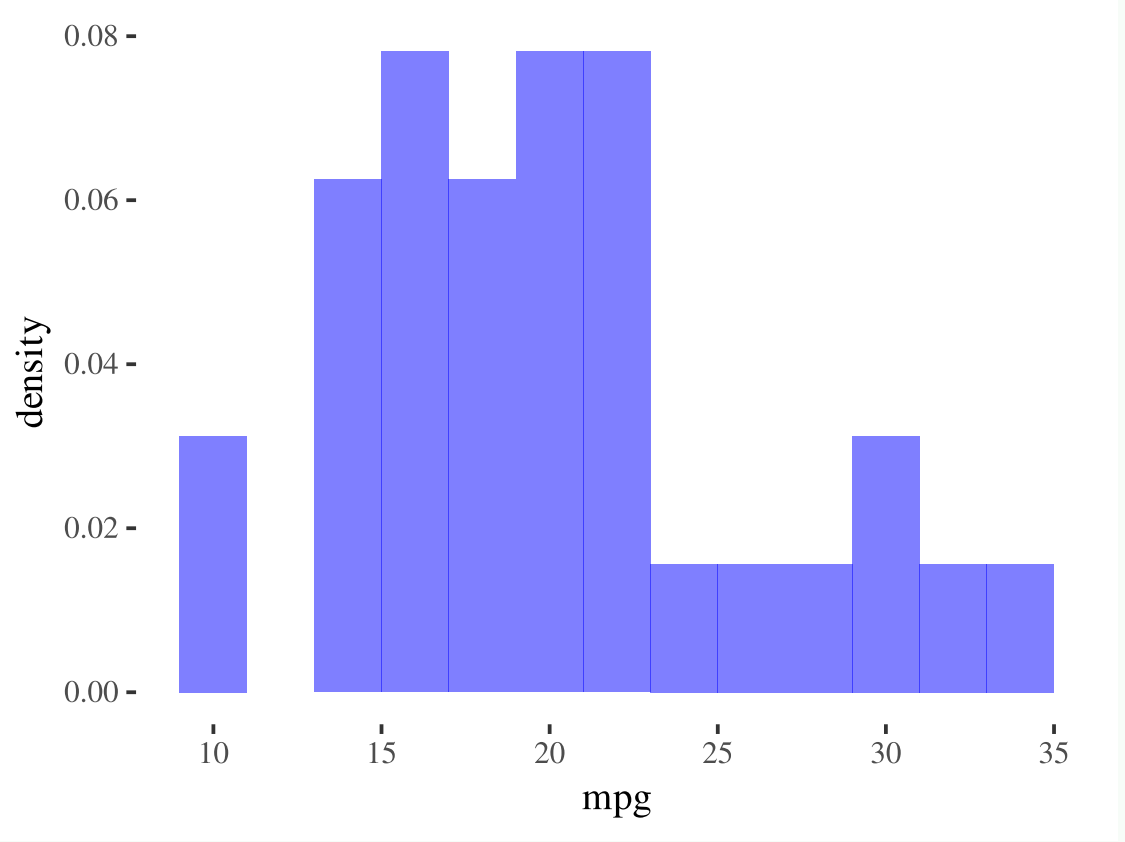
Histogram with density overlay

```
# Density plot of mpg (miles per gallon)  # Densit
ggplot(mtcars, aes(x = mpg))
```



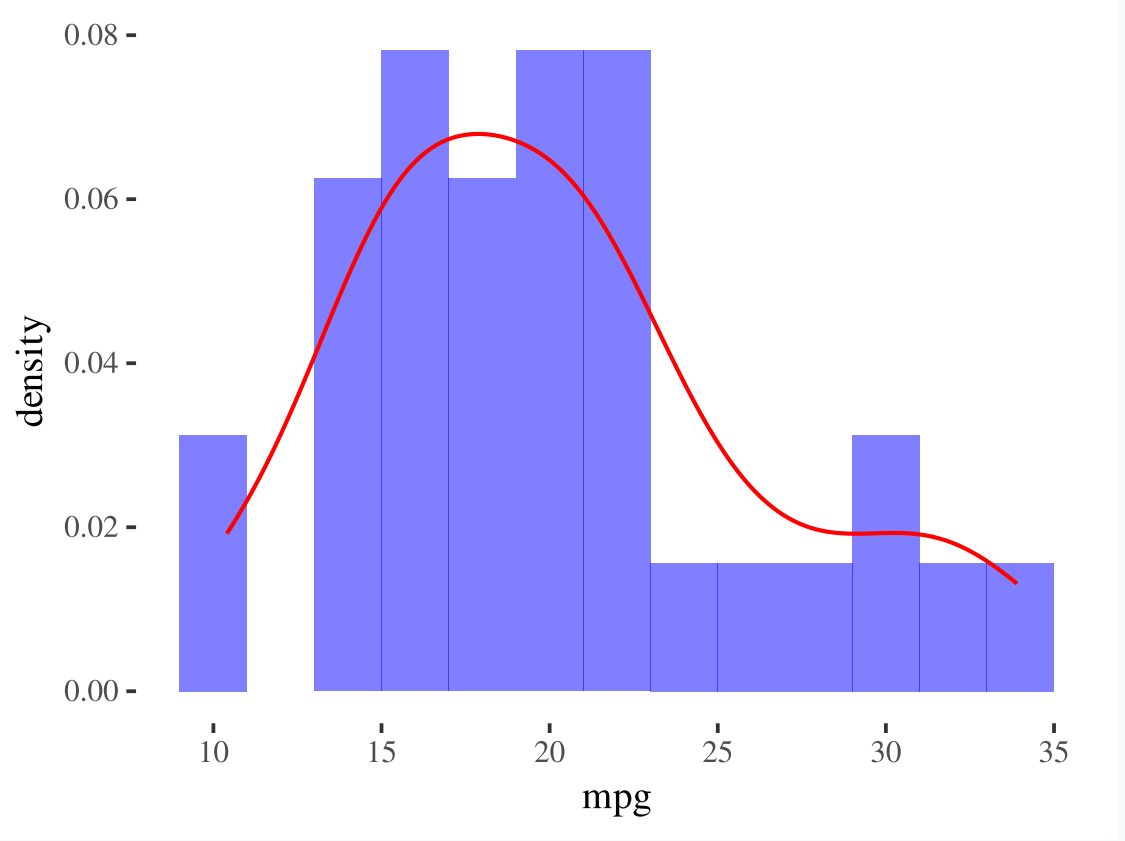
Histogram with density overlay

```
# Density plot of mpg (miles per gallon)  # Densit
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(aes(y = ..density..),
                 binwidth = 2,
                 alpha = 0.5,
                 fill = "blue")
```



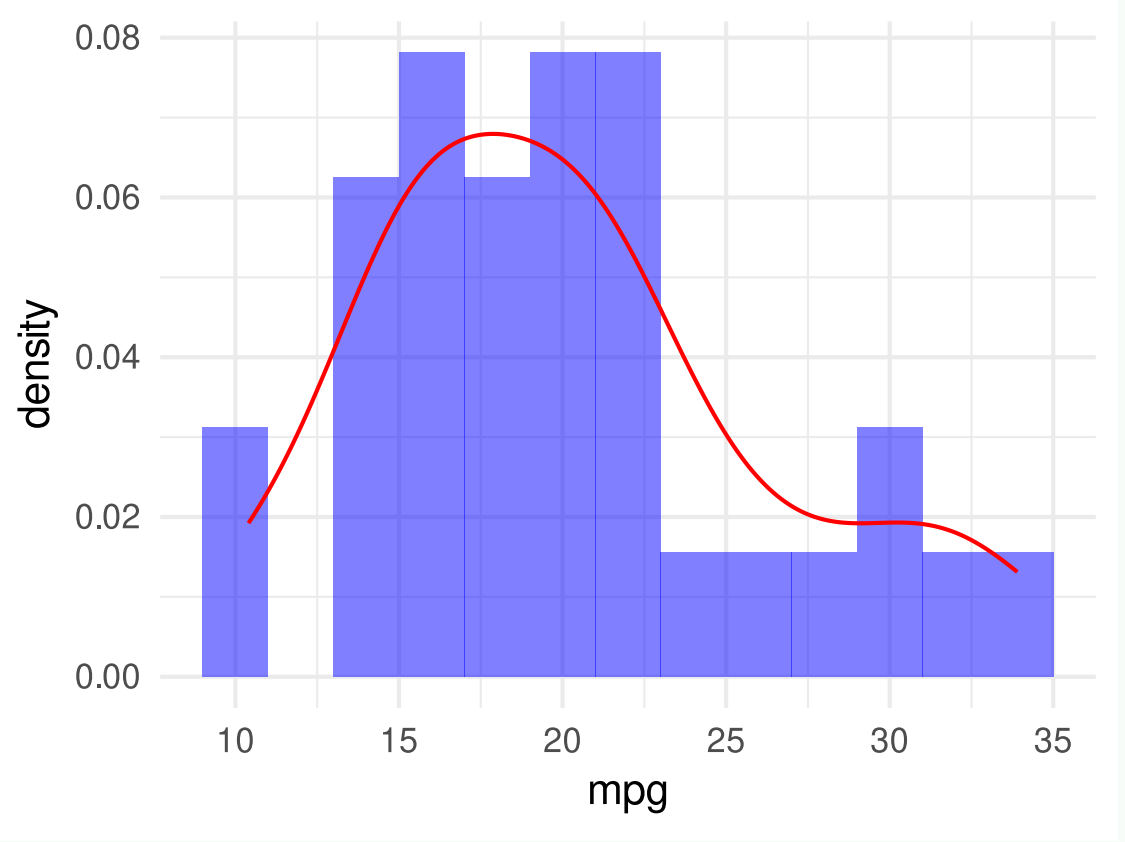
Histogram with density overlay

```
# Density plot of mpg (miles per gallon)  # Densit
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(aes(y = ..density..),
                 binwidth = 2,
                 alpha = 0.5,
                 fill = "blue") +
  geom_density(color = "red")
```



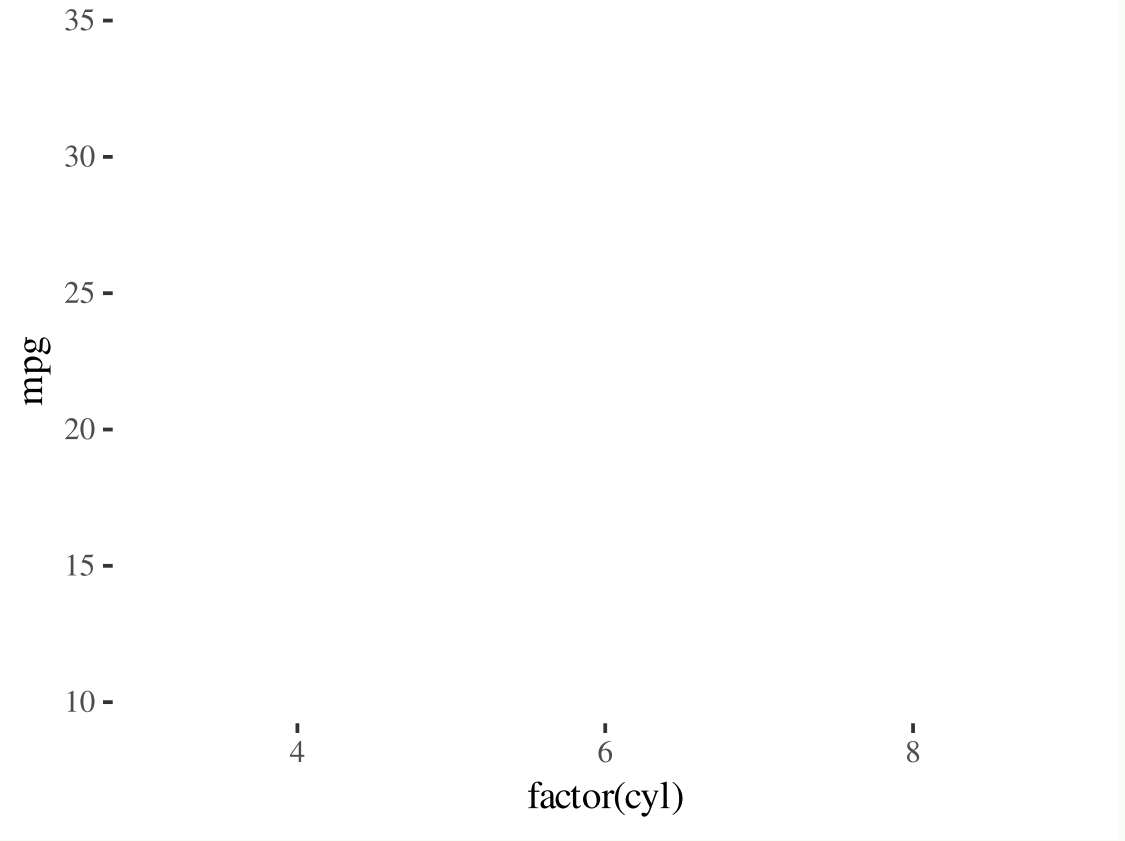
Histogram with density overlay

```r
# Density plot of mpg (miles per gallon)  # Densit
ggplot(mtcars, aes(x = mpg)) +
  geom_histogram(aes(y = ..density..),
                 binwidth = 2,
                 alpha = 0.5,
                 fill = "blue") +
  geom_density(color = "red") +
  theme_minimal()
```



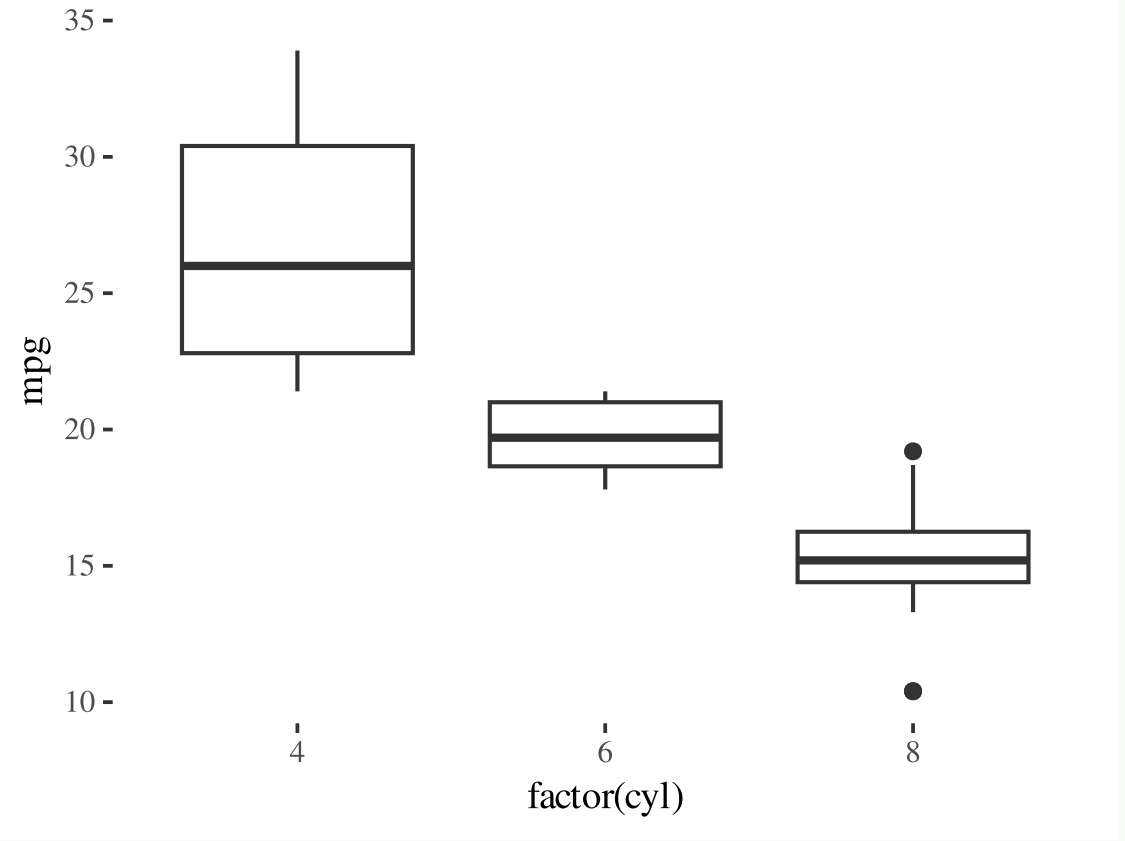
Boxplot

```
# Boxplot of mpg by number of cylinders in the mtc
ggplot(mtcars, aes(x = factor(cyl), y = mpg))
```



