INTERMEDIATE DATA VIZ IN R

ggplot2 + forcats

Jake Riley

6/4/2020

Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

TODAY'S TALK

- Intro
- What is ggplot
- Tips & Tricks
- Best Practices

AN INTRO

- Jake Riley
- Clinical Data Analyst
- Avid `tidyverse` answerer on stackoverflow
- Dogdad

Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

BEFORE WE GET STARTED

- this talk is aimed at intermediate `ggplot2` users
- everything is within the `tidyverse` framework & R for Data Science (R4DS)

- the pipe `%>%` is used in many places and allows us to create a sequence of manipulations

```
iris %>% arrange(Species)

arrange(iris, Species)
```

- the `+` used with `ggplot()` is another type of pipe

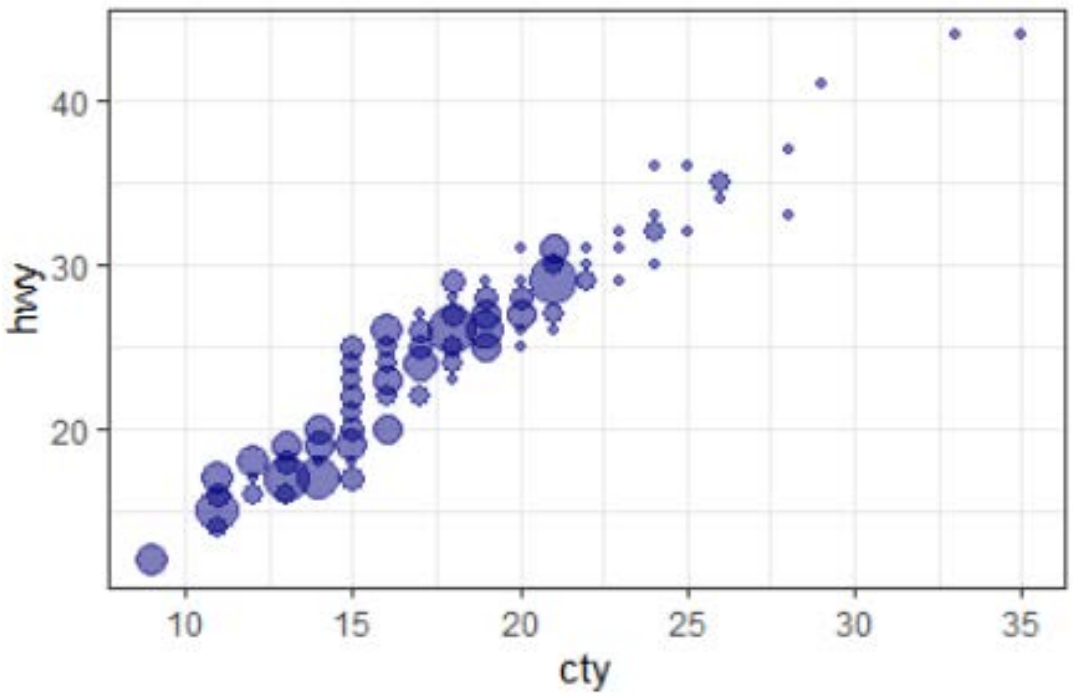
- you can pipe from a `dplyr` sequence into a `ggplot()` sequence

WHY ggplot?

- grammar of graphics
- just like every sentence has a **subject, verb, and noun**, every chart has a **coordinate system, geom, and aesthetics**

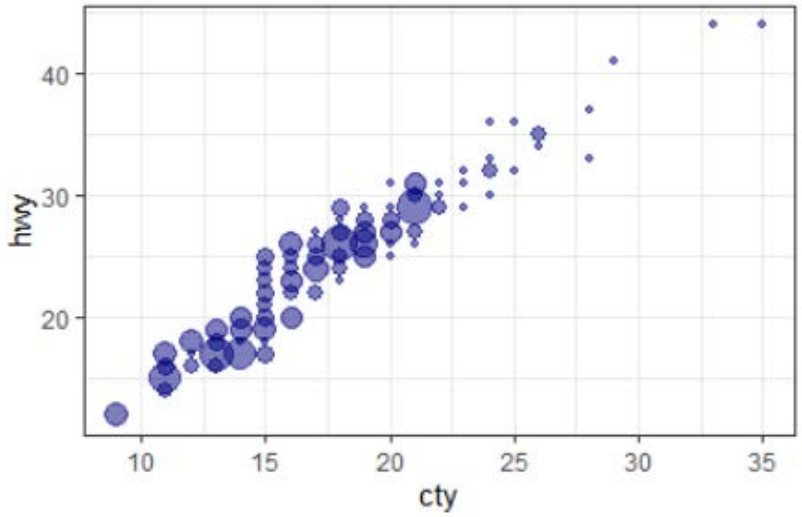
- the hope is that we will invent new types of charts

```
library(tidyverse)

p <-
  ggplot(mpg) +
  geom_count(
    aes(cty, hwy),
    alpha = 0.5, color = "navyblue"
  ) +
  theme_bw() +
  theme(legend.position = "none")
```

