Statistical programming

Creating graphics with `ggplot2`

Marc Comas-Cufí

Today's session

- `ggplot2`: a system for declaratively creating graphics, based on "The Grammar of Graphics"
- `ggplot2` extensions.

R plotting systems

- **graphics**. Defaults R plotting system. Fast for exploratory analysis. Nice graphics are constructed step by step using different calls.
- **grid** package based.
  - **lattice**. Fast plots for exploratory analysis. By default plots are nicer than base system. Tuning is difficult.
  - **ggplot2**. System implementing a layered grammar of graphics.

Visualising data with `ggplot2`

Fuel economy data from 1999 to 2008 for 38 popular models of cars:

```
1 library(tidyverse)
2 data(mpg, package='ggplot2')
3 mpg
```

```
# A tibble: 234 × 11
  manufacturer model displ  year   cyl trans   drv     cty   hwy fl
  <chr>        <chr> <dbl> <int> <int> <chr>   <chr> <int> <int> <chr>
1 audi         a4      1.8  1999     4 auto(l… f        18    29 p
2 audi         a4      1.8  1999     4 manual… f        21    29 p
3 audi         a4      2    2008     4 manual… f        20    31 p
4 audi         a4      2    2008     4 auto(a… f        21    30 p
5 audi         a4      2.8  1999     6 auto(l… f        16    26 p
# … with 229 more rows, and 1 more variable: class <chr>
```

Creating graphics with ggplot2

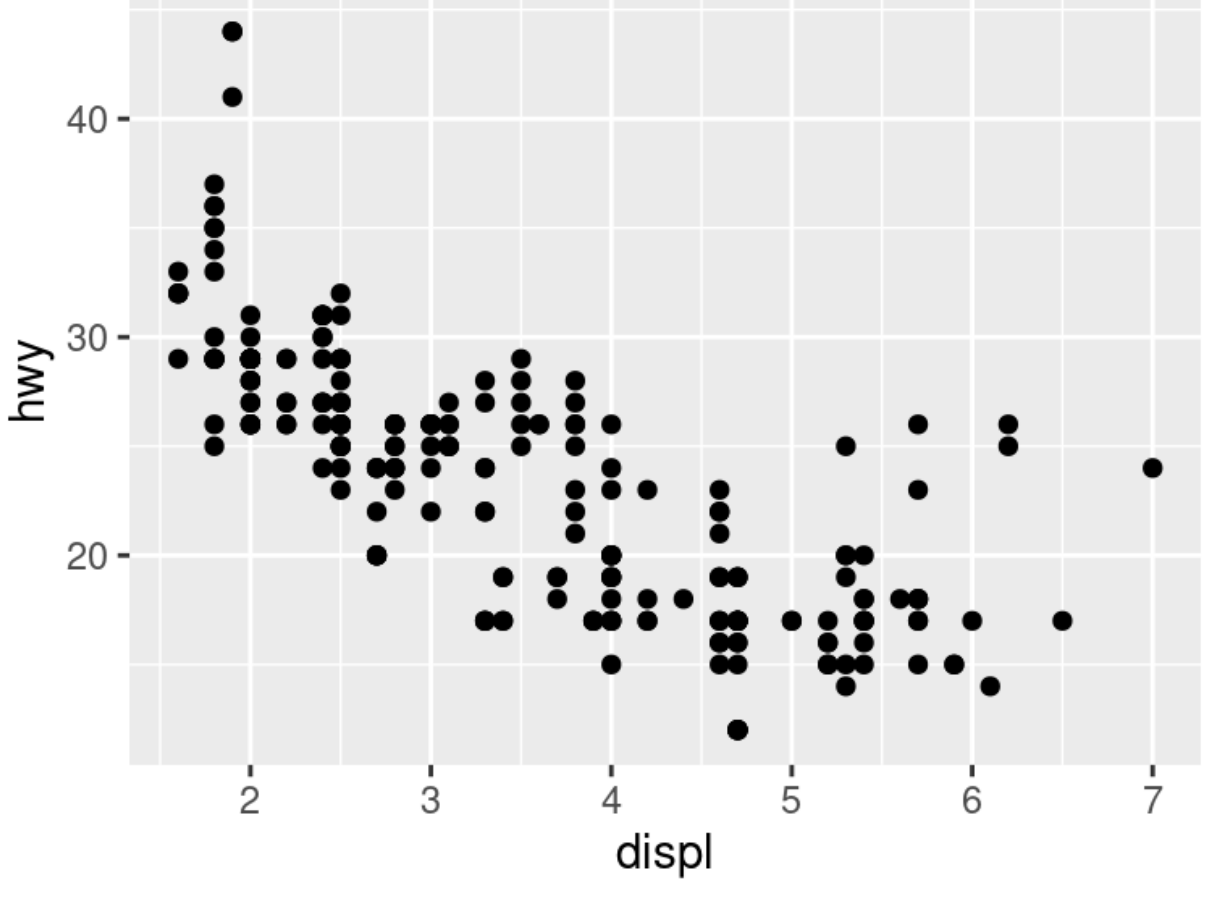
```
ggplot(data = <DATA>) +                          # INITIAL LAYER
 <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>)) +  # NEXT LAYER
 ⋮
 <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>)) +  # LAST LAYER
 <TUNNING>
```

- DATA: available variables
- GEOM_FUNCTION: what should be plotted
- MAPPINGS: relations between variables and aesthetics