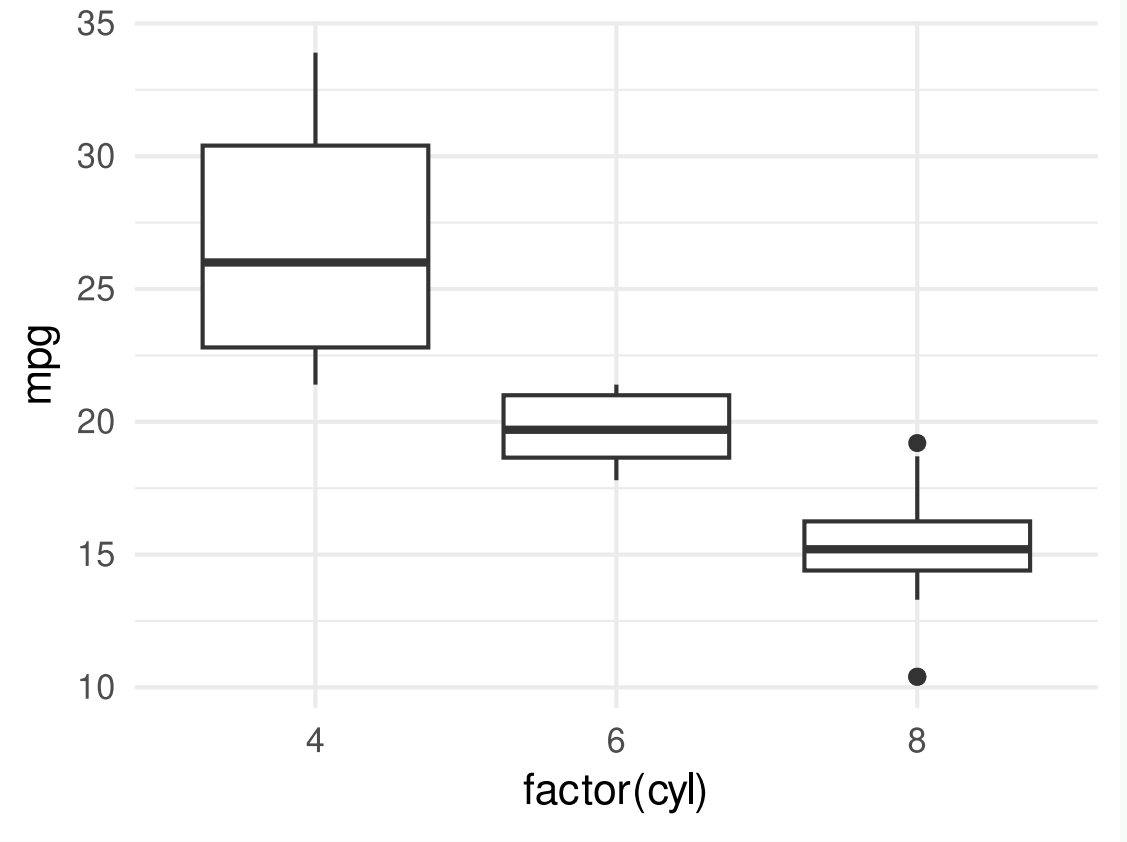
Boxplot

```
# Boxplot of mpg by number of cylinders in the mtc
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_boxplot()
```



Boxplot

```
# Boxplot of mpg by number of cylinders in the mtc
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_boxplot() +
  theme_minimal()
```

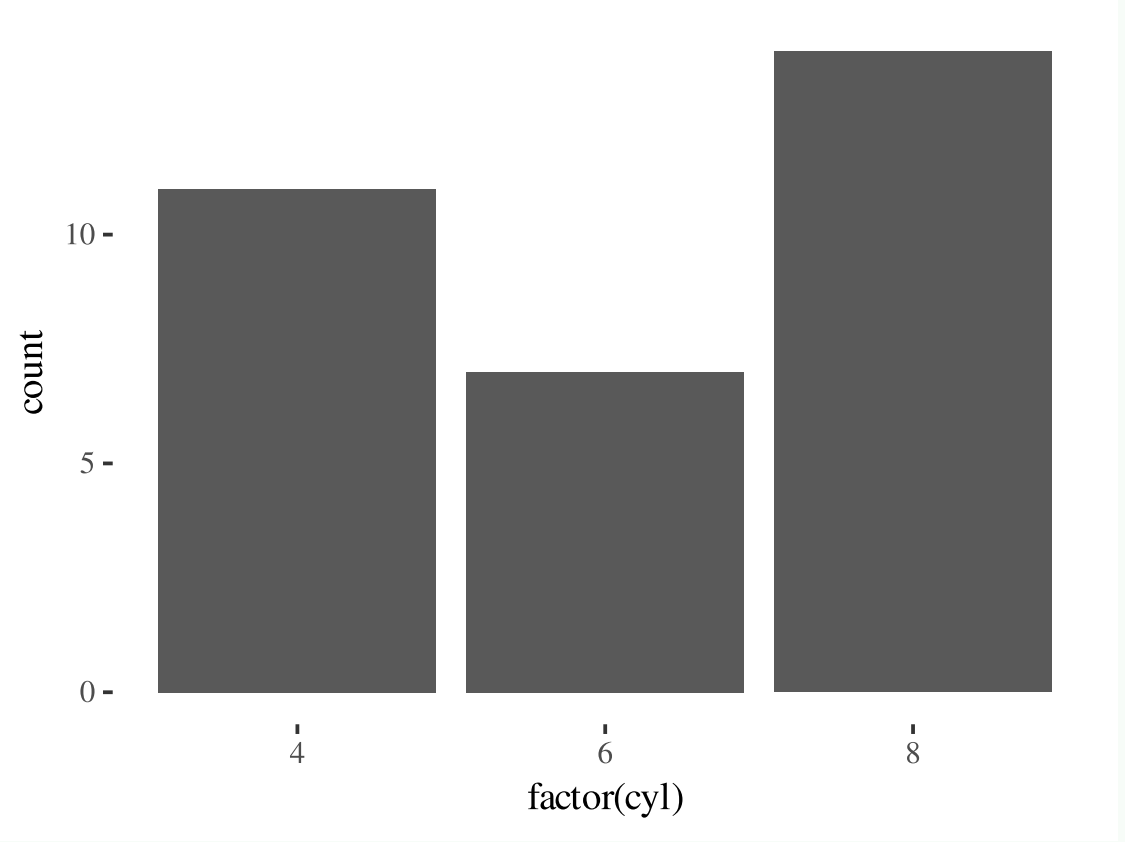


Barplot

```
# Barplot of number of cars by number of cylinders
ggplot(mtcars, aes(x = factor(cyl)))
```

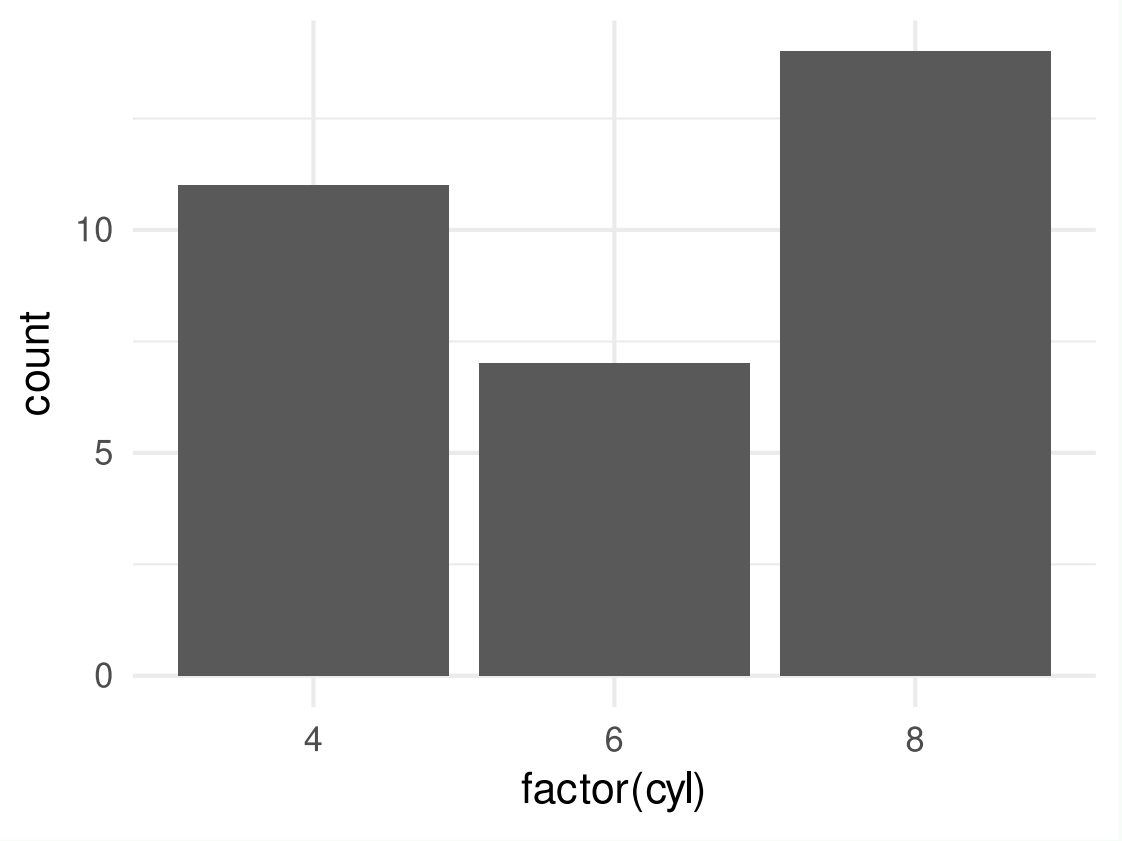
Barplot

```
# Barplot of number of cars by number of cylinders
ggplot(mtcars, aes(x = factor(cyl))) +
  geom_bar()
```



Barplot

```
# Barplot of number of cars by number of cylinders
ggplot(mtcars, aes(x = factor(cyl))) +
  geom_bar() +
  theme_minimal()
```

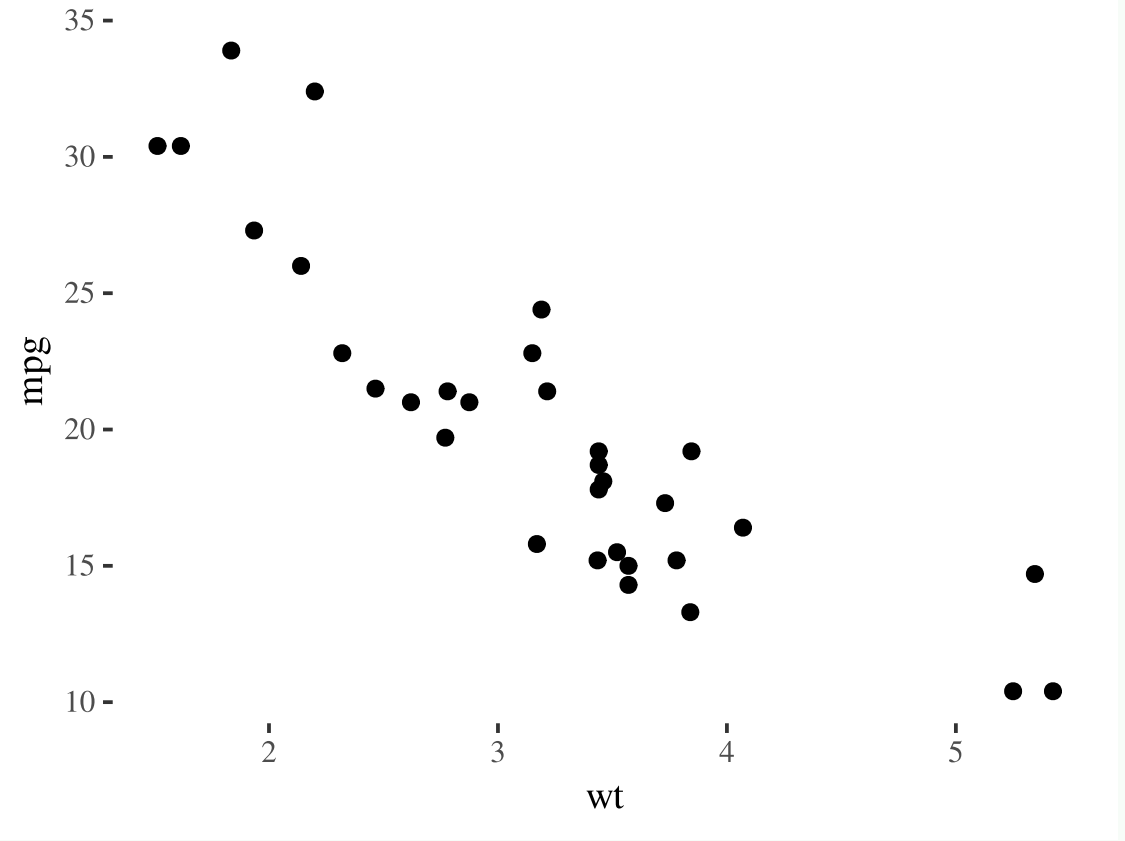


Scatter plot

```
# Scatterplot of mpg vs. weight in the mtcars data
ggplot(mtcars, aes(x = wt, y = mpg))
```

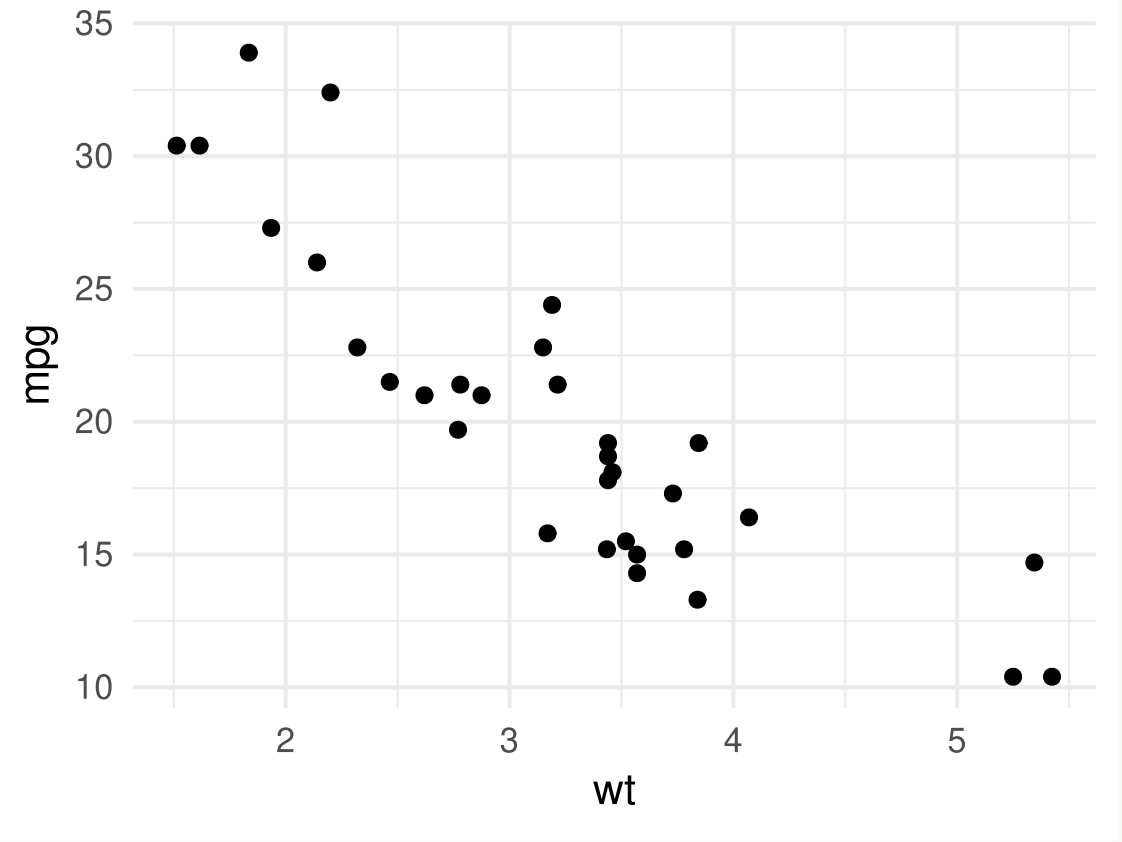
Scatter plot

```
# Scatterplot of mpg vs. weight in the mtcars data
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()
```



