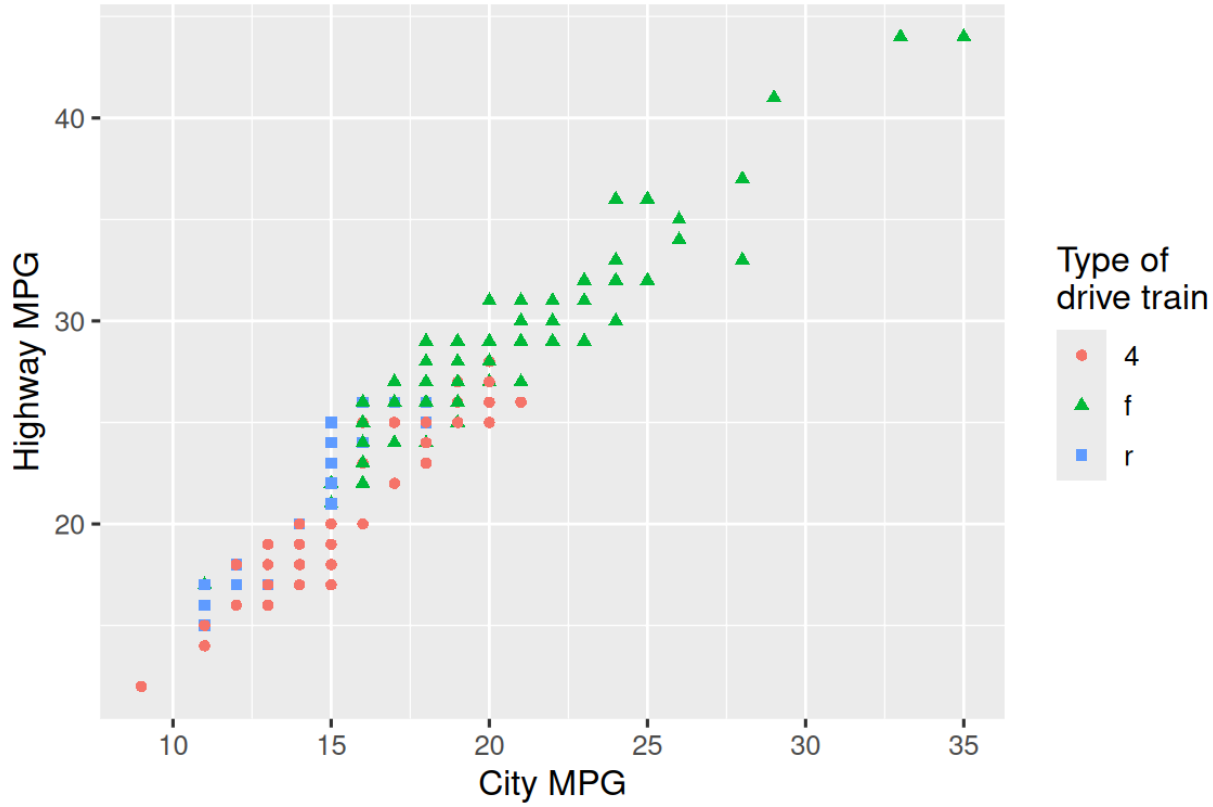
CHAPTER 11 EXERCISES & EXECUTABLES

11.2.1 Exercises

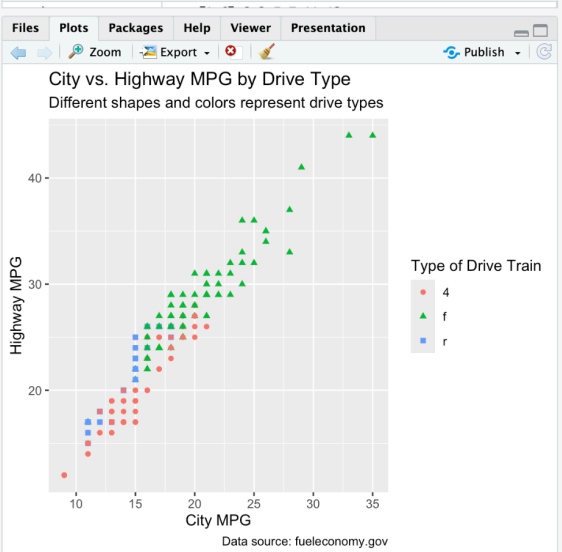
1. Create one plot on the fuel economy data with customized `title`, `subtitle`, `caption`, `x`, `y`, and `color` labels.