Creating graphics with `ggplot2`

- Mapping: `displ` → `x`, `hwy` → `y`

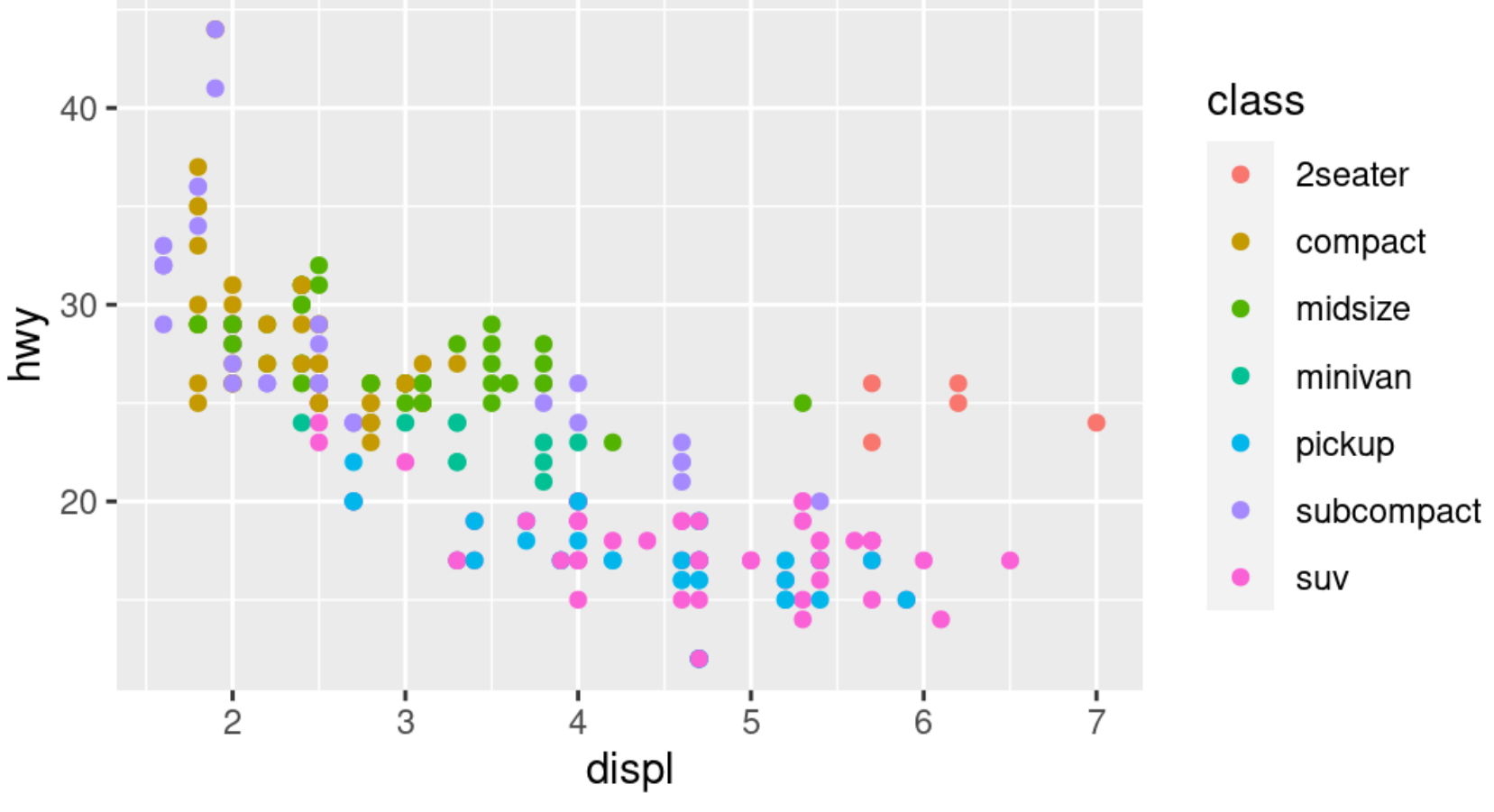
```
1 ggplot(data = mpg) +
2   geom_point(mapping = aes(x = displ, y = hwy))
```



Creating graphics with `ggplot2`

- Mapping: `displ` → `x`, `hwy` → `y`, `class` → `color`

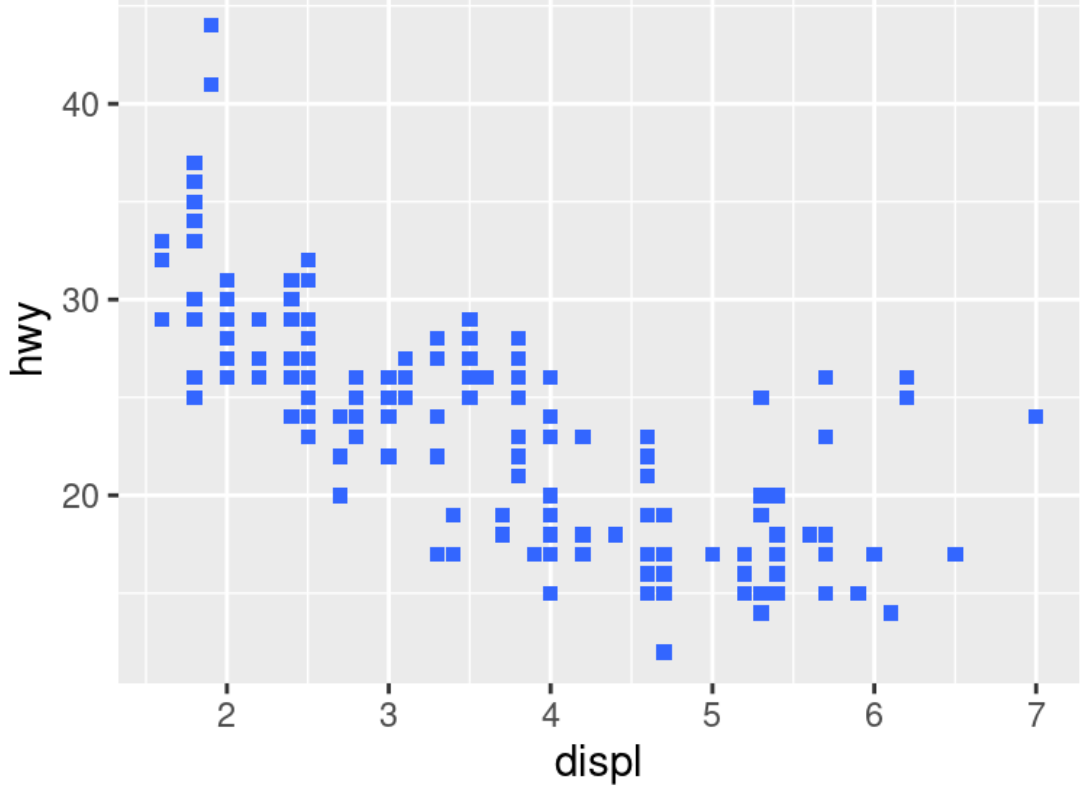
```
1 ggplot(data = mpg) +
2   geom_point(mapping = aes(x = displ, y = hwy, color = class))
```