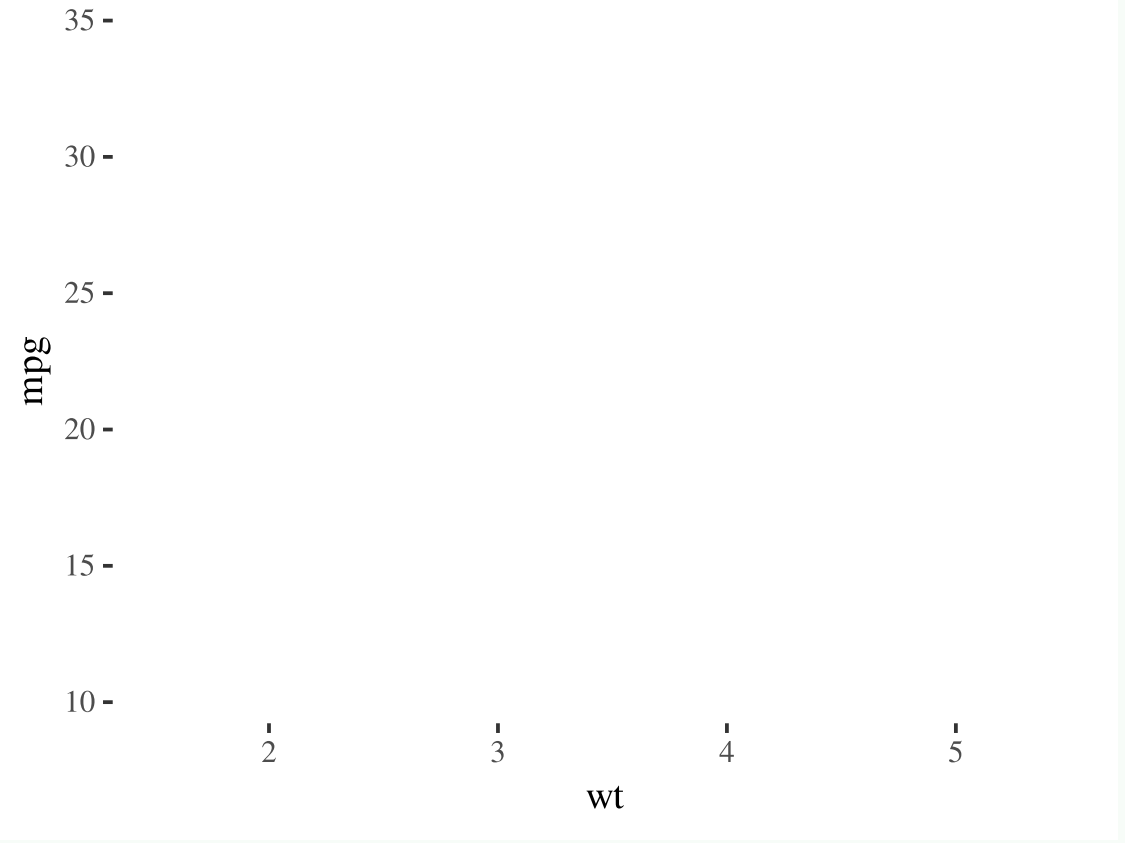
Scatter plot

```
# Scatterplot of mpg vs. weight in the mtcars data
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  theme_minimal()
```



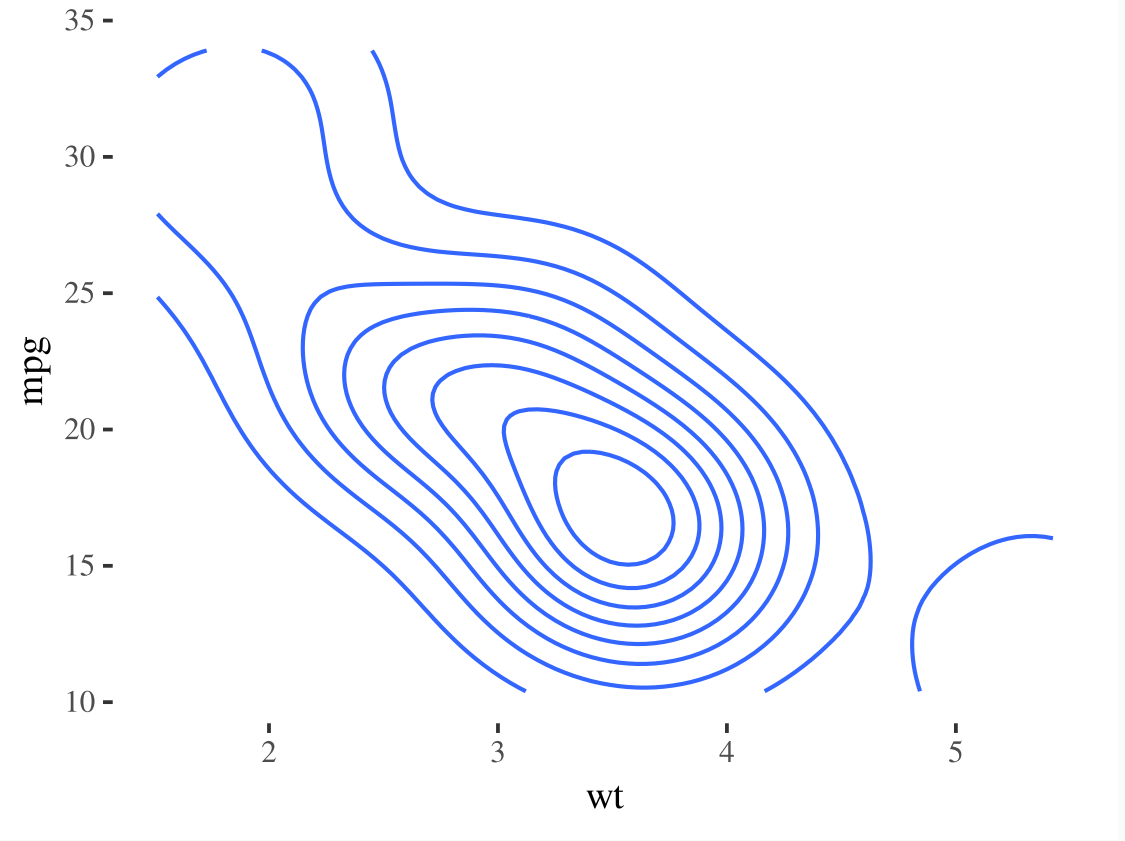
Contour plot

```
# Contour plot of mpg vs. weight in the mtcars dat
ggplot(mtcars, aes(x = wt, y = mpg))
```



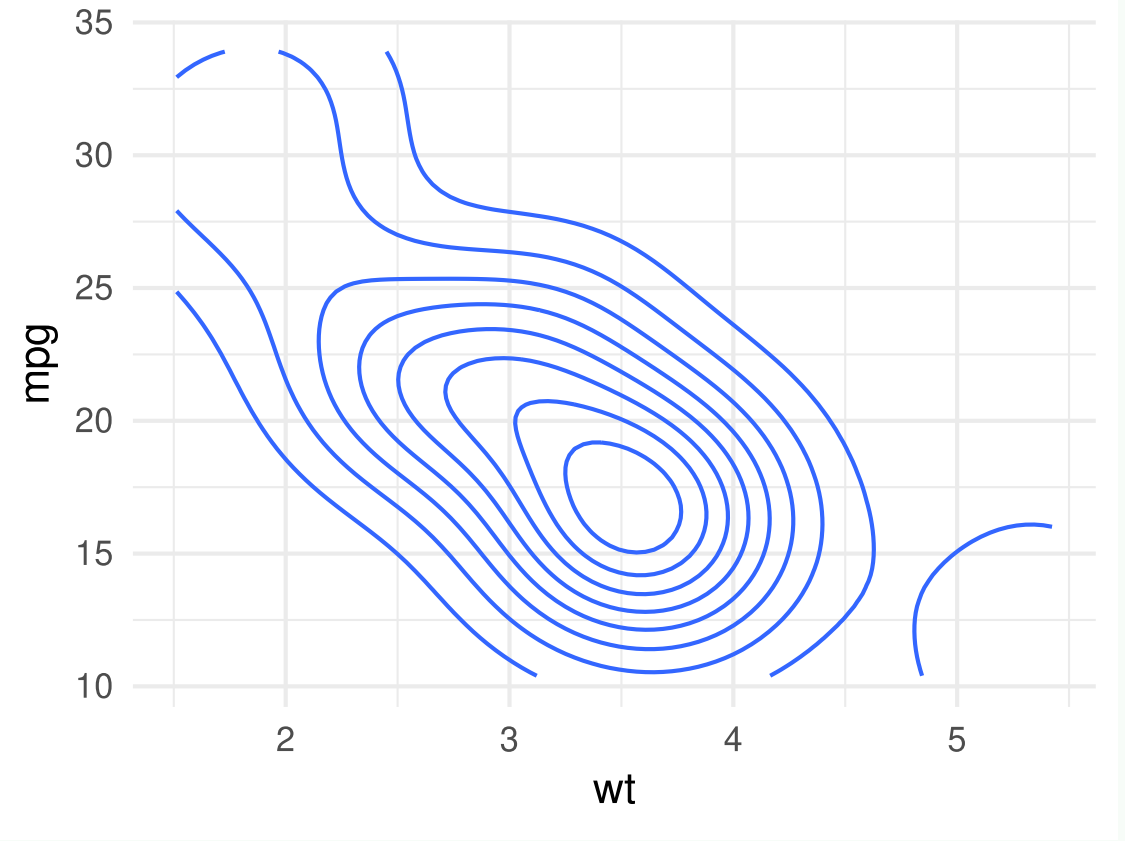
Contour plot

```
# Contour plot of mpg vs. weight in the mtcars dat
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_density_2d()
```



Contour plot

```
# Contour plot of mpg vs. weight in the mtcars dat
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_density_2d() +
  theme_minimal()
```

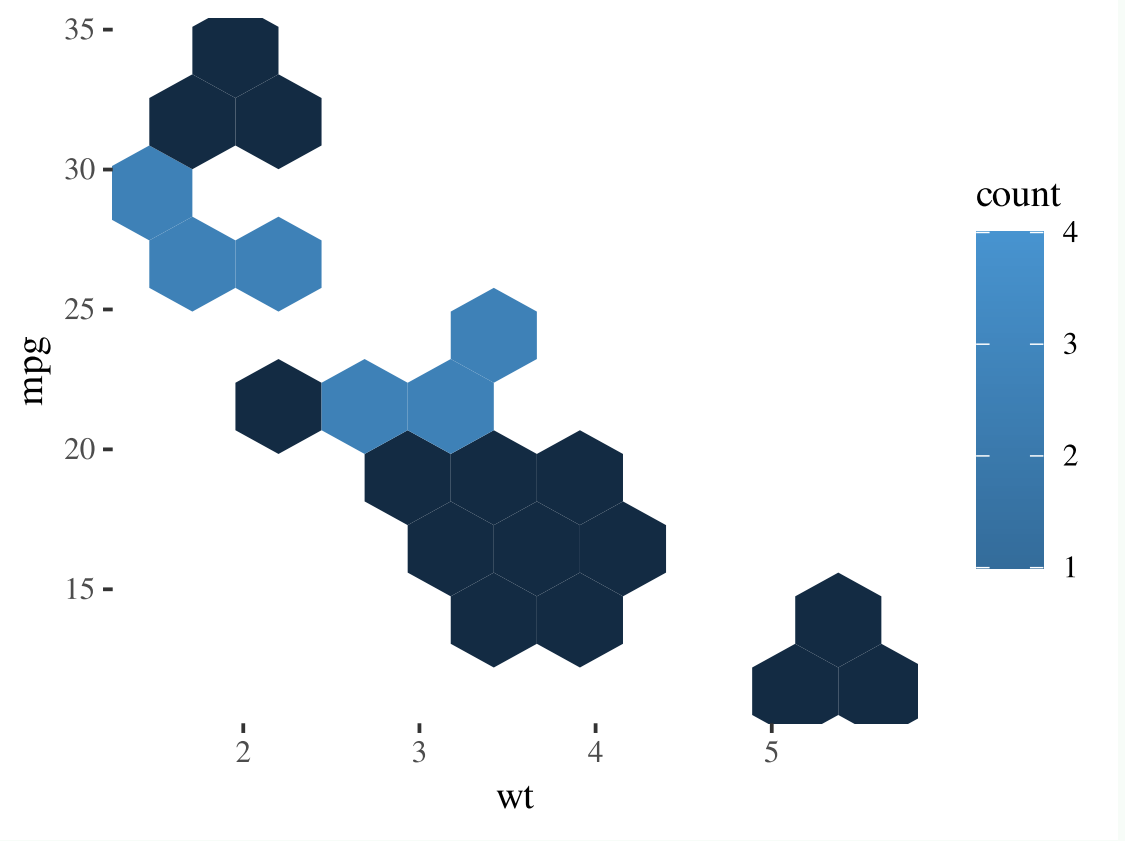


Hex Bins plot

```
# Hex bins plot of mpg vs. weight in the mtcars da
ggplot(mtcars, aes(x = wt, y = mpg))
```

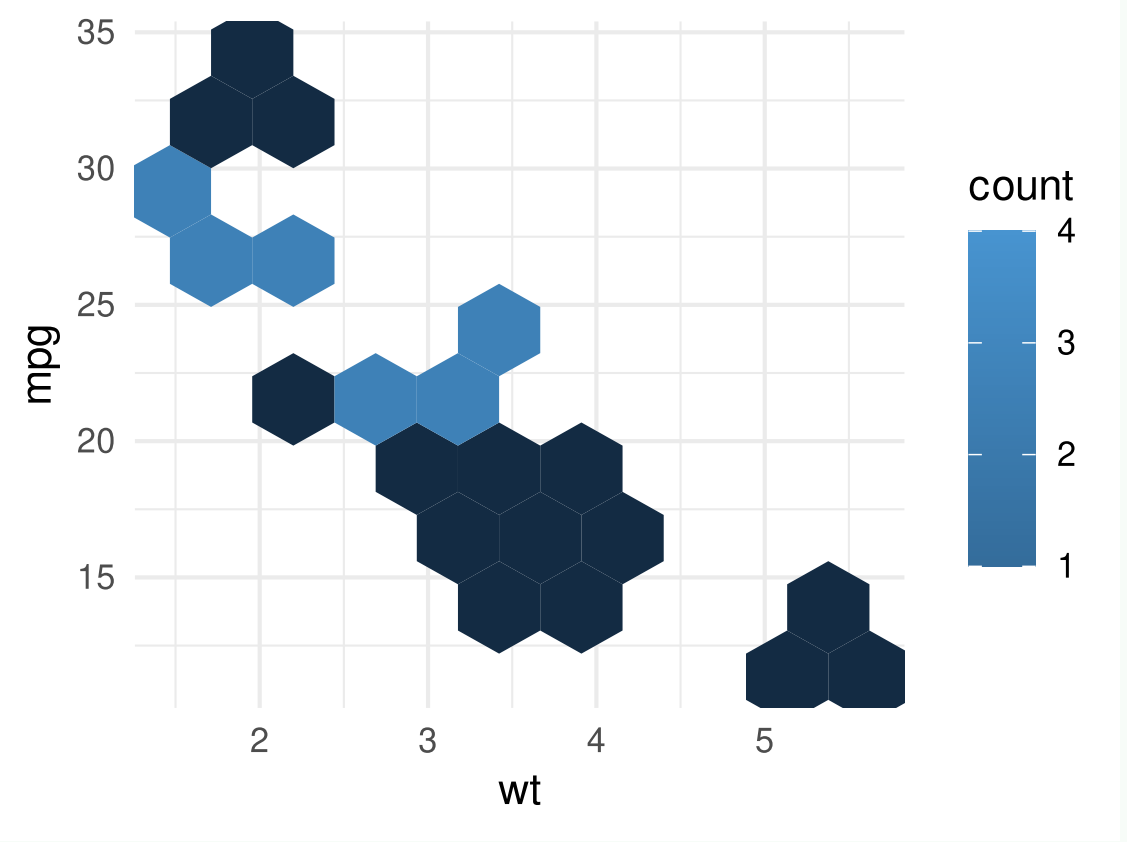
Hex Bins plot

```r
# Hex bins plot of mpg vs. weight in the mtcars da
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_hex(bins = 8)
```