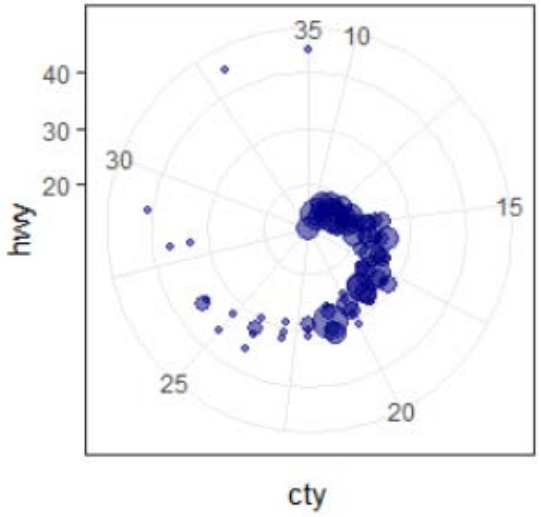


AN EXAMPLE

```
p
```




```
p + coord_polar()
```

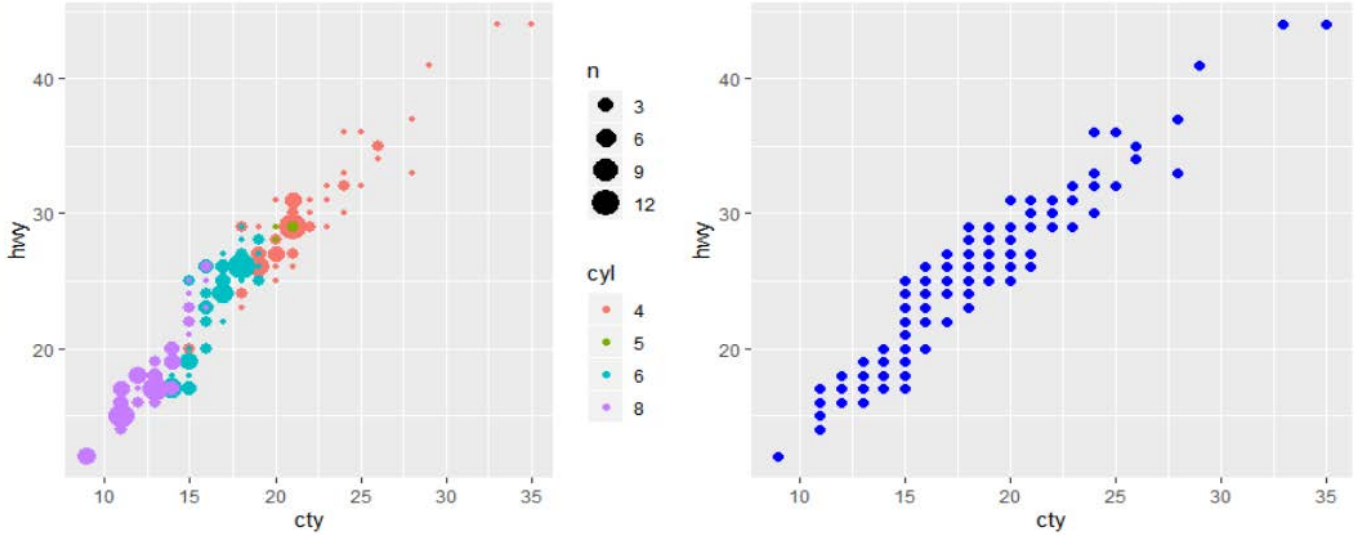


Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

DEMYSTIFYING AES( )

- **aes( )** = aesthetics
- dynamic, data driven **variables** go inside the **aes( )**
- constant, static **values** go outside
- the first 2 arguments of **aes( )** are **x** and **y** and usually omitted