Creating graphics with `ggplot2`

- Mapping: `displ` → `x`, `hwy` → `y`
- We can fix the value of aesthetics

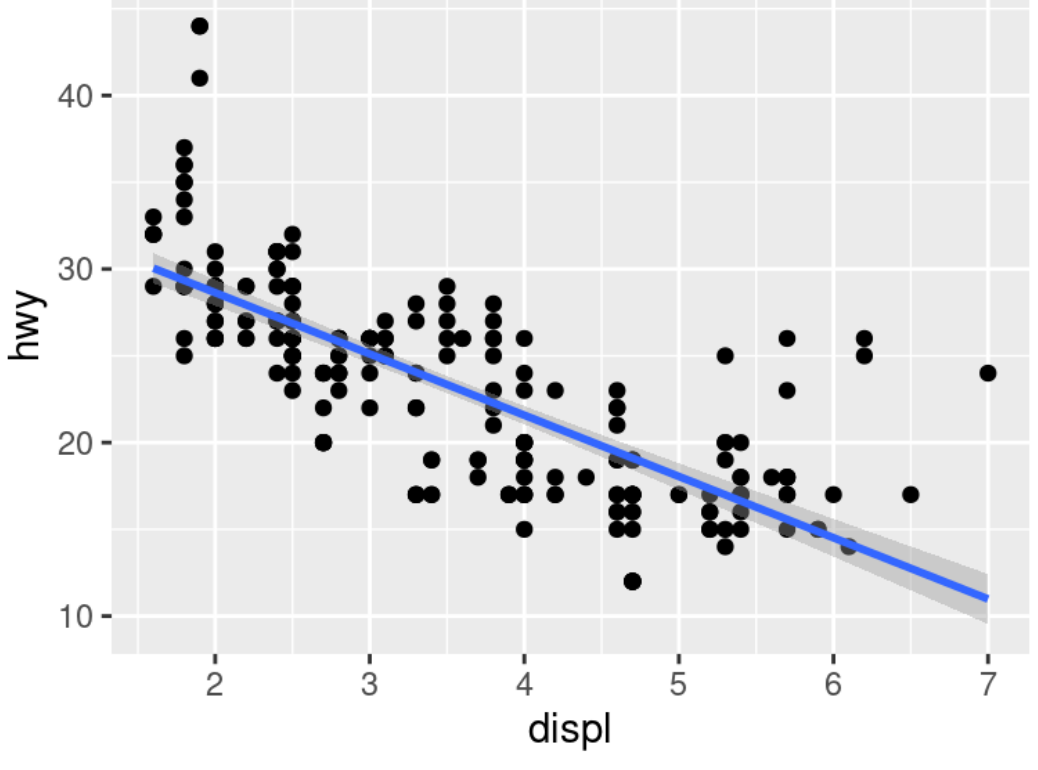
```
1 ggplot(data = mpg) +
2   geom_point(mapping = aes(x = displ, y = hwy), color = "#3366ff", shape = 15)
```



Creating graphics with `ggplot2`

- Mapping: `displ` → `x`, `hwy` → `y`
- We can add more layers

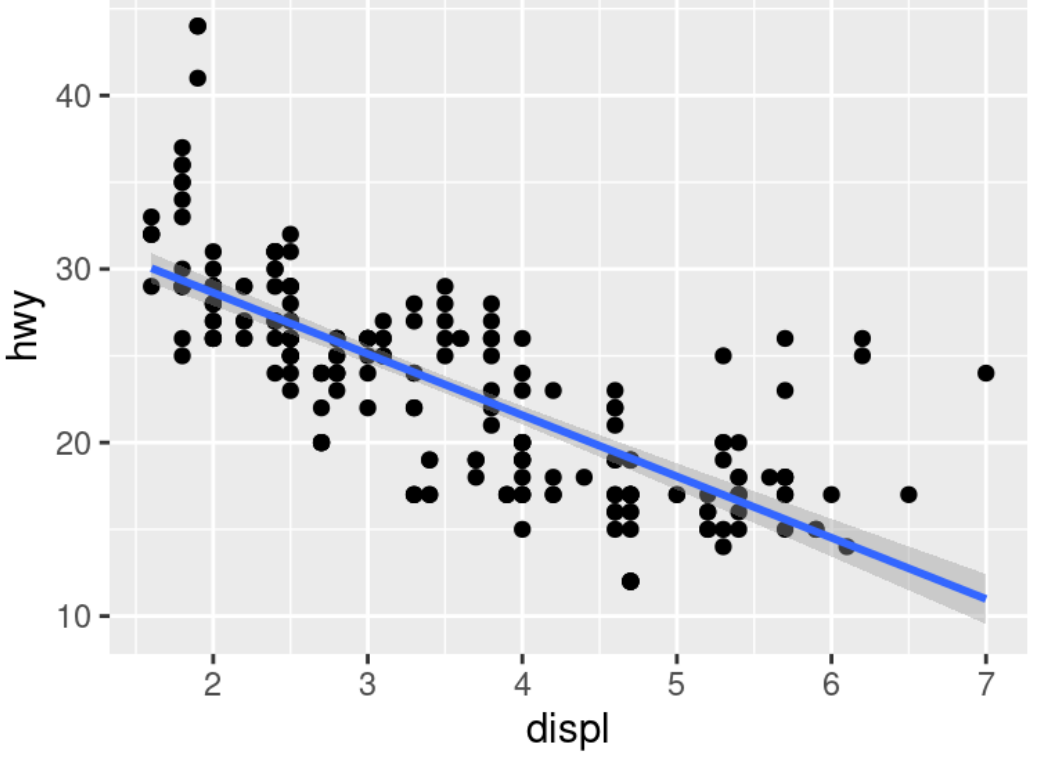
```
1 ggplot(data = mpg) +
2   geom_point(mapping = aes(x = displ, y = hwy)) +
3   geom_smooth(mapping = aes(x = displ, y = hwy), method = 'lm')
```