Hex Bins plot

```r
# Hex bins plot of mpg vs. weight in the mtcars da
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_hex(bins = 8) +
  theme_minimal()
```



Error Bars plot

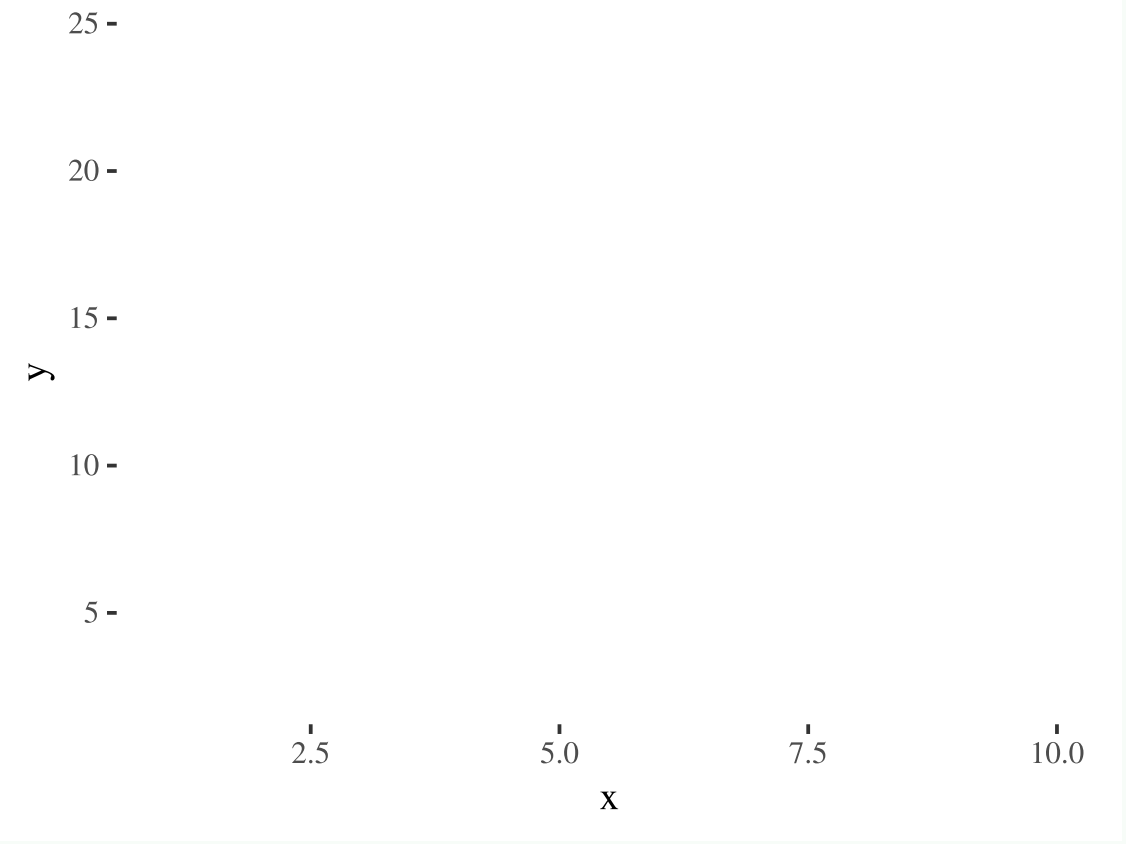
```
# Example data for trend plot
set.seed(42)
```

Error Bars plot

```r
# Example data for trend plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)
```

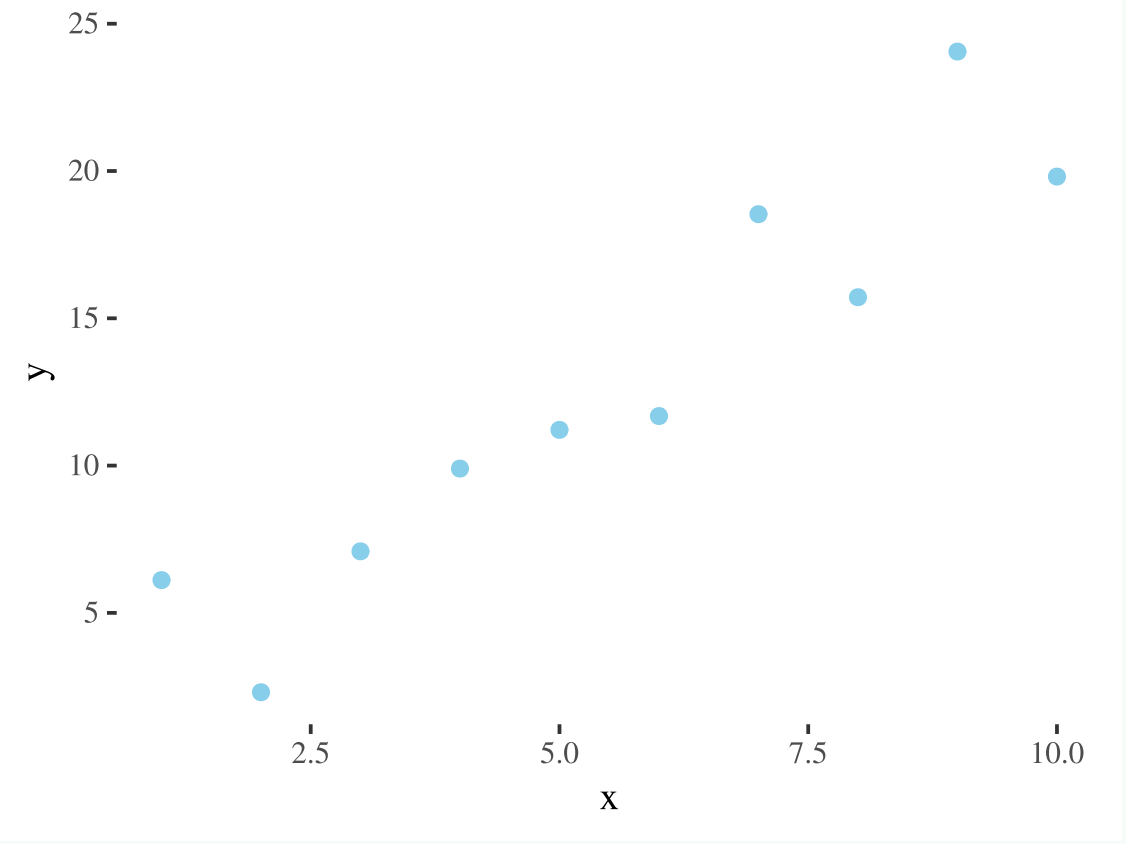
Error Bars plot

```r
# Example data for trend plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

# Trend plot with error bars
ggplot(example_data, aes(x = x, y = y))
```



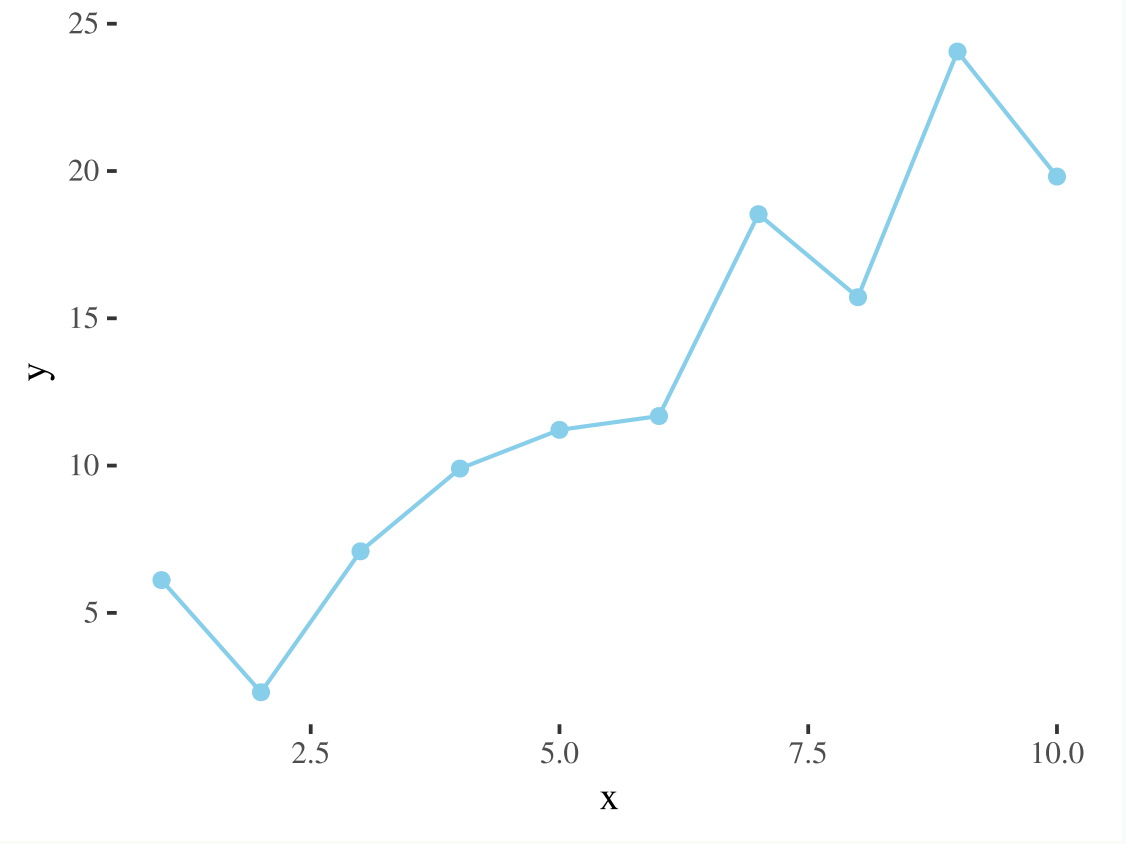
Error Bars plot

```r
# Example data for trend plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

# Trend plot with error bars
ggplot(example_data, aes(x = x, y = y)) +
  geom_point(color = "skyblue", linewidth = 3)
```



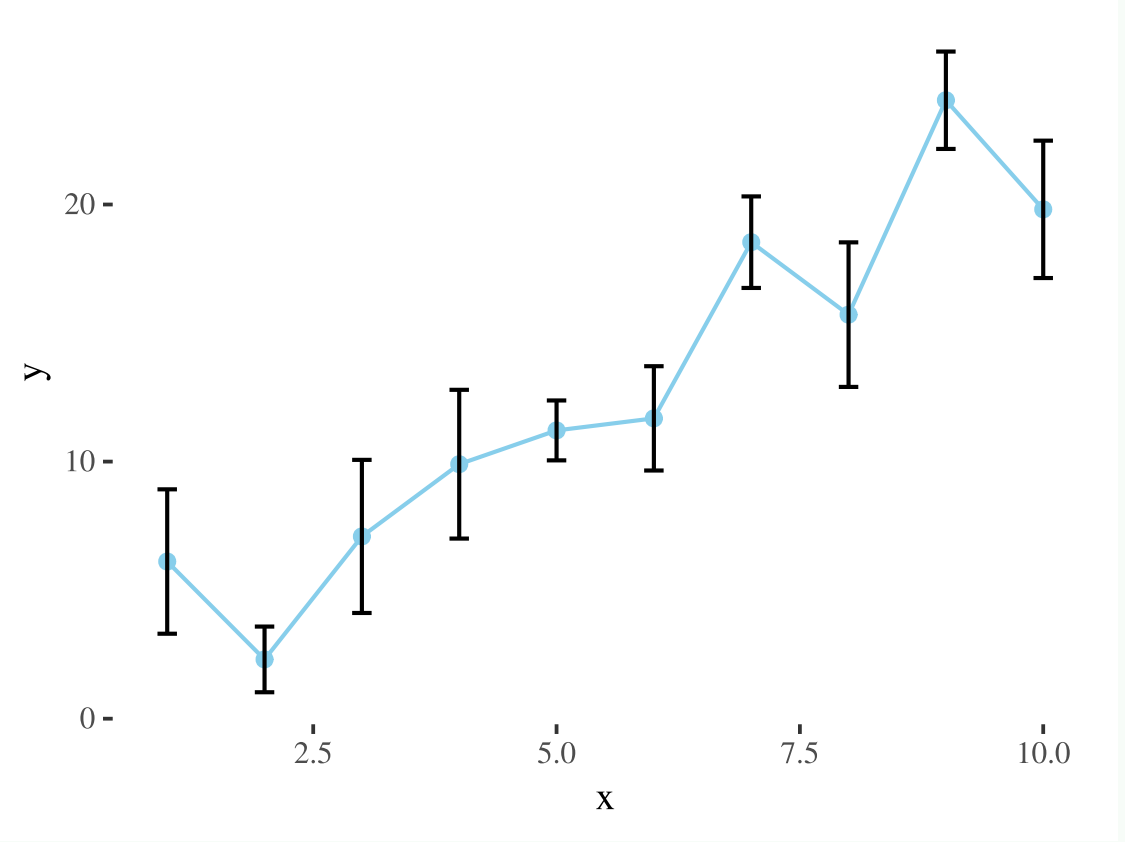
Error Bars plot

```r
# Example data for trend plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

# Trend plot with error bars
ggplot(example_data, aes(x = x, y = y)) +
  geom_point(color = "skyblue", linewidth = 3) +
  geom_line(color = "skyblue")
```



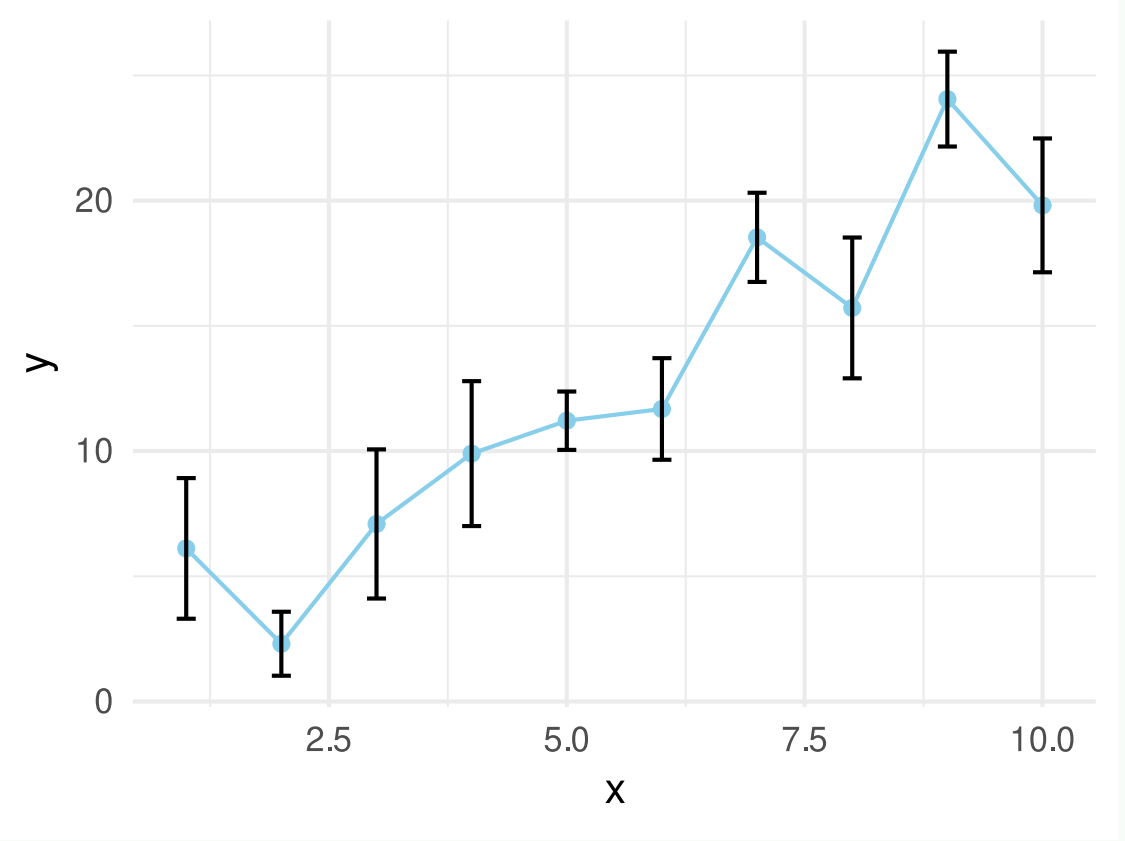
Error Bars plot

```r
# Example data for trend plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

# Trend plot with error bars
ggplot(example_data, aes(x = x, y = y)) +
  geom_point(color = "skyblue", linewidth = 3) +
  geom_line(color = "skyblue") +
  geom_errorbar(aes(ymin = y - se, ymax = y + se),
```