2. Recreate the following plot using the fuel economy data. Note that both the colors and shapes of points vary by type of drive train.




```
ggplot(mpg, aes(x = cty, y = hwy, color = drv, shape = drv)) +
  geom_point() +
  labs(
    title = "City vs. Highway MPG by Drive Type",
    subtitle = "Different shapes and colors represent drive types",
    x = "City MPG",
    y = "Highway MPG",
    color = "Type of Drive Train",
    shape = "Type of Drive Train",
    caption = "Data source: fueleconomy.gov"
  )
```

```
8: In simpleLoess(y, x, w, span, degree = degree, parametric = parametric,  :
  reciprocal condition number  0
9: In simpleLoess(y, x, w, span, degree = degree, parametric = parametric,  :
  There are other near singularities as well. 0.25
> ggplot(mpg, aes(x = cty, y = hwy, color = drv, shape = drv)) +
+   geom_point() +
+   labs(
+     title = "City vs. Highway MPG by Drive Type",
+     subtitle = "Different shapes and colors represent drive types",
+     x = "City MPG",
+     y = "Highway MPG",
+     color = "Type of Drive Train",
+     shape = "Type of Drive Train",
+     caption = "Data source: fueleconomy.gov"
+   )
>
```

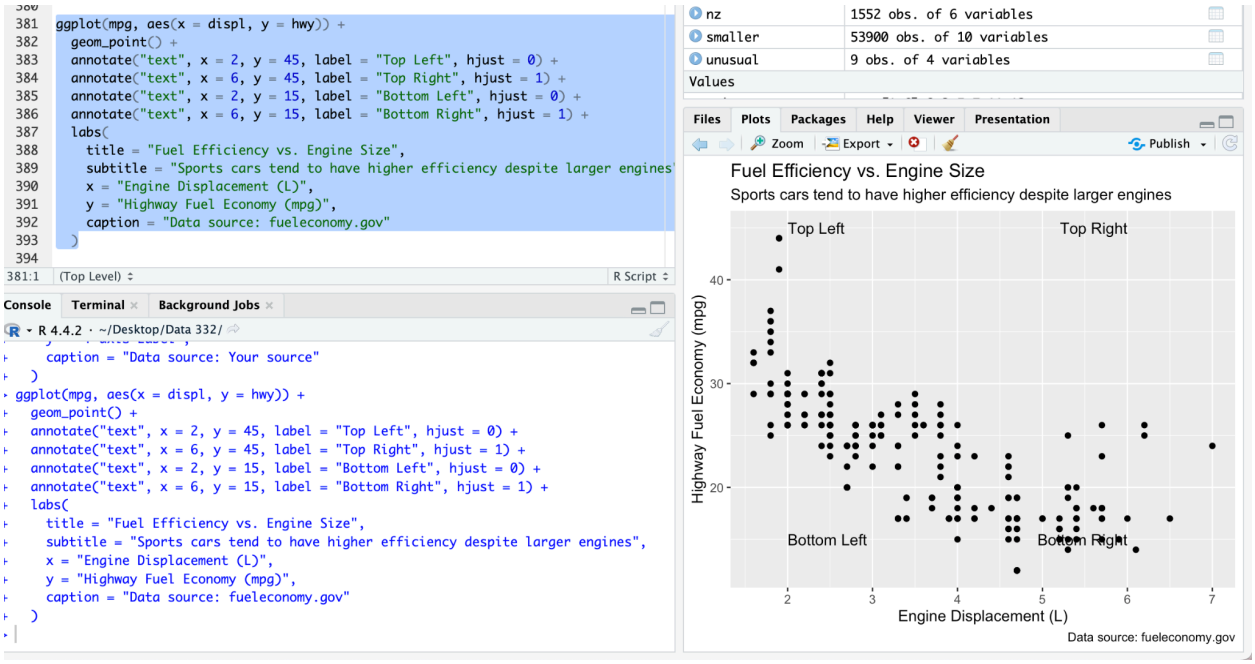


3. Take an exploratory graphic that you've created in the last month, and add informative titles to make it easier for others to understand.