Creating graphics with `ggplot2`

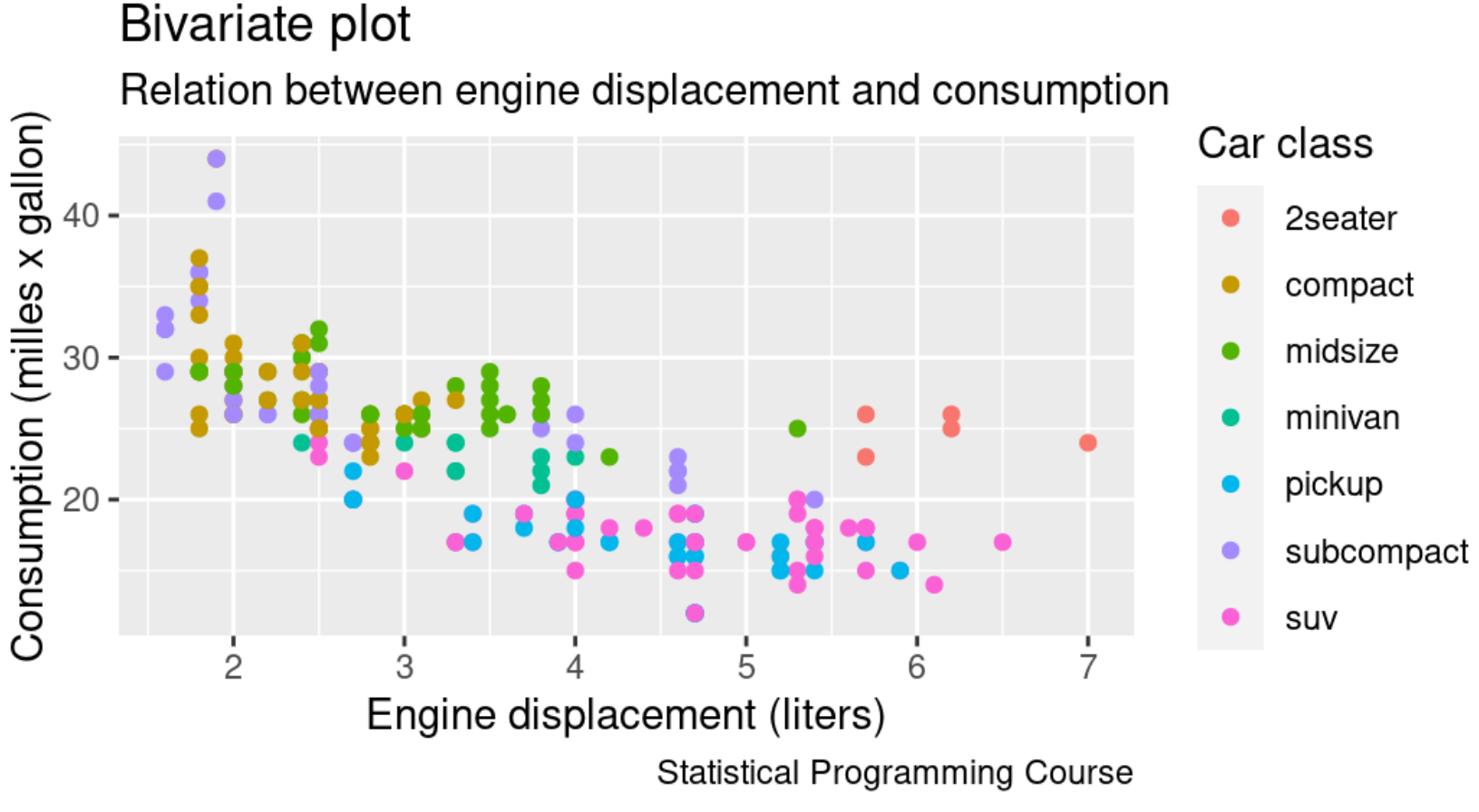
- Mapping: `displ` → `x`, `hwy` → `y`
- We can add more layers

```
1 ggplot(data = mpg, aes(x = displ, y = hwy)) +
2   geom_point() +
3   geom_smooth(method = 'lm')
```



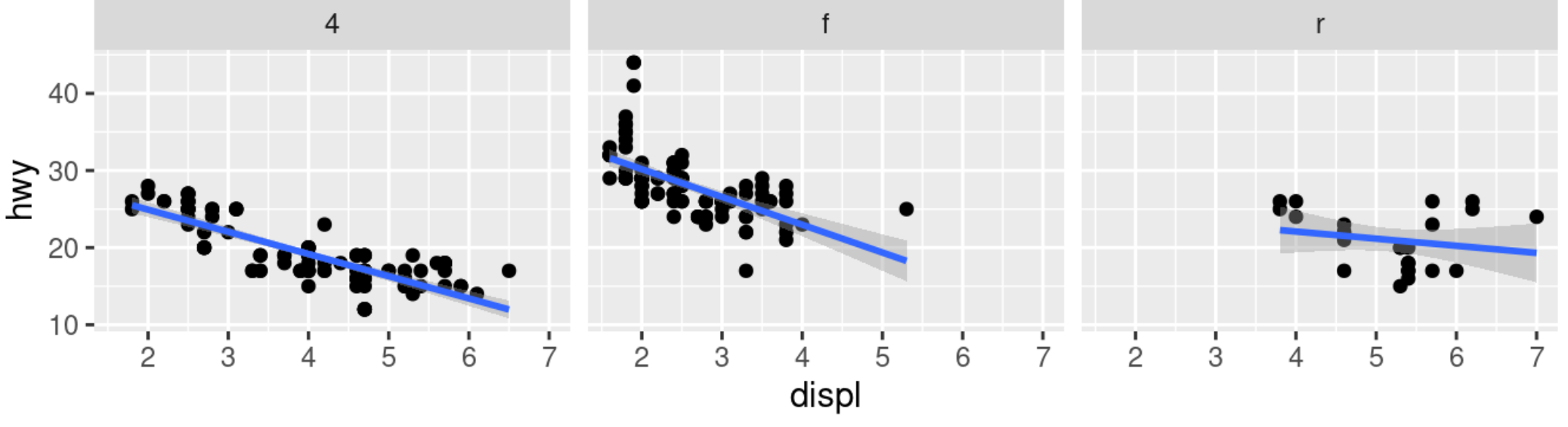
Adding labels

```
ggplot(data = mpg, aes(x = displ, y = hwy, color = class)) +
  geom_point() +
  labs(title = "Bivariate plot",
       subtitle = "Relation between engine displacement and consumption",
       x = 'Engine displacement (liters)', y = 'Consumption (milles x gallon)',
       color = 'Car class', caption = "Statistical Programming Course")
```