Error Bars plot

```r
# Example data for trend plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

# Trend plot with error bars
ggplot(example_data, aes(x = x, y = y)) +
  geom_point(color = "skyblue", linewidth = 3) +
  geom_line(color = "skyblue") +
  geom_errorbar(aes(ymin = y - se, ymax = y + se),
  theme_minimal()
```



Uncertainty - Confidence Bands

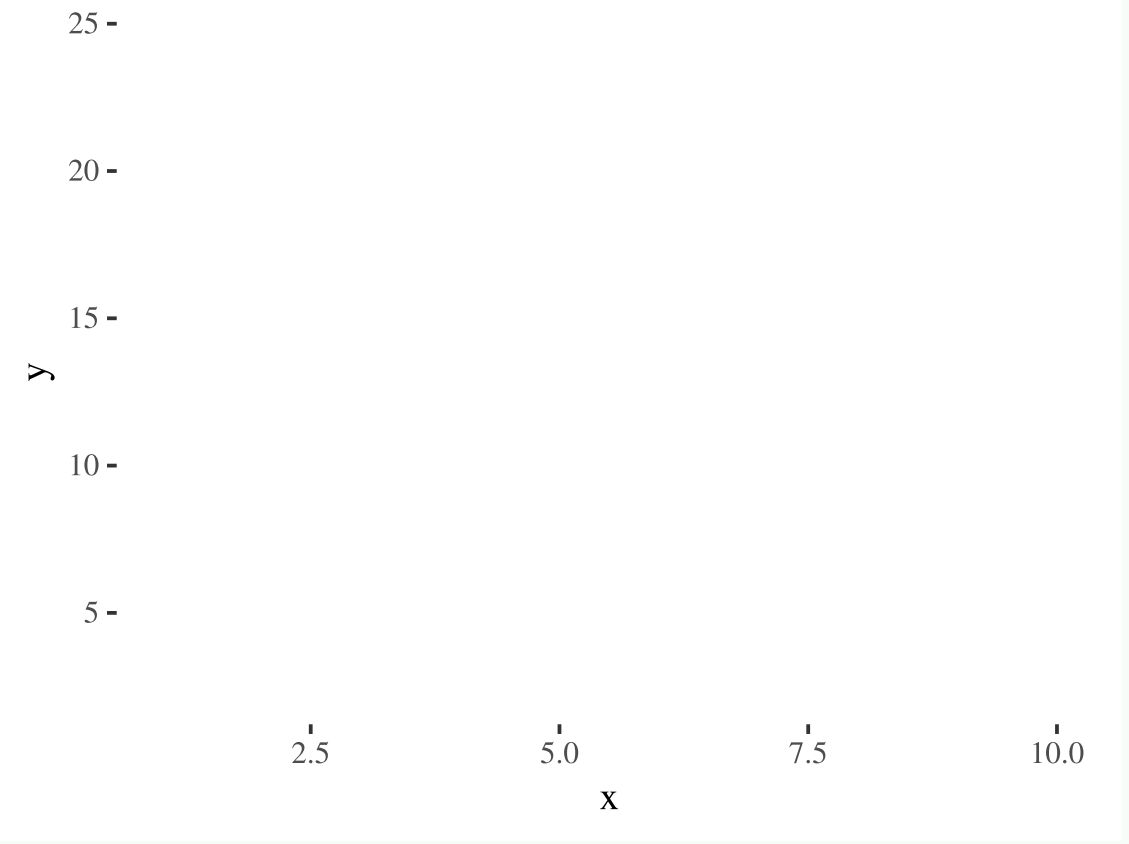
```
# Uncertainty - Confidence Bands plot
set.seed(42)
```

Uncertainty - Confidence Bands

```r
# Uncertainty - Confidence Bands plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)
```

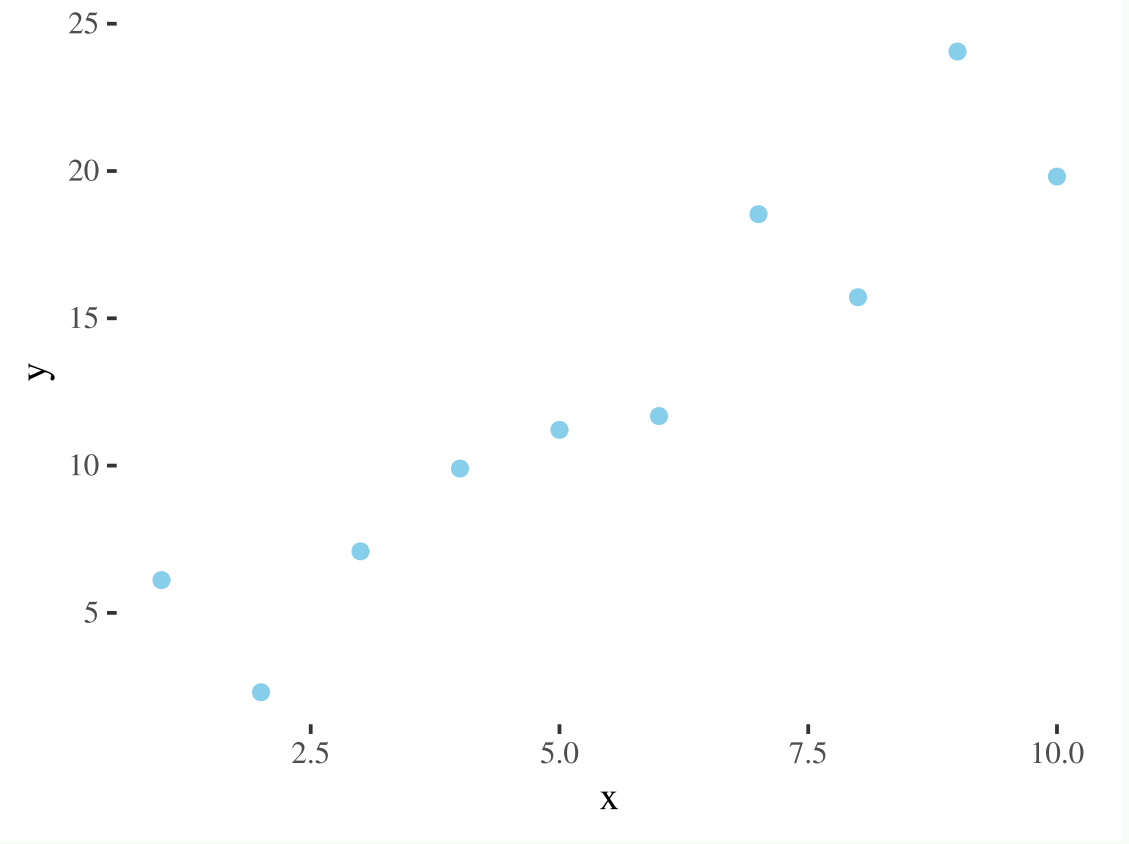
Uncertainty - Confidence Bands

```r
# Uncertainty - Confidence Bands plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

ggplot(example_data, aes(x = x, y = y))
```



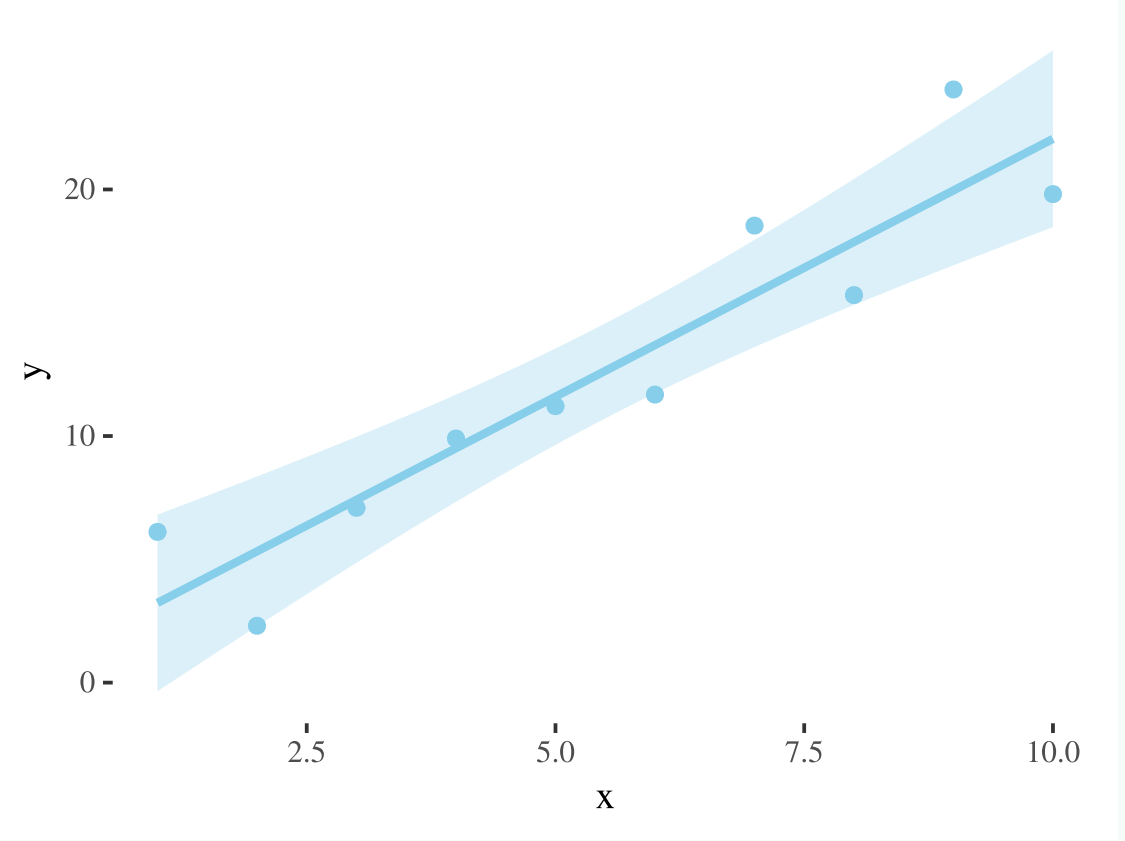
Uncertainty - Confidence Bands

```r
# Uncertainty - Confidence Bands plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

ggplot(example_data, aes(x = x, y = y)) +
  geom_point(color = "skyblue", linewidth = 3)
```



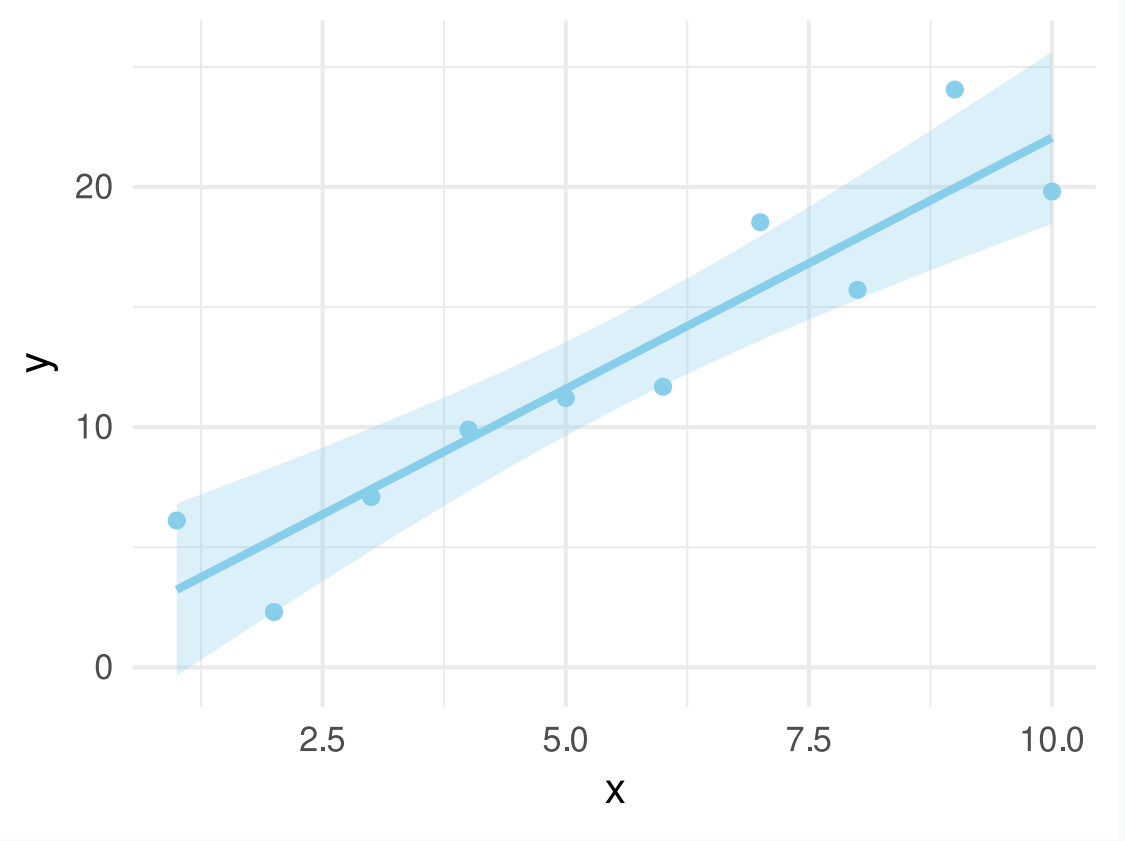
Uncertainty - Confidence Bands

```r
# Uncertainty - Confidence Bands plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

ggplot(example_data, aes(x = x, y = y)) +
  geom_point(color = "skyblue", linewidth = 3) +
  geom_smooth(method = "lm", se = TRUE,
              color = "skyblue",
              fill = "skyblue", alpha = 0.3)
```



Uncertainty - Confidence Bands

```r
# Uncertainty - Confidence Bands plot
set.seed(42)
example_data <- data.frame(
  x = 1:10,
  y = 2 * (1:10) + rnorm(10, mean = 0, sd = 3),
  se = runif(10, min = 1, max = 3)
)

ggplot(example_data, aes(x = x, y = y)) +
  geom_point(color = "skyblue", linewidth = 3) +
  geom_smooth(method = "lm", se = TRUE,
              color = "skyblue",
              fill = "skyblue", alpha = 0.3) +
  theme_minimal()
```