-

11.3.1 Exercises

1. Use geom_text() with infinite positions to place text at the four corners of the plot.



-

2. Use annotate() to add a point geom in the middle of your last plot without having to create a tibble. Customize the shape, size, or color of the point.

```
- ggplot(mpg, aes(x = displ, y = hwy)) +
-   geom_point() +
-   annotate("point", x = 4, y = 30, color = "red", size = 4, shape = 17) +
-   labs(
-     title = "Fuel Efficiency vs. Engine Size",
-     subtitle = "Sports cars tend to have higher efficiency despite larger engines",
-     x = "Engine Displacement (L)",
-     y = "Highway Fuel Economy (mpg)",
-     caption = "Data source: fueleconomy.gov"
-   )
```

3. How do labels with geom_text() interact with faceting? How can you add a label to a single facet? How can you put a different label in each facet? (Hint: Think about the dataset that is being passed to geom_text().)

- geom_text() works within facets by repeating the same label in each facet when applied to the whole dataset.
- To add a label to a single facet, create a smaller dataset that includes the label and map it only to the relevant facet using facet_*() settings.
- To put different labels in each facet, create a data frame with the labels mapped to the facet variable.

4. What arguments to geom_label() control the appearance of the background box?

- The following geom_label() arguments control the background box appearance:

- **fill** – Sets the background color of the label.
- **color** – Sets the border and text color.
- **size** – Adjusts the text size.
- **label.size** – Sets the size of the label's border.
- **label.padding** – Adjusts the padding between text and the border.
- **label.r** – Sets the label's corner radius for rounded edges.

5. What are the four arguments to arrow()? How do they work? Create a series of plots that demonstrate the most important options.

- The four main arguments to arrow() are:

1. **angle** – Sets the angle of the arrowhead in degrees.
2. **length** – Defines the length of the arrowhead (using unit() for measurement).
3. **ends** – Specifies where the arrowhead appears ("last", "first", or "both").
4. **type** – Sets the type of arrow ("open", "closed", or "both").

- Example plot - ggplot(mpg, aes(x = displ, y = hwy)) +
- geom_point() +
- annotate("segment", x = 2, xend = 4, y = 20, yend = 30,
- arrow = arrow(angle = 30, length = unit(0.2, "inches"), ends = "last", type = "closed"))

11.4.6 Exercises

1. Why doesn't the following code override the default scale?

```
df <- tibble(
  x = rnorm(10000),
  y = rnorm(10000)
)

ggplot(df, aes(x, y)) +
  geom_hex() +
  scale_color_gradient(low = "white", high = "red") +
        coord_fixed()
```

- The code fails because geom_hex() uses fill, not color, for the hexagons. The scale_color_gradient() function adjusts the color aesthetic, not the fill aesthetic. To fix this, we can use scale_fill_gradient() instead.

2. What is the first argument to every scale? How does it compare to labs()?
- The first argument to every scale_*() function specifies which **aesthetic** (like color, fill, x, or y) the scale applies to. For example, scale_color_gradient() modifies the color aesthetic, while scale_x_continuous() modifies the x-axis.

3. Change the display of the presidential terms by:
    1. Combining the two variants that customize colors and x axis breaks.
    2. Improving the display of the y axis.
    3. Labelling each term with the name of the president.
    4. Adding informative plot labels.
    5. Placing breaks every 4 years (this is trickier than it seems!).
- To create a plot of presidential terms with improved display and labels, you need to:
1. **Combine color and x-axis customization** using scale_color_manual() and scale_x_continuous() for breaks.
2. **Improve y-axis display** by adjusting limits or labels with scale_y_continuous().
3. **Label each term** using geom_text() or annotate().
4. **Add informative plot labels** using labs().
5. **Set breaks every 4 years** by adjusting the breaks argument in scale_x_continuous().