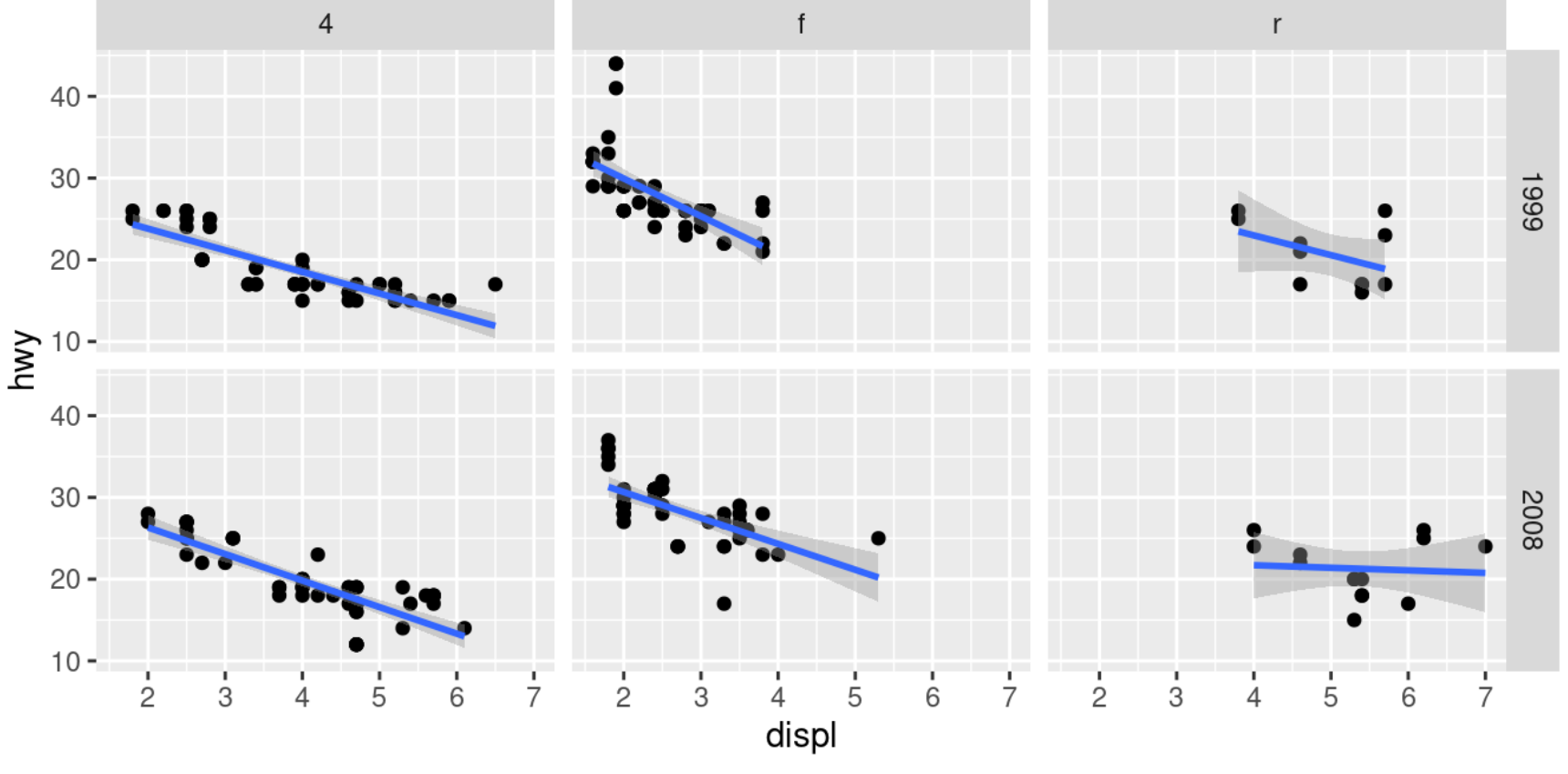
Faceting (`facet_wrap()` and `facet_grid()`)

```
1 ggplot(mpg, aes(x = displ, y = hwy)) +
2   geom_point() + geom_smooth(method = 'lm') +
3   facet_wrap(~drv)
```