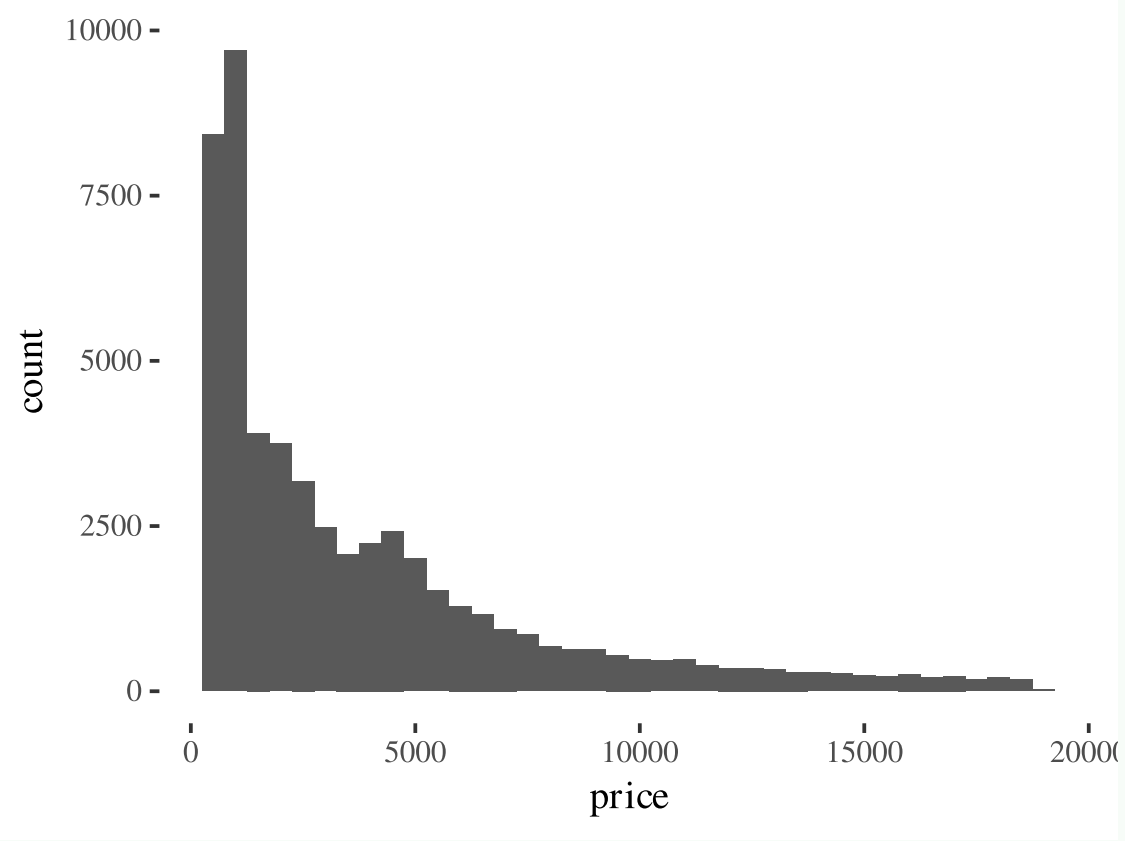


Variable Transformations

```
ggplot(data = diamonds, aes(x = price))
```

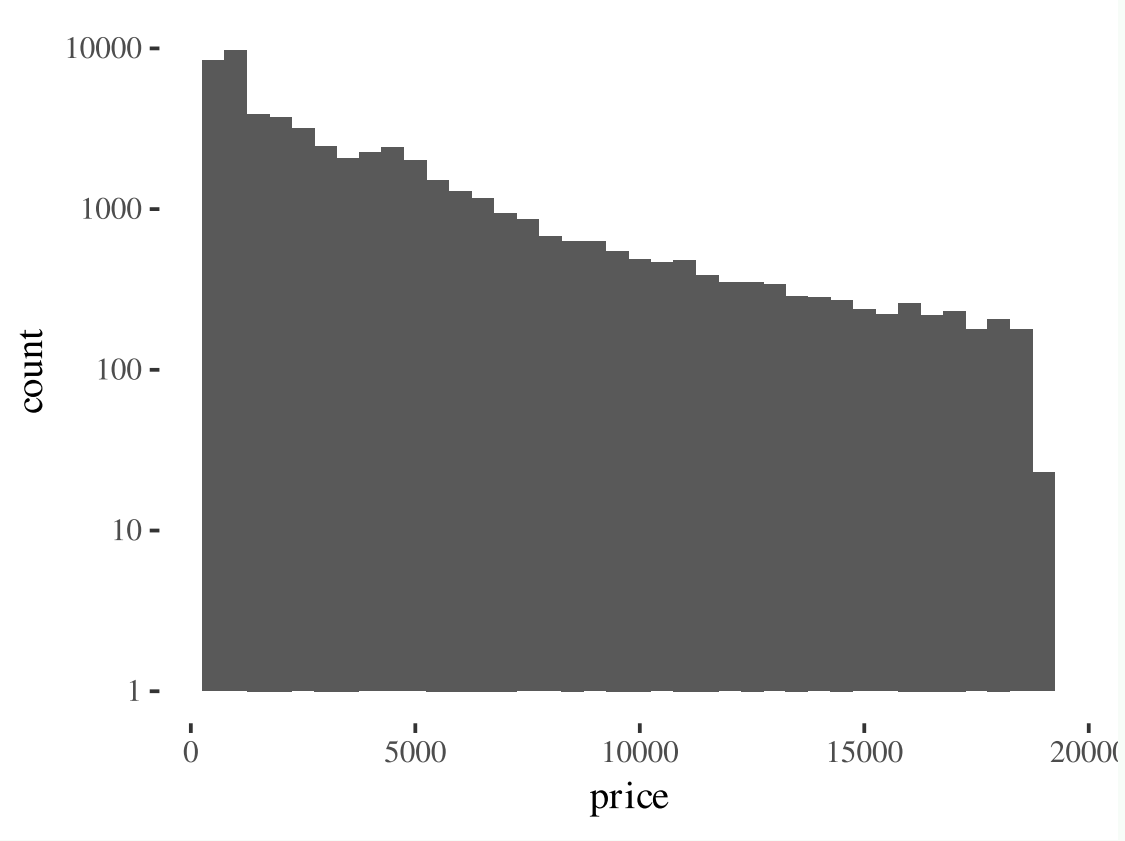
Variable Transformations

```
ggplot(data = diamonds, aes(x = price)) +
  geom_histogram(binwidth = 500)
```



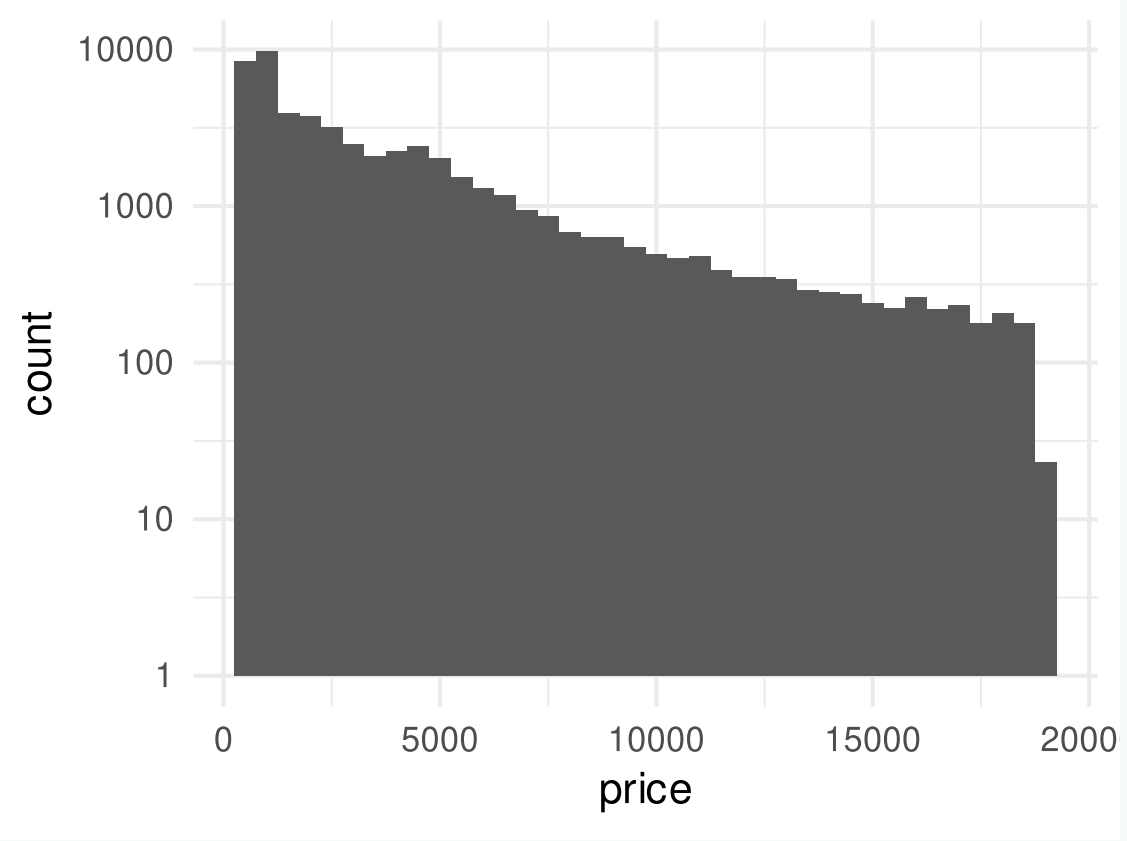
Variable Transformations

```
ggplot(data = diamonds, aes(x = price)) +
  geom_histogram(binwidth = 500) +
  scale_y_log10()
```



Variable Transformations

```
ggplot(data = diamonds, aes(x = price)) +
  geom_histogram(binwidth = 500) +
  scale_y_log10() +
  theme_minimal()
```



Facet

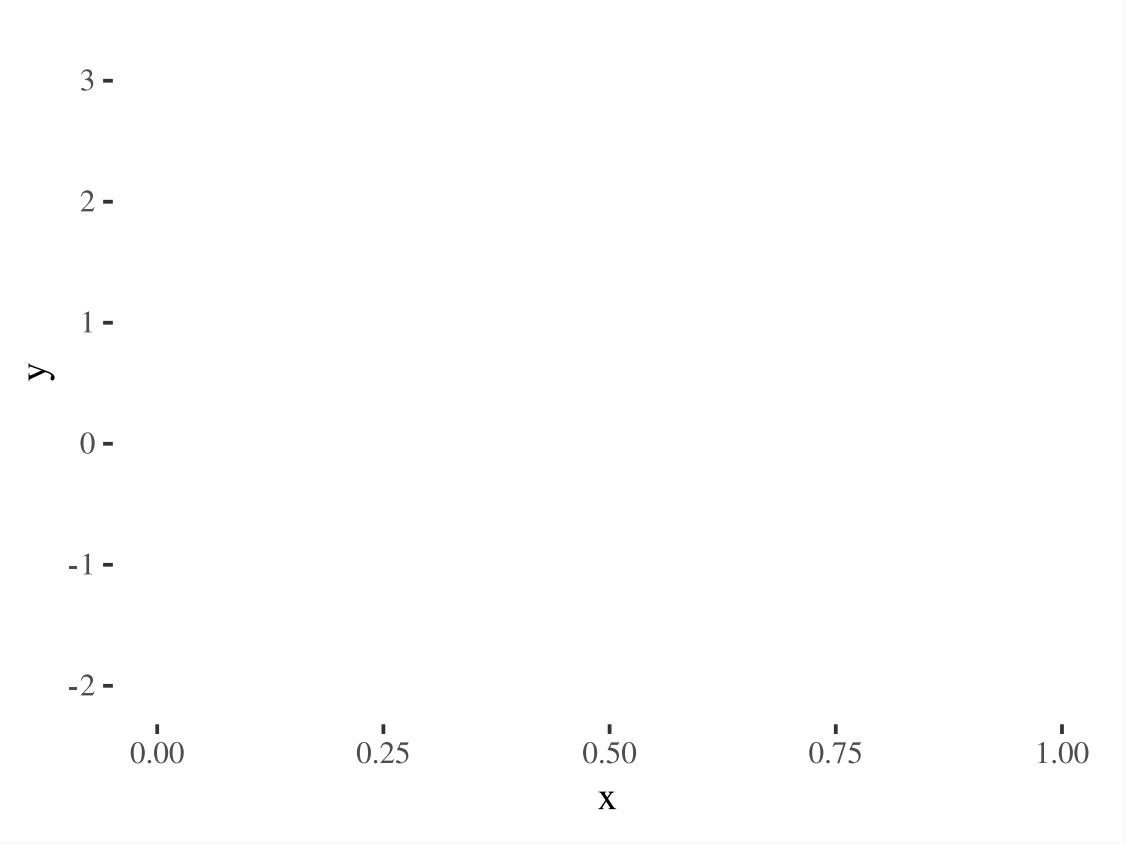
```
# Create example data
set.seed(123)
```

Facet

```r
# Create example data
set.seed(123)
example_data <- data.frame(
  category = factor(rep(c("A", "B", "C", "D"),
                        each = 50)),
  x = runif(200),
  y = rnorm(200)
)
```

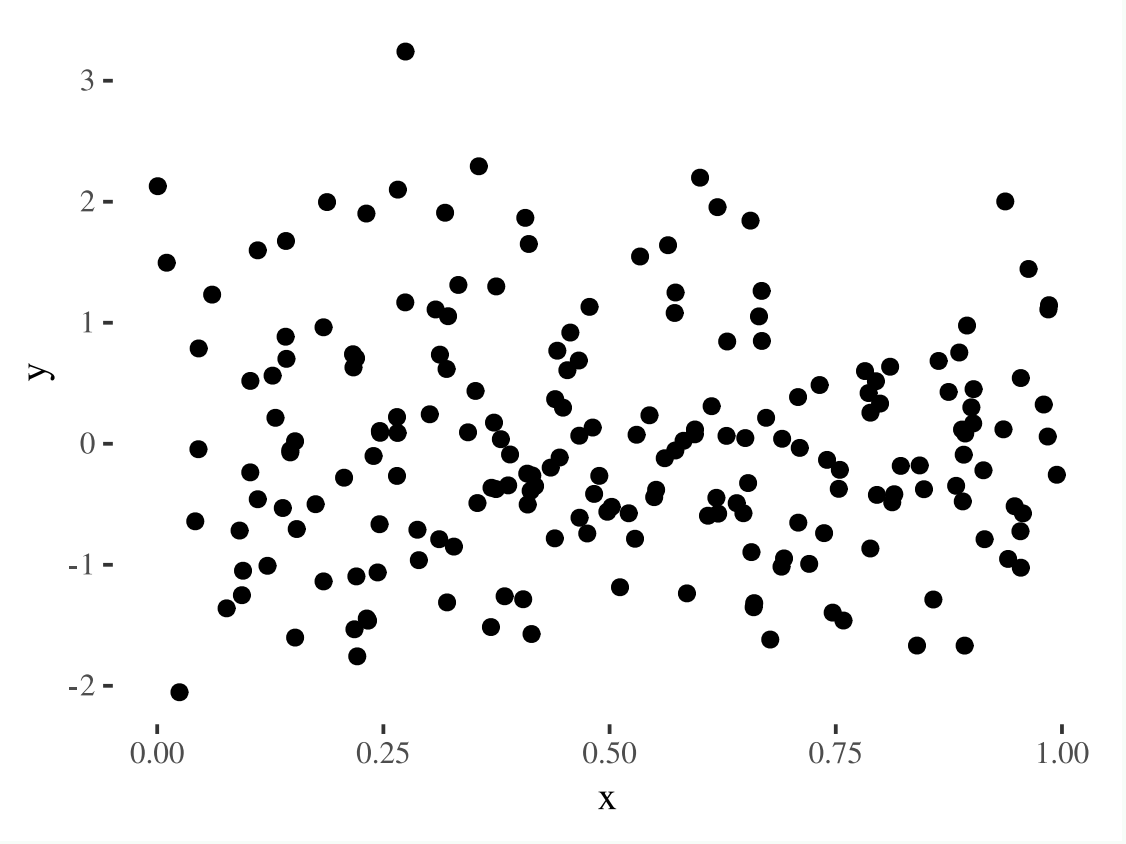
Facet

```r
# Create example data
set.seed(123)
example_data <- data.frame(
  category = factor(rep(c("A", "B", "C", "D"),
                        each = 50)),
  x = runif(200),
  y = rnorm(200)
)

# Create the base plot
ggplot(example_data, aes(x = x, y = y))
```