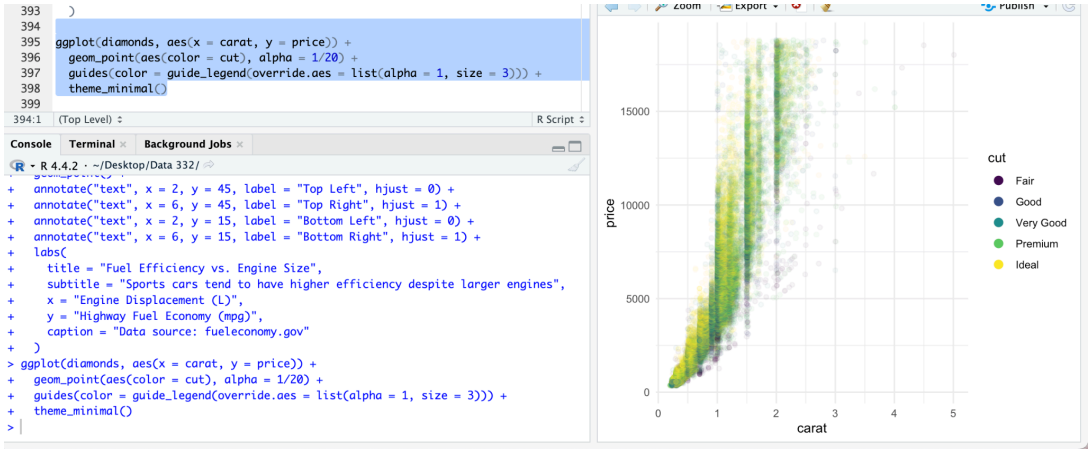
4. First, create the following plot. Then, modify the code using override.aes to make the legend easier to see.

```
ggplot(diamonds, aes(x = carat, y = price)) +
    geom_point(aes(color = cut), alpha = 1/20)
```

- ggplot(diamonds, aes(x = carat, y = price)) +
- geom_point(aes(color = cut), alpha = 1/20) +
- guides(color = guide_legend(override.aes = list(alpha = 1, size = 3)))