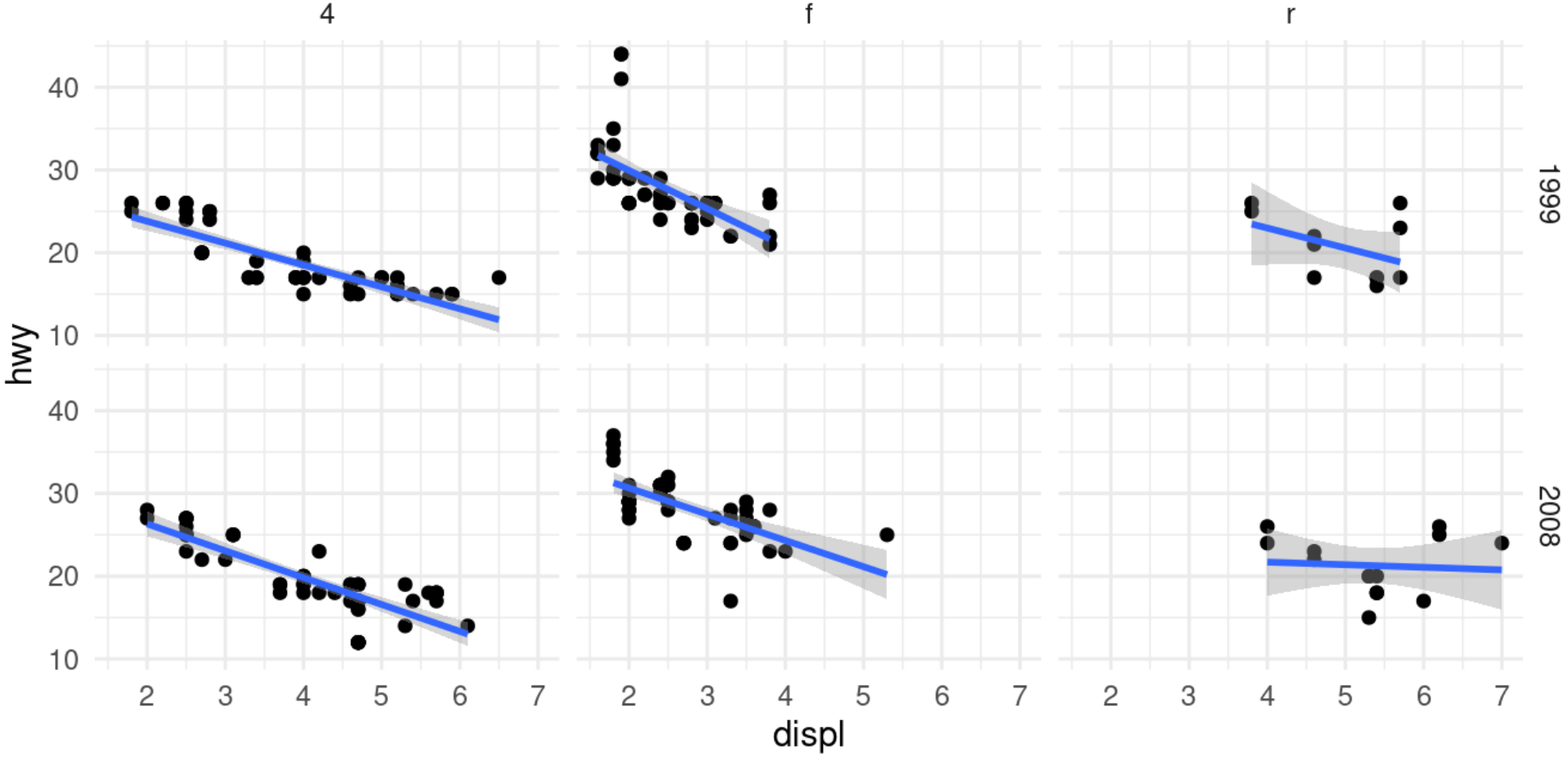
Faceting (`facet_wrap()` and `facet_grid()`)

```
1 ggplot(mpg, aes(x = displ, y = hwy)) +
2   geom_point() + geom_smooth(method = 'lm') +
3   facet_grid(year~drv)
```



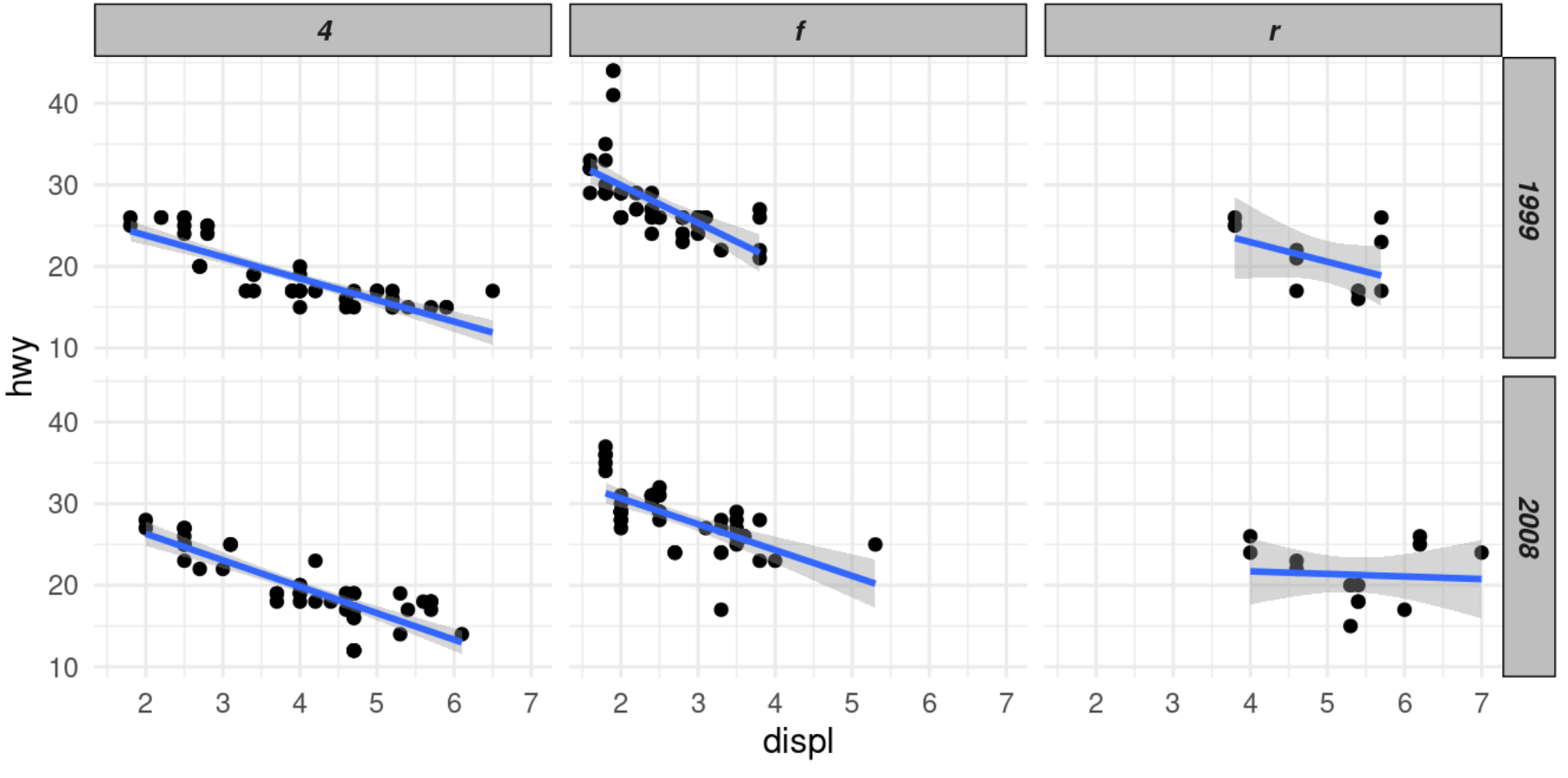
Themes

```
1 ggplot(mpg, aes(x = displ, y = hwy)) +
2   geom_point() + geom_smooth(method = 'lm') +
3   facet_grid(year~drv) +
4   theme_minimal()
```