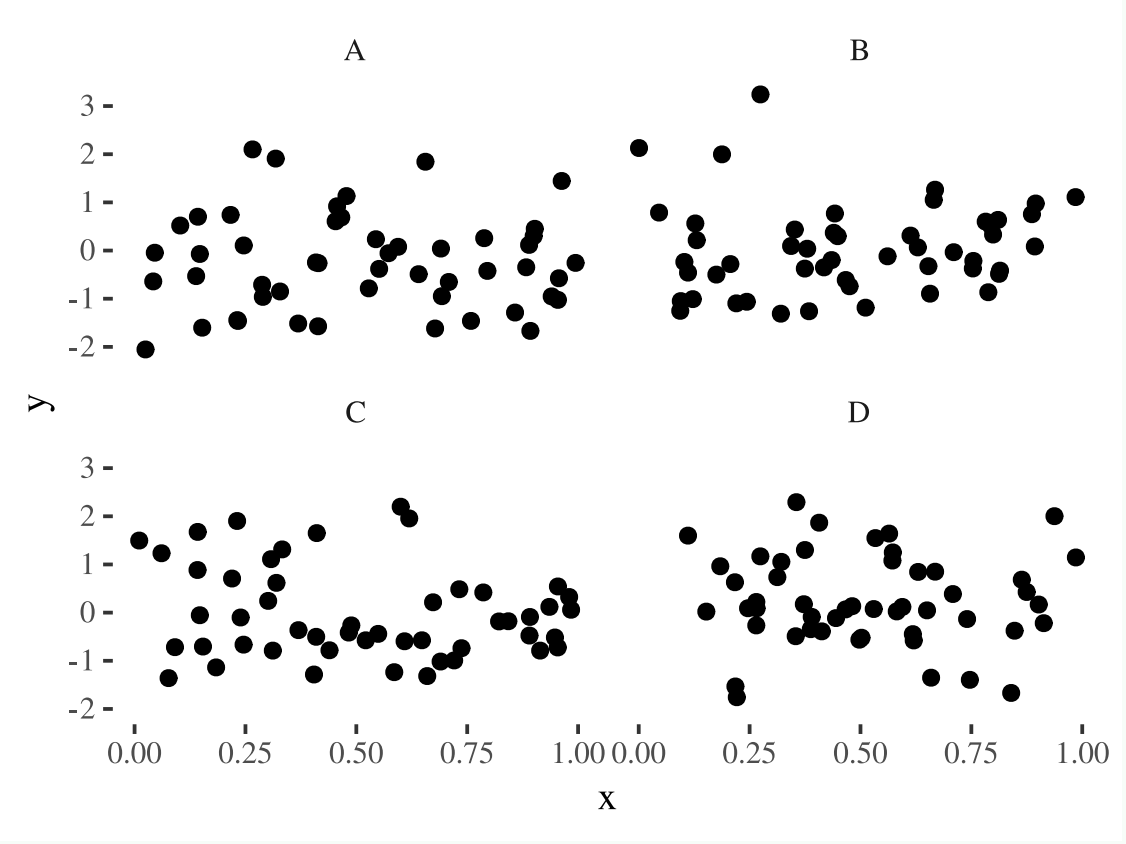
Facet

```r
# Create example data
set.seed(123)
example_data <- data.frame(
  category = factor(rep(c("A", "B", "C", "D"),
                        each = 50)),
  x = runif(200),
  y = rnorm(200)
)

# Create the base plot
ggplot(example_data, aes(x = x, y = y)) +
  geom_point()
```



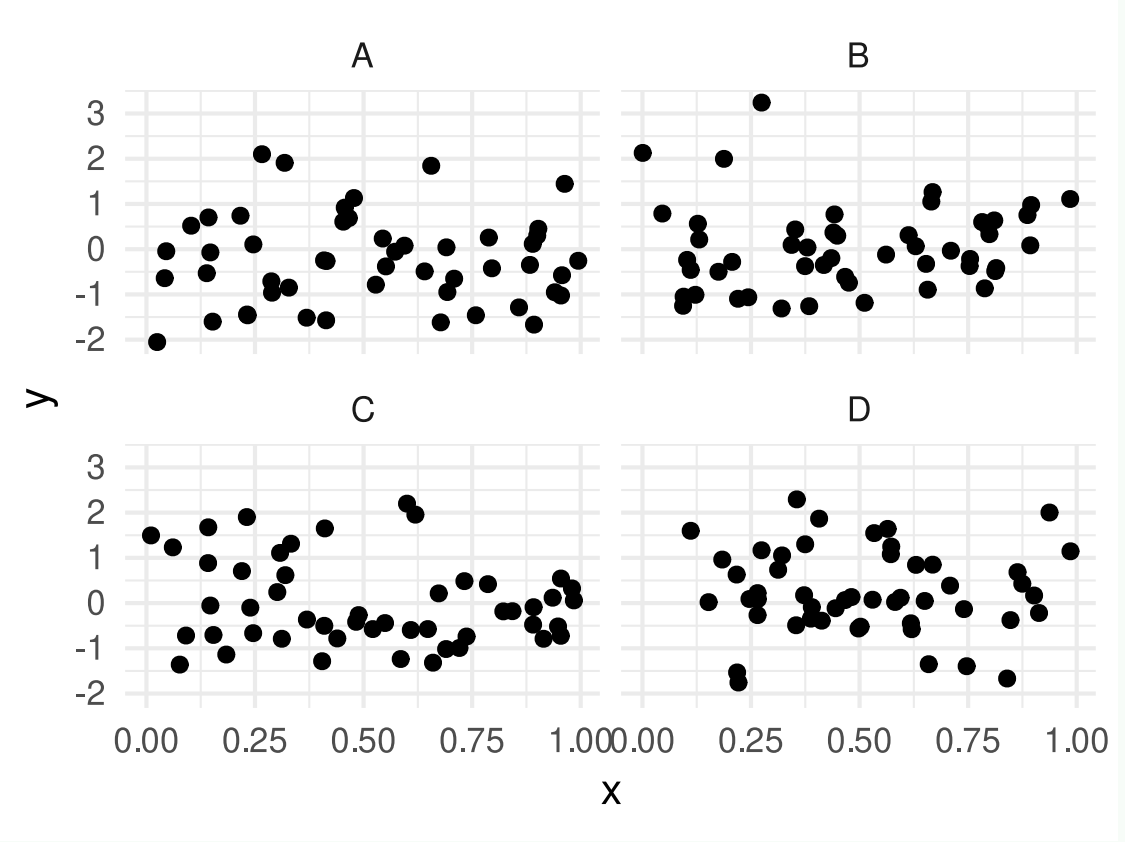
Facet

```r
# Create example data
set.seed(123)
example_data <- data.frame(
  category = factor(rep(c("A", "B", "C", "D"),
                        each = 50)),
  x = runif(200),
  y = rnorm(200)
)

# Create the base plot
ggplot(example_data, aes(x = x, y = y)) +
  geom_point() +
  facet_wrap(~category, nrow = 2)
```



Facet

```r
# Create example data
set.seed(123)
example_data <- data.frame(
  category = factor(rep(c("A", "B", "C", "D"),
                        each = 50)),
  x = runif(200),
  y = rnorm(200)
)

# Create the base plot
ggplot(example_data, aes(x = x, y = y)) +
  geom_point() +
  facet_wrap(~category, nrow = 2) +
  theme_minimal()
```



Hate Crime and income inequality

A FiveThirtyEight article published in 2017 claimed that higher rates of hate crimes were tied to greater income inequality.

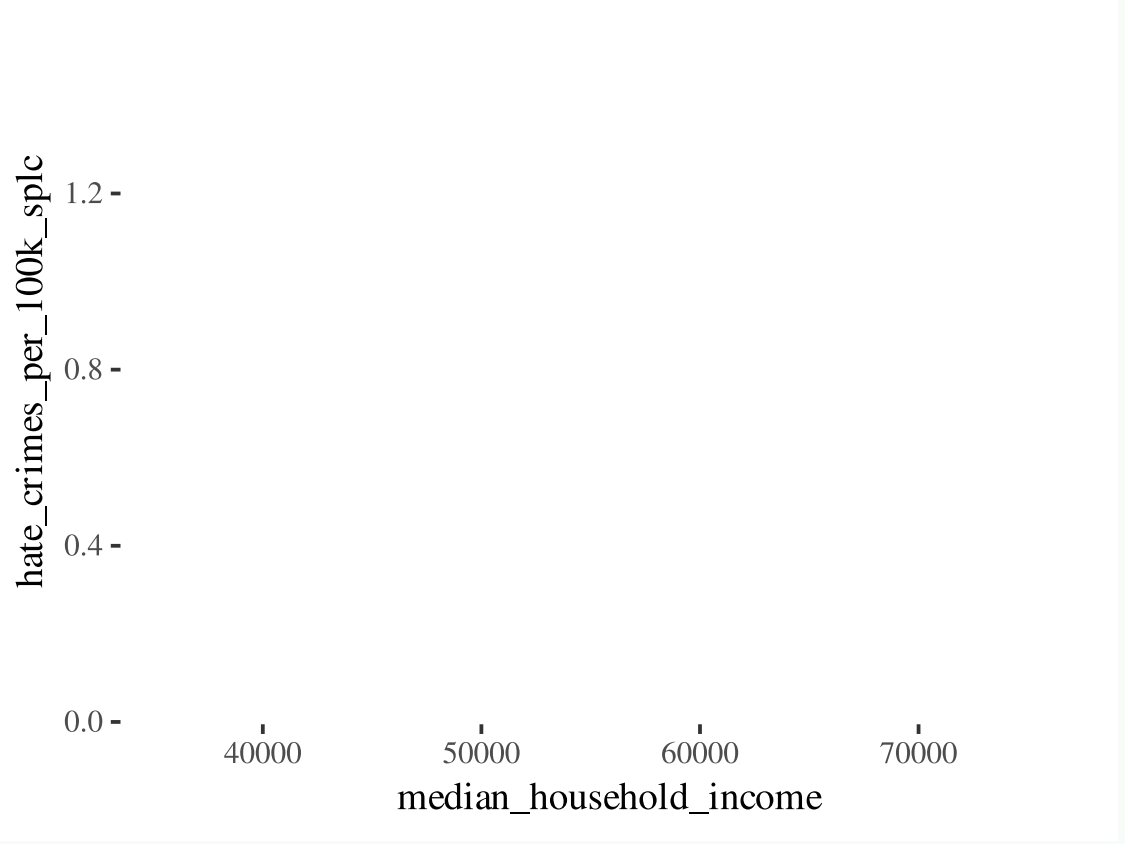
```
hate_crimes <- read_csv("https://raw.githubusercontent.com/deepbas/statdatasets/main/HateCrime.csv")
Rows: 51
Columns: 17
$ ...1                                  <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10…
$ state                                 <chr> "Alabama", "Alaska", "Arizona…
$ median_household_income               <dbl> 42278, 67629, 49254, 44922, 6…
$ share_unemployed_seasonal             <dbl> 0.060, 0.064, 0.063, 0.052, 0…
$ share_population_in_metro_areas       <dbl> 0.64, 0.63, 0.90, 0.69, 0.97,…
$ share_population_with_high_school_degree <dbl> 0.821, 0.914, 0.842, 0.824, 0…
$ share_non_citizen                     <dbl> 0.02, 0.04, 0.10, 0.04, 0.13,…
$ share_white_poverty                   <dbl> 0.12, 0.06, 0.09, 0.12, 0.09,…
$ gini_index                            <dbl> 0.472, 0.422, 0.455, 0.458, 0…
$ share_non_white                       <dbl> 0.35, 0.42, 0.49, 0.26, 0.61,…
$ share_voters_voted_trump              <dbl> 0.63, 0.53, 0.50, 0.60, 0.33,…
$ hate_crimes_per_100k_splc             <dbl> 0.12583893, 0.14374012, 0.225…
$ avg_hatecrimes_per_100k_fbi           <dbl> 1.8064105, 1.6567001, 3.41392…
$ state_code                            <chr> "AL", "AK", "AZ", "AR", "CA",…
$ region                                <chr> "South", "West", "West", "Sou…
$ division                              <chr> "East South Central", "Pacifi…
$ support                               <chr> "Trump", "Trump", "Split", "T…
```

Layering geoms

```
base <- ggplot(hate_crimes,
               aes(x=median_household_income,
                   y=hate_crimes_per_100k_splc))
```

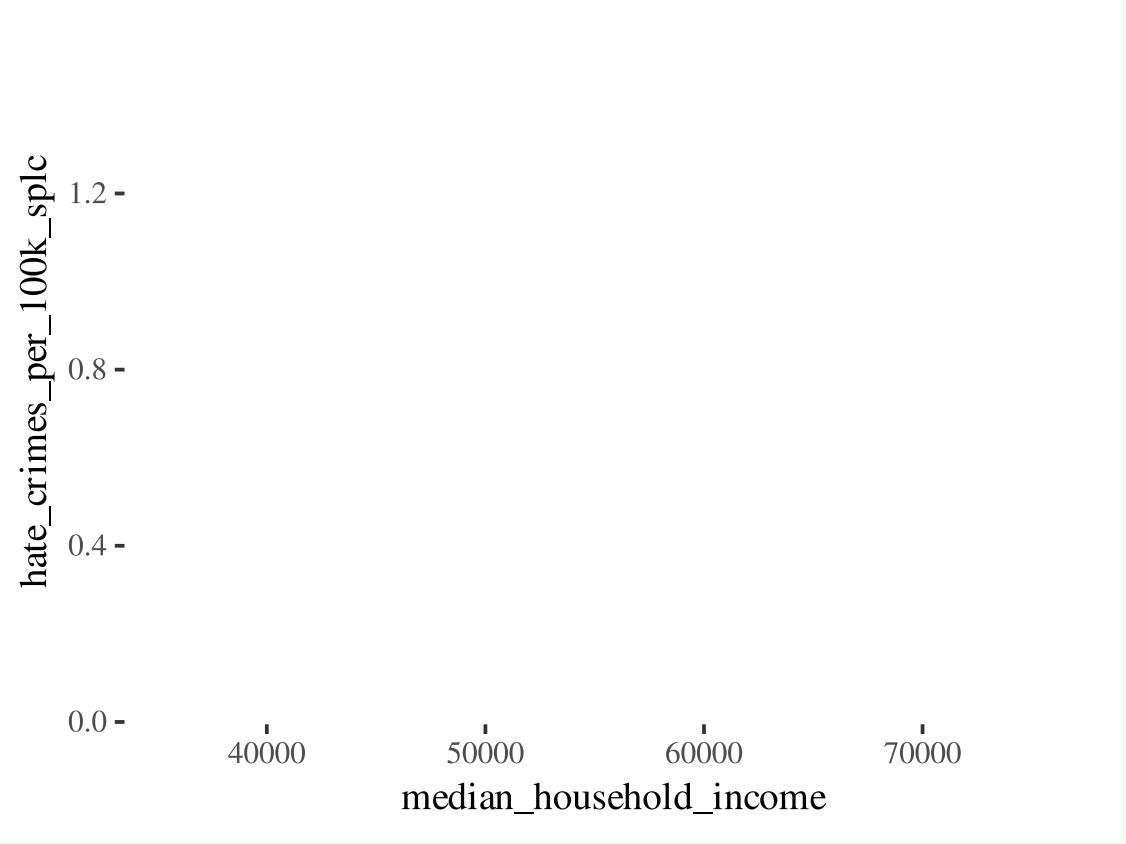
Layering geoms

```
base <- ggplot(hate_crimes,
               aes(x=median_household_income,
                   y=hate_crimes_per_100k_splc))
base
```