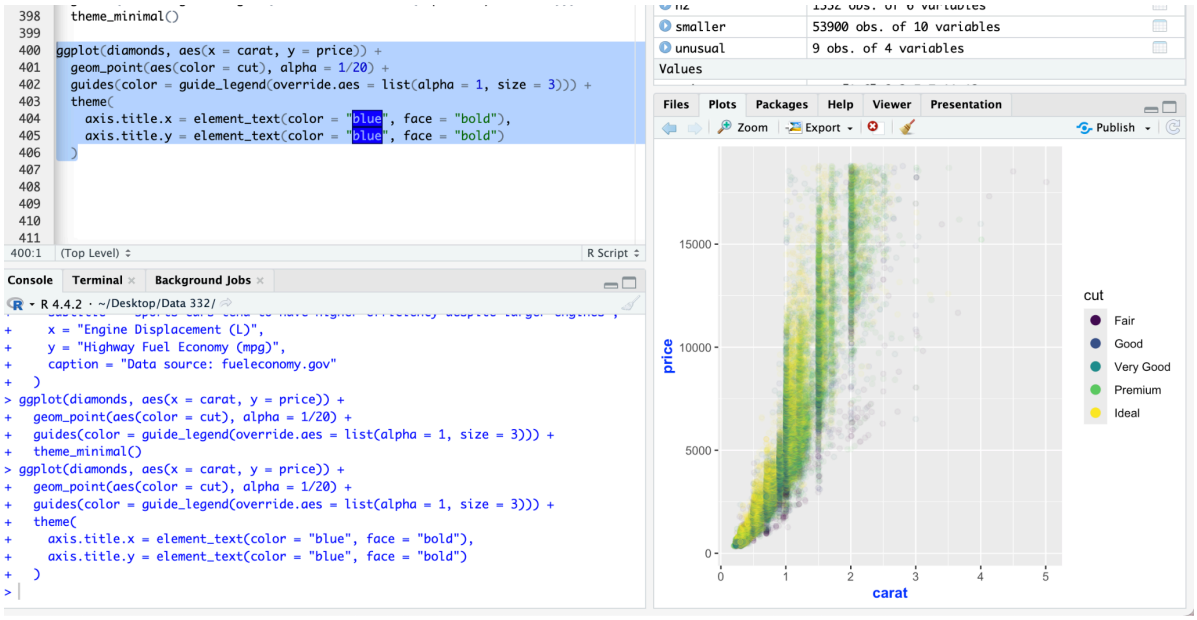
11.5.1 Exercises

1. Pick a theme offered by the ggthemes package and apply it to the last plot you made.



-

2. Make the axis labels of your plot blue and bolded.



-

11.6.1 Exercises

1. What happens if you omit the parentheses in the following plot layout. Can you explain why this happens?

```
p1 <- ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  labs(title = "Plot 1")
p2 <- ggplot(mpg, aes(x = drv, y = hwy)) +
  geom_boxplot() +
  labs(title = "Plot 2")
p3 <- ggplot(mpg, aes(x = cty, y = hwy)) +
  geom_point() +
  labs(title = "Plot 3")
```

```
(p1 | p2) / p3
```

- If you omit the parentheses in (p1 | p2) / p3, R will interpret it as p1 | (p2 / p3), which is not a valid combination of plots because | and / are operators for arranging plots from the **patchwork** package. Parentheses control the order of operations, ensuring that p1 | p2 is processed first as a row, and then combined vertically with p3.

2. Using the three plots from the previous exercise, recreate the following patchwork.

Fig. A:

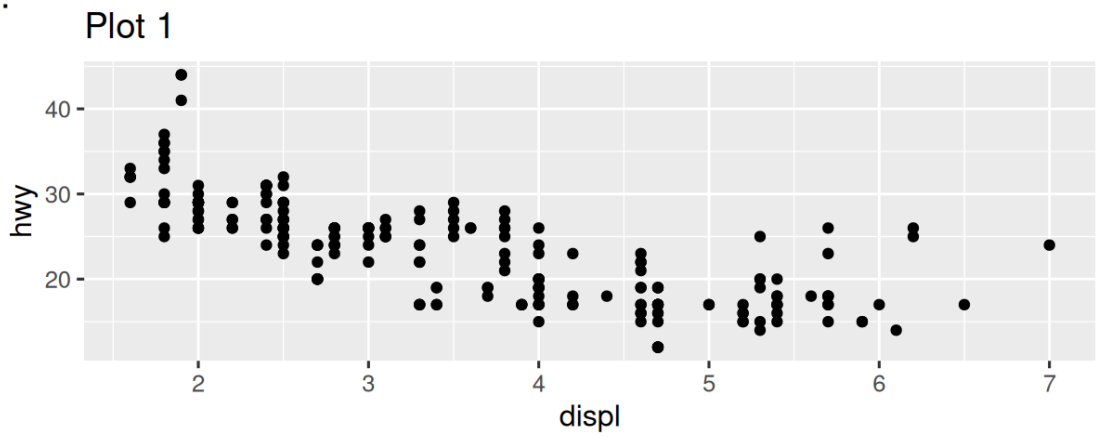


Fig. B:

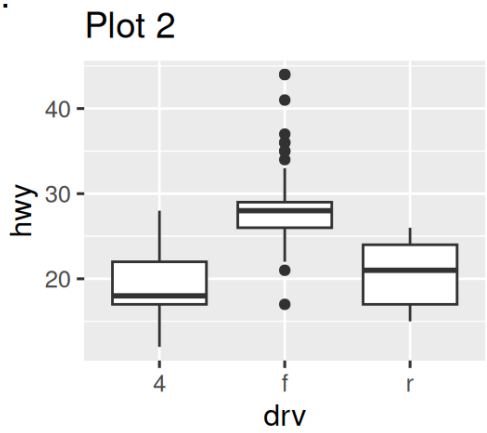


Fig. C:

- If you omit the parentheses, R processes the plot combination from left to right. The operator | has higher precedence than /, so p1 | p2 is combined first into a row layout. The / operator then tries to place p3 below the result. Since p3 is not aligned with p1 | p2 in terms of dimensions, it can cause an error or incorrect layout. Adding parentheses ensures that the combination order is correctly defined.