Themes

```
ggplot(mpg, aes(x = displ, y = hwy)) +
  geom_point() + geom_smooth(method = 'lm') +
  facet_grid(year~drv) +
  theme_minimal() +
  theme(strip.text = element_text(face = "bold.italic"),
        strip.background = element_rect(fill = 'grey'))
```



Saving a `ggplot2` figure

```
# Check size with par('din')
ggsave(plot = p, filename =  "filename.pdf", width = 6, height = 4)
ggsave(plot = p, filename =  "filename.svg", width = 6, height = 4)
```

If `plot` parameter is omitted, last plot will be saved.

Learning more about `ggplot2`

- https://ggplot2.tidyverse.org
- Cheat sheet html
- R Graphics Cookbook https://r-graphics.org/
- The R Graph Gallery https://www.r-graph-gallery.com/ggplot2-package.html
- `gggplot2`: Elegant Graphics for Data Analysis https://ggplot2-book.org/

Descriptive statistics: Univariate analysis

Summarising a categorical variable

- How is the distribution of `origin`?

```
p1 = ggplot(data=flights) +
  geom_bar(aes(x = origin))

p2 = ggplot(data=flights) +
  geom_bar(aes(x = origin,
               y=(..count..)/sum(..count..)))

library(patchwork)
p1 + p2
```

Summarising a categorical variable

```r
dtab = flights %>% count(origin) %>%
  mutate(p = sprintf("%0.1f%%", 100*prop.table(n)),
         cn = rev(cumsum(rev(n))),
         y = cn + diff(c(cn,0))/2 )

ggplot(data=flights) +
  geom_bar(aes(x="",y=(..count..), fill=origin)) +
  geom_text(data=dtab, aes(x="", y=y, label = p)) +
  coord_polar(theta = 'y', start = pi/2, direction = 1) +
  theme_void()
```

Summarising a numerical variable

- How is the distribution of `dep_delay`?

```
ggplot() +
  geom_histogram(data=flights, aes(x=dep_delay), bins = 10)
```

Summarising a numerical variable

- How is the distribution of `dep_delay`?

```
ggplot() +
  geom_histogram(data=flights, aes(x=dep_delay), breaks = c(-50, 0, 50, 200, 1500))
```

Summarising a numerical variable

- How is the distribution of `dep_delay`?

```
ggplot() +
  geom_boxplot(data=flights, aes(x=dep_delay))
```

Summarising a numerical variable

- How is the distribution of `dep_delay`?

```
ggplot() +
  geom_density(data=flights, aes(x=dep_delay), col=NA, fill = 'blue', alpha=0.4)
```

Summarising a numerical variable

- How is the distribution of `dep_delay`?

```
ggplot() +
  geom_density(data=flights, aes(x=dep_delay), col=NA, fill = 'blue', alpha=0.4) +
  coord_cartesian(xlim = c(-30,60))
```

Summarising a numerical variable

- How is the distribution of `dep_delay`?

```
ggplot() +
  geom_histogram(data=flights, aes(x=dep_delay), breaks=seq(-50,1500,5),
                 fill='blue', alpha=0.4) +
  coord_cartesian(xlim = c(-30,60))
```

Relation between numerical and categorical variables

- How is the distribution of `dep_delay` and `origin`?

```
ggplot() +
  geom_boxplot(data=flights, aes(x=origin, y=dep_delay))
```

Relation between numerical and categorical variables

- How is the distribution of `dep_delay` and `origin`?

```
ggplot() +
  geom_density(data=flights, aes(x=dep_delay), col=NA, fill = 'blue', alpha=0.4) +
  coord_cartesian(xlim = c(-30,60)) +
  facet_wrap(~origin, ncol = 1)
```

Relation between numerical and categorical variables

- How is the distribution of `dep_delay` and `origin`?

```
ggplot() +
  geom_histogram(data=flights, aes(x=dep_delay), breaks=seq(-50,1500,5),
                 fill='blue', alpha=0.4) +
  coord_cartesian(xlim = c(-30,60)) +
  facet_wrap(~origin, ncol = 1)
```

Relation between numerical and categorical variables

- How is the distribution of `dep_delay` and `origin`?

```
library(ggridges)
ggplot() +
  geom_density_ridges(data=flights, aes(x=dep_delay, y = origin),
                      scale=2, col=NA, fill = 'blue', alpha=0.4) +
  coord_cartesian(xlim = c(-30,60))
```

Relation between categorical variables

```
flights = flights %>%
  filter(!is.na(arr_delay)) %>%
  mutate(arrival = if_else(arr_delay > 0, 'delayed', 'on-time'))
```

- How is the distribution of `origin` and `on.time`

```
ggplot(data=flights) +
  geom_bar(aes(x=arrival, fill=origin))
```

Relation between categorical variables

```
1 flights_n = flights %>% count(origin, arrival)
```

- `origin` relative frequencies

```
1 dplot = group_by(flights_n, arrival) %>% mutate(p = prop.table(n))
2 ggplot(data=dplot) +
3   geom_bar(aes(x=arrival, y=p, fill=origin), stat = 'identity')
```

Relation between categorical variables

```
1 flights_n = flights %>% count(origin, arrival)
```

- `arrival` relative frequencies

```
1 dplot = group_by(flights_n, origin) %>% mutate(p = prop.table(n))
2 ggplot(data=dplot) +
3   geom_bar(aes(x=origin, y=p, fill=arrival), stat = 'identity')
```

Relation between numerical variables

- How is the distribution of `dep_delay` and `arr_delay`?

```
ggplot(data=flights) +
  geom_point(aes(x=dep_delay, y=arr_delay))
```

Relation between numerical variables

```
ggplot(data=flights) +
  geom_point(aes(x=dep_delay, y=arr_delay, alpha = ..n..), size = 1, stat = 'sum')
```

```
# Equivalent,
# ggplot(data=count(flights, dep_delay, arr_delay)) +
#   geom_point(aes(x=dep_delay, y=arr_delay, alpha = n), size = 1, stat = 'identity')
```

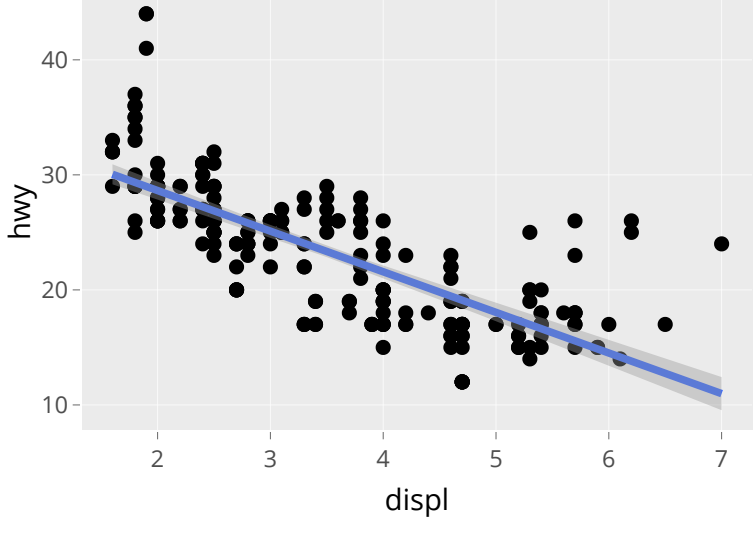
ggplot2 extensions

Packages with more themes

- `ggthemes`. https://github.com/jrnold/ggthemes
- `ggtech`. https://github.com/ricardo-bion/ggtech
- `ggthemr`: https://github.com/cttobin/ggthemr
- `cowplot`: https://cran.r-project.org/web/packages/cowplot/vignettes/introduction.html
- `bbplot`. https://bbc.github.io/rcookbook
- `hrbrthemes`: https://github.com/hrbrmstr/hrbrthemes

Interactive plots with `plotly`

```
1  library(plotly)
2
3  p = ggplot(data = mpg, aes(x = displ, y = hwy)) +
4    geom_point() +
5    geom_smooth(method = 'lm')
6
7  ggplotly(p)
```



Animations with gganimate

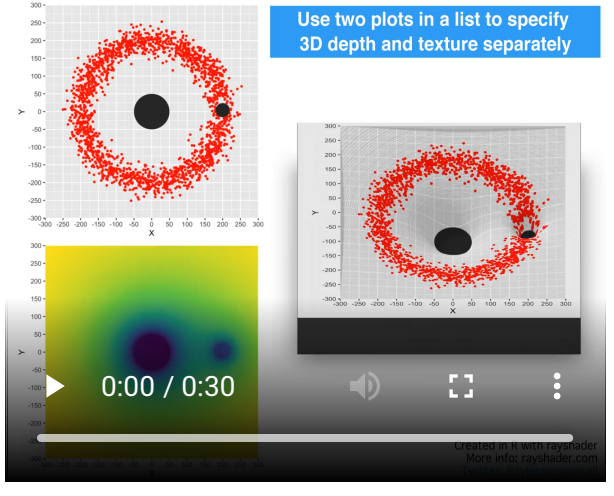
```
library(gganimate)
library(gapminder)

p <- ggplot(gapminder, aes(gdpPercap, lifeExp, size = pop, colour = country)) +
  geom_point(alpha = 0.7, show.legend = FALSE) +
  scale_colour_manual(values = country_colors) +
  scale_size(range = c(2, 12)) +
  scale_x_log10() +
  facet_wrap(~continent) +
  # Here comes the gganimate specific bits
  labs(title = 'Year: {frame_time}', x = 'GDP per capita', y = 'life expectancy') +
  transition_time(year) +
  ease_aes('linear')
```

Animations with `gganimate`

```
1 animate(p, nframes = 20, fps = 5, width = 500, height=400)
```

3D plots with rayshader



- https://www.tylermw.com/3d-ggplots-with-rayshader/

BUT REMEMBER:

- Don’t use gratuitous 3D
- It’s difficult to interpret static 3D visualizations

Other extensions

- `ggmap` for maps https://github.com/dkahle/ggmap
- `ggtern`for ternary diagrams http://www.ggtern.com/
- `ggdendro` for dendrograms https://github.com/andrie/ggdendro

That’s all for today

Next week session

- Overview of probability
  - Probability
  - Random variables
  - The Central limit theorem
  - Simulation