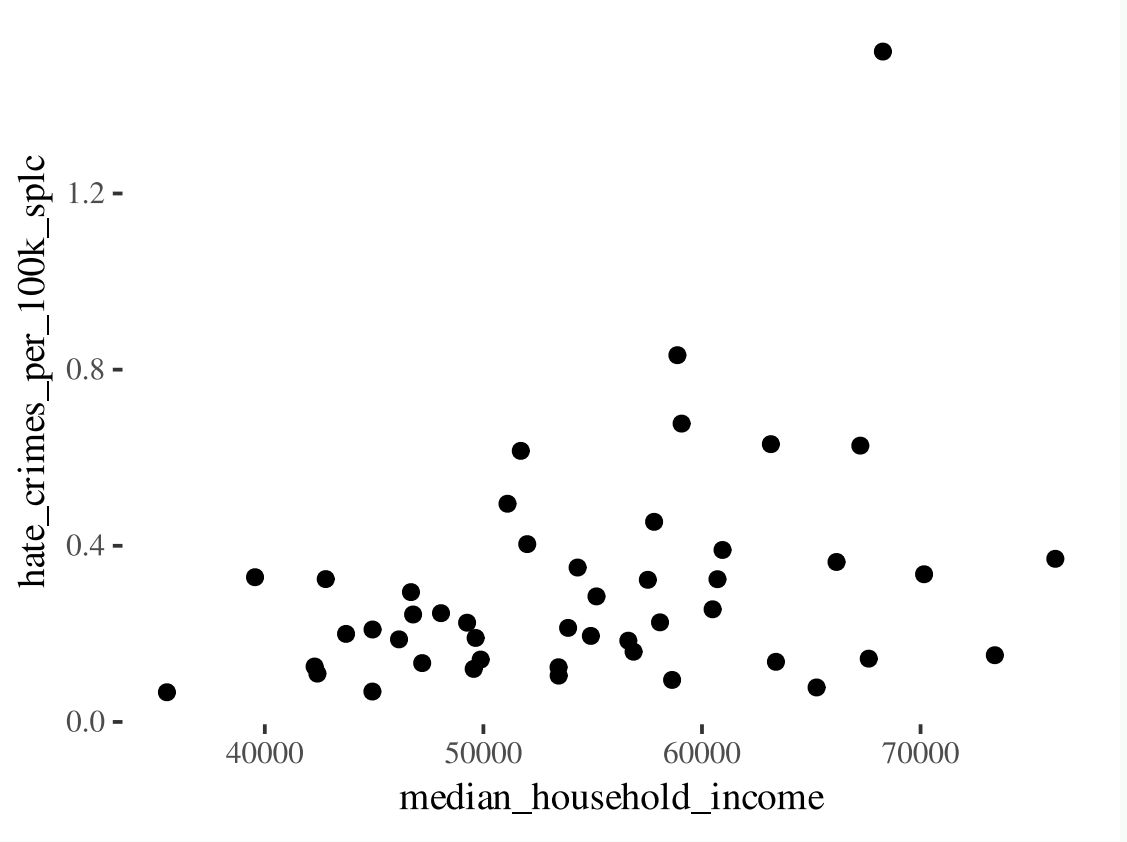
Layering geoms

```
base
```



Layering geoms

```
base +
  geom_point()
```

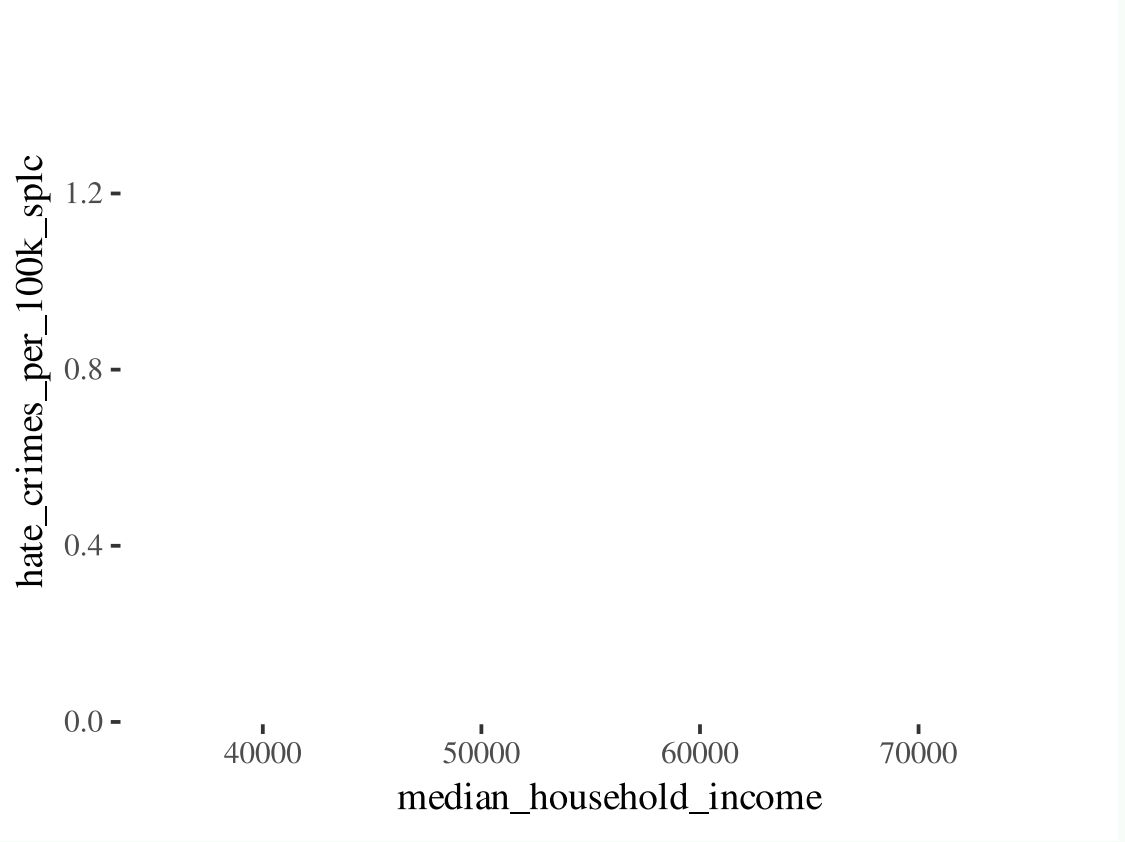


A better plot

```
library(ggrepel)
```

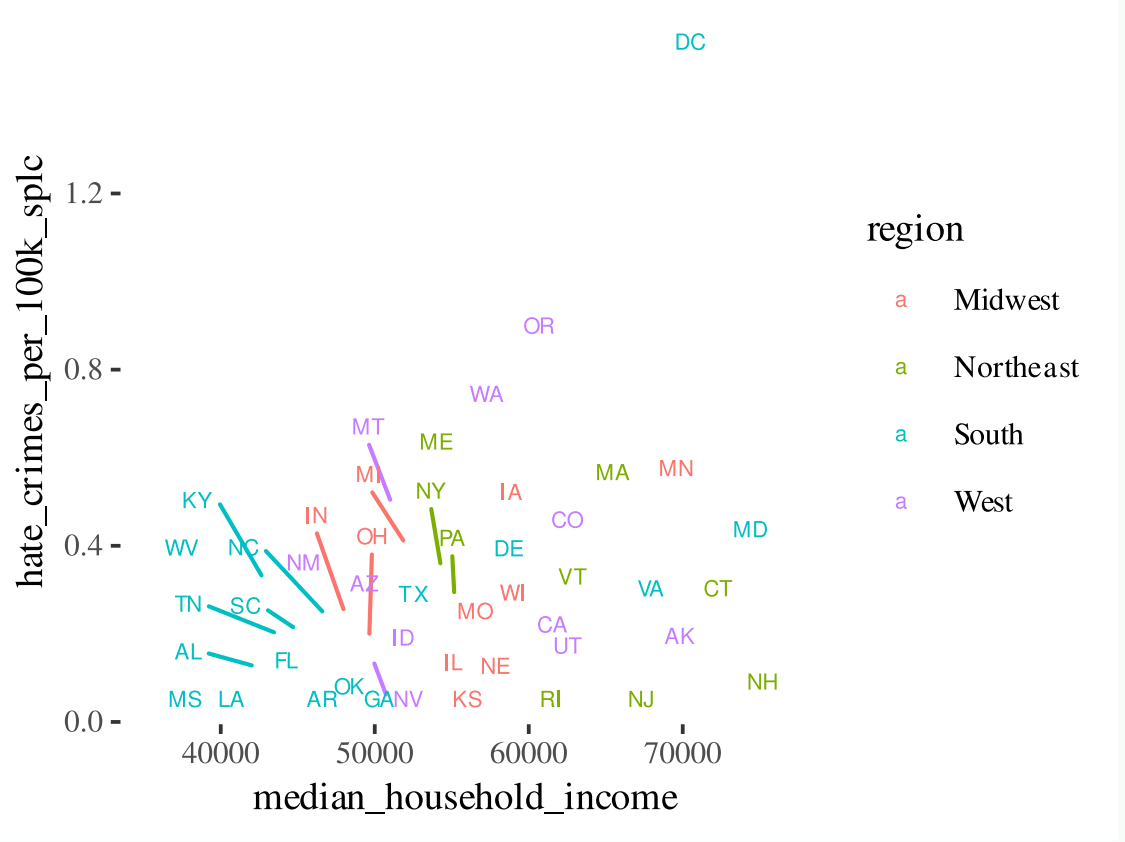
A better plot

```
library(ggrepel)
base
```



A better plot

```
library(ggrepel)
base +
  geom_text_repel(aes(label=state_code,
                      color=region),
                  size = 2)
```

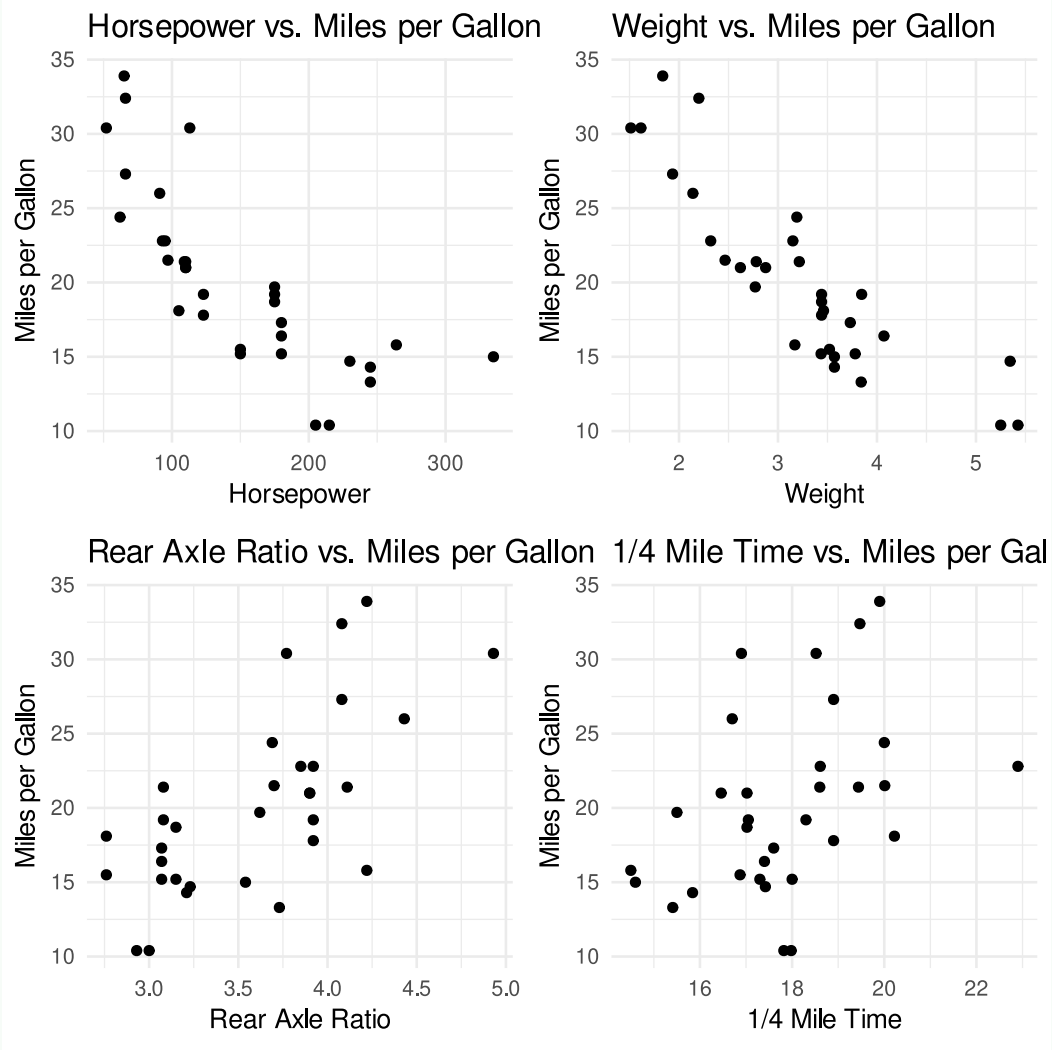


Multi-panel plots in a grid

```
plot1 <- ggplot(data = mtcars, aes(x = hp, y = mpg)
  geom_point() +
  labs(title = "Horsepower vs. Miles per Gallon",
       x = "Horsepower",
       y = "Miles per Gallon") +
  theme(plot.title = element_text(size = 6)) +
  theme_minimal()

plot2 <- ggplot(data = mtcars, aes(x = wt, y = mpg)
  geom_point() +
  labs(title = "Weight vs. Miles per Gallon",
       x = "Weight",
       y = "Miles per Gallon") +
  theme(plot.title = element_text(size = 6)) +
  theme_minimal()
```

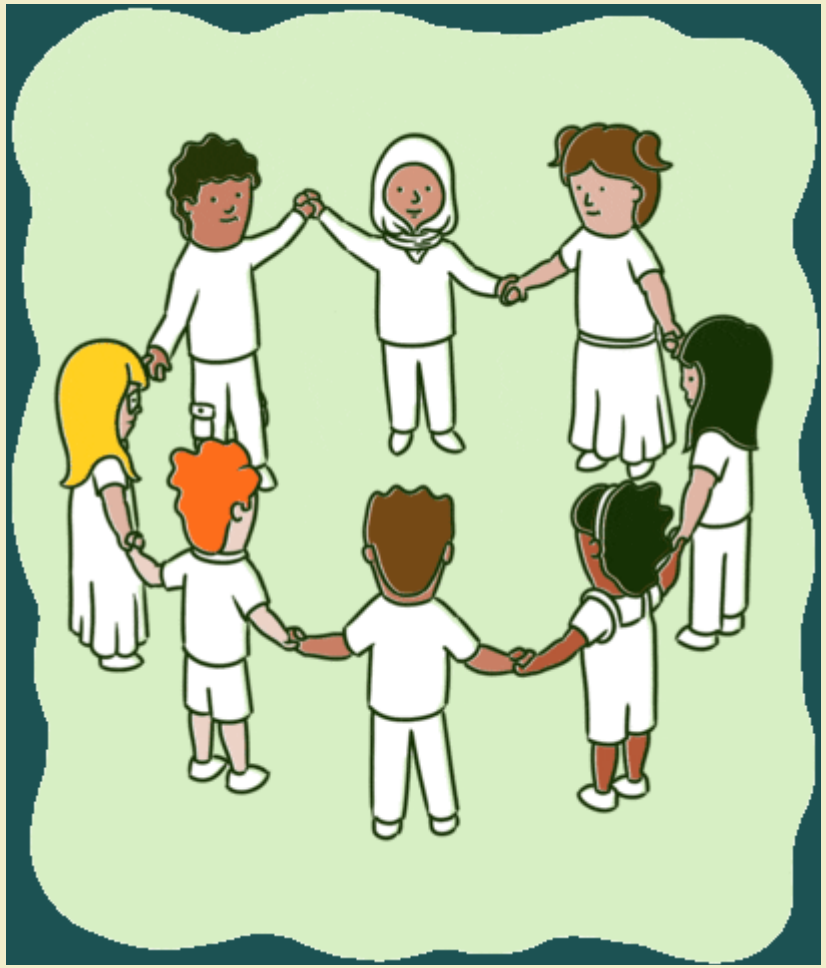
```
plot3 <- ggplot(data = mtcars, aes(x = drat, y = mp
  geom_point() +
  labs(title = "Rear Axle Ratio vs. Miles per Gallo
       x = "Rear Axle Ratio",
       y = "Miles per Gallon") +
  theme(plot.title = element_text(size = 6)) +
  theme_minimal()

plot4 <- ggplot(data = mtcars, aes(x = qsec, y = mp
  geom_point() +
  labs(title = "1/4 Mile Time vs. Miles per Gallon"
       x = "1/4 Mile Time",
       y = "Miles per Gallon") +
  theme(plot.title = element_text(size = 6)) +
  theme_minimal()
```

```
library(gridExtra)
grid.arrange(plot1, plot2, plot3, plot4, nrow = 2)
```



GROUP ACTIVITY 1

10:00



- Let's go over to maize server/ local Rstudio and our class moodle
- Get the class activity 4 file
- Please work on the problems
- Ask me questions