Executables

#Chapter 11 Communication.....

#11.1.1 Prerequisites

```
library(tidyverse)
library(scales)
library(ggrepel)
library(patchwork)

#11.2 Labels

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  labs(
    x = "Engine displacement (L)",
    y = "Highway fuel economy (mpg)",
    color = "Car type",
    title = "Fuel efficiency generally decreases with engine size",
    subtitle = "Two seaters (sports cars) are an exception because of their light weight",
    caption = "Data from fueleconomy.gov"
  )

df <- tibble(
  x = 1:10,
  y = cumsum(x^2)
)

ggplot(df, aes(x, y)) +
  geom_point() +
  labs(
    x = quote(x[i]),
    y = quote(sum(x[i] ^ 2, i == 1, n))
  )

#11.3 Annotations

label_info <- mpg |>
  group_by(drv) |>
  arrange(desc(displ)) |>
  slice_head(n = 1) |>
  mutate(
    drive_type = case_when(
```

```
      drv == "f" ~ "front-wheel drive",
      drv == "r" ~ "rear-wheel drive",
      drv == "4" ~ "4-wheel drive"
    )
  ) |>
  select(displ, hwy, drv, drive_type)

ggplot(mpg, aes(x = displ, y = hwy, color = drv)) +
  geom_point(alpha = 0.3) +
  geom_smooth(se = FALSE) +
  geom_text(
    data = label_info,
    aes(x = displ, y = hwy, label = drive_type),
    fontface = "bold", size = 5, hjust = "right", vjust = "bottom"
  ) +
  theme(legend.position = "none")

potential_outliers <- mpg |>
  filter(hwy > 40 | (hwy > 20 & displ > 5))

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  geom_text_repel(data = potential_outliers, aes(label = model)) +
  geom_point(data = potential_outliers, color = "red") +
  geom_point(
    data = potential_outliers,
    color = "red", size = 3, shape = "circle open"
  )

trend_text <- "Larger engine sizes tend to have lower fuel economy." |>
  str_wrap(width = 30)
trend_text

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  annotate(
    geom = "label", x = 3.5, y = 38,
    label = trend_text,
    hjust = "left", color = "red"
  ) +
```

```
  annotate(
    geom = "segment",
    x = 3, y = 35, xend = 5, yend = 25, color = "red",
    arrow = arrow(type = "closed")
  )

#11.4 Scales

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = class))

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = class)) +
  scale_x_continuous() +
  scale_y_continuous() +
  scale_color_discrete()

#11.4.2 Axis ticks and legend keys

ggplot(mpg, aes(x = displ, y = hwy, color = drv)) +
  geom_point() +
  scale_y_continuous(breaks = seq(15, 40, by = 5))

ggplot(mpg, aes(x = displ, y = hwy, color = drv)) +
  geom_point() +
  scale_x_continuous(labels = NULL) +
  scale_y_continuous(labels = NULL) +
  scale_color_discrete(labels = c("4" = "4-wheel", "f" = "front", "r" = "rear"))

# Left
ggplot(diamonds, aes(x = price, y = cut)) +
  geom_boxplot(alpha = 0.05) +
  scale_x_continuous(labels = label_dollar())

# Right
ggplot(diamonds, aes(x = price, y = cut)) +
  geom_boxplot(alpha = 0.05) +
  scale_x_continuous(
    labels = label_dollar(scale = 1/1000, suffix = "K"),
    breaks = seq(1000, 19000, by = 6000)
```

```
  )

ggplot(diamonds, aes(x = cut, fill = clarity)) +
  geom_bar(position = "fill") +
  scale_y_continuous(name = "Percentage", labels = label_percent())

presidential |>
  mutate(id = 33 + row_number()) |>
  ggplot(aes(x = start, y = id)) +
  geom_point() +
  geom_segment(aes(xend = end, yend = id)) +
  scale_x_date(name = NULL, breaks = presidential$start, date_labels = "'%y")

#11.4.3 Legend layout

base <- ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = class))

base + theme(legend.position = "right") # the default
base + theme(legend.position = "left")
base +
  theme(legend.position = "top") +
  guides(color = guide_legend(nrow = 3))
base +
  theme(legend.position = "bottom") +
  guides(color = guide_legend(nrow = 3))

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  theme(legend.position = "bottom") +
  guides(color = guide_legend(nrow = 2, override.aes = list(size = 4)))
#> `geom_smooth()` using method = 'loess' and formula = 'y ~ x'

#11.4.4 Replacing a scale

# Left
ggplot(diamonds, aes(x = carat, y = price)) +
  geom_bin2d()
```

```
# Right
ggplot(diamonds, aes(x = log10(carat), y = log10(price))) +
  geom_bin2d()

ggplot(diamonds, aes(x = carat, y = price)) +
  geom_bin2d() +
  scale_x_log10() +
  scale_y_log10()

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = drv))

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = drv)) +
  scale_color_brewer(palette = "Set1")

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = drv, shape = drv)) +
  scale_color_brewer(palette = "Set1")

presidential |>
  mutate(id = 33 + row_number()) |>
  ggplot(aes(x = start, y = id, color = party)) +
  geom_point() +
  geom_segment(aes(xend = end, yend = id)) +
  scale_color_manual(values = c(Republican = "#E81B23", Democratic = "#00AEF3"))

df <- tibble(
  x = rnorm(10000),
  y = rnorm(10000)
)

ggplot(df, aes(x, y)) +
  geom_hex() +
  coord_fixed() +
  labs(title = "Default, continuous", x = NULL, y = NULL)

ggplot(df, aes(x, y)) +
  geom_hex() +
  coord_fixed() +
```

```
  scale_fill_viridis_c() +
  labs(title = "Viridis, continuous", x = NULL, y = NULL)

ggplot(df, aes(x, y)) +
  geom_hex() +
  coord_fixed() +
  scale_fill_viridis_b() +
  labs(title = "Viridis, binned", x = NULL, y = NULL)

#11.4.5 Zooming

# Left
ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = drv)) +
  geom_smooth()

# Right
mpg |>
  filter(displ >= 5 & displ <= 6 & hwy >= 10 & hwy <= 25) |>
  ggplot(aes(x = displ, y = hwy)) +
  geom_point(aes(color = drv)) +
  geom_smooth()

# Left
ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = drv)) +
  geom_smooth() +
  scale_x_continuous(limits = c(5, 6)) +
  scale_y_continuous(limits = c(10, 25))

# Right
ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = drv)) +
  geom_smooth() +
  coord_cartesian(xlim = c(5, 6), ylim = c(10, 25))

suv <- mpg |> filter(class == "suv")
compact <- mpg |> filter(class == "compact")

# Left
```

```
ggplot(suv, aes(x = displ, y = hwy, color = drv)) +
  geom_point()

# Right
ggplot(compact, aes(x = displ, y = hwy, color = drv)) +
  geom_point()

x_scale <- scale_x_continuous(limits = range(mpg$displ))
y_scale <- scale_y_continuous(limits = range(mpg$hwy))
col_scale <- scale_color_discrete(limits = unique(mpg$drv))

# Left
ggplot(suv, aes(x = displ, y = hwy, color = drv)) +
  geom_point() +
  x_scale +
  y_scale +
  col_scale

# Right
ggplot(compact, aes(x = displ, y = hwy, color = drv)) +
  geom_point() +
  x_scale +
  y_scale +
  col_scale

#11.5 Themes

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point(aes(color = class)) +
  geom_smooth(se = FALSE) +
  theme_bw()

ggplot(mpg, aes(x = displ, y = hwy, color = drv)) +
  geom_point() +
  labs(
    title = "Larger engine sizes tend to have lower fuel economy",
    caption = "Source: https://fueleconomy.gov."
  ) +
  theme(
    legend.position = c(0.6, 0.7),
```

```
    legend.direction = "horizontal",
    legend.box.background = element_rect(color = "black"),
    plot.title = element_text(face = "bold"),
    plot.title.position = "plot",
    plot.caption.position = "plot",
    plot.caption = element_text(hjust = 0)
  )
#> Warning: A numeric `legend.position` argument in `theme()` was deprecated in ggplot2
#> 3.5.0.
#> ℹ Please use the `legend.position.inside` argument of `theme()` instead.

#11.6 Layout

p1 <- ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  labs(title = "Plot 1")
p2 <- ggplot(mpg, aes(x = drv, y = hwy)) +
  geom_boxplot() +
  labs(title = "Plot 2")
p1 + p2

p3 <- ggplot(mpg, aes(x = cty, y = hwy)) +
  geom_point() +
  labs(title = "Plot 3")
(p1 | p3) / p2



p1 <- ggplot(mpg, aes(x = drv, y = cty, color = drv)) +
  geom_boxplot(show.legend = FALSE) +
  labs(title = "Plot 1")

p2 <- ggplot(mpg, aes(x = drv, y = hwy, color = drv)) +
  geom_boxplot(show.legend = FALSE) +
  labs(title = "Plot 2")

p3 <- ggplot(mpg, aes(x = cty, color = drv, fill = drv)) +
  geom_density(alpha = 0.5) +
  labs(title = "Plot 3")

p4 <- ggplot(mpg, aes(x = hwy, color = drv, fill = drv)) +
```

```
  geom_density(alpha = 0.5) +
  labs(title = "Plot 4")

p5 <- ggplot(mpg, aes(x = cty, y = hwy, color = drv)) +
  geom_point(show.legend = FALSE) +
  facet_wrap(~drv) +
  labs(title = "Plot 5")

(guide_area() / (p1 + p2) / (p3 + p4) / p5) +
  plot_annotation(
    title = "City and highway mileage for cars with different drive trains",
    caption = "Source: https://fueleconomy.gov."
  ) +
  plot_layout(
    guides = "collect",
    heights = c(1, 3, 2, 4)
  ) &
  theme(legend.position = "top")

library(ggplot2)

ggplot(mpg, aes(x = displ, y = hwy, color = class)) +
  geom_point() +
  geom_smooth(se = FALSE) +
  labs(
    title = "Fuel Efficiency vs. Engine Size",
    subtitle = "Sports cars tend to have higher efficiency despite larger engines",
    x = "Engine Displacement (L)",
    y = "Highway Fuel Economy (mpg)",
    color = "Car Type",
    caption = "Data source: fueleconomy.gov"
  )



ggplot(mpg, aes(x = cty, y = hwy, color = drv, shape = drv)) +
  geom_point() +
  labs(
    title = "City vs. Highway MPG by Drive Type",
    subtitle = "Different shapes and colors represent drive types",
```

```
    x = "City MPG",
    y = "Highway MPG",
    color = "Type of Drive Train",
    shape = "Type of Drive Train",
    caption = "Data source: fueleconomy.gov"
  )

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  labs(
    title = "Main Title Explaining the Plot's Insight",
    subtitle = "Additional details to give context",
    x = "X-axis Label",
    y = "Y-axis Label",
    caption = "Data source: Your source"
  )

ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  annotate("text", x = 2, y = 45, label = "Top Left", hjust = 0) +
  annotate("text", x = 6, y = 45, label = "Top Right", hjust = 1) +
  annotate("text", x = 2, y = 15, label = "Bottom Left", hjust = 0) +
  annotate("text", x = 6, y = 15, label = "Bottom Right", hjust = 1) +
  labs(
    title = "Fuel Efficiency vs. Engine Size",
    subtitle = "Sports cars tend to have higher efficiency despite larger engines",
    x = "Engine Displacement (L)",
    y = "Highway Fuel Economy (mpg)",
    caption = "Data source: fueleconomy.gov"
  )

ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point(aes(color = cut), alpha = 1/20) +
  guides(color = guide_legend(override.aes = list(alpha = 1, size = 3))) +
  theme_minimal()

ggplot(diamonds, aes(x = carat, y = price)) +
  geom_point(aes(color = cut), alpha = 1/20) +
  guides(color = guide_legend(override.aes = list(alpha = 1, size = 3))) +
  theme(
```

```
  axis.title.x = element_text(color = "blue", face = "bold"),
  axis.title.y = element_text(color = "blue", face = "bold")
)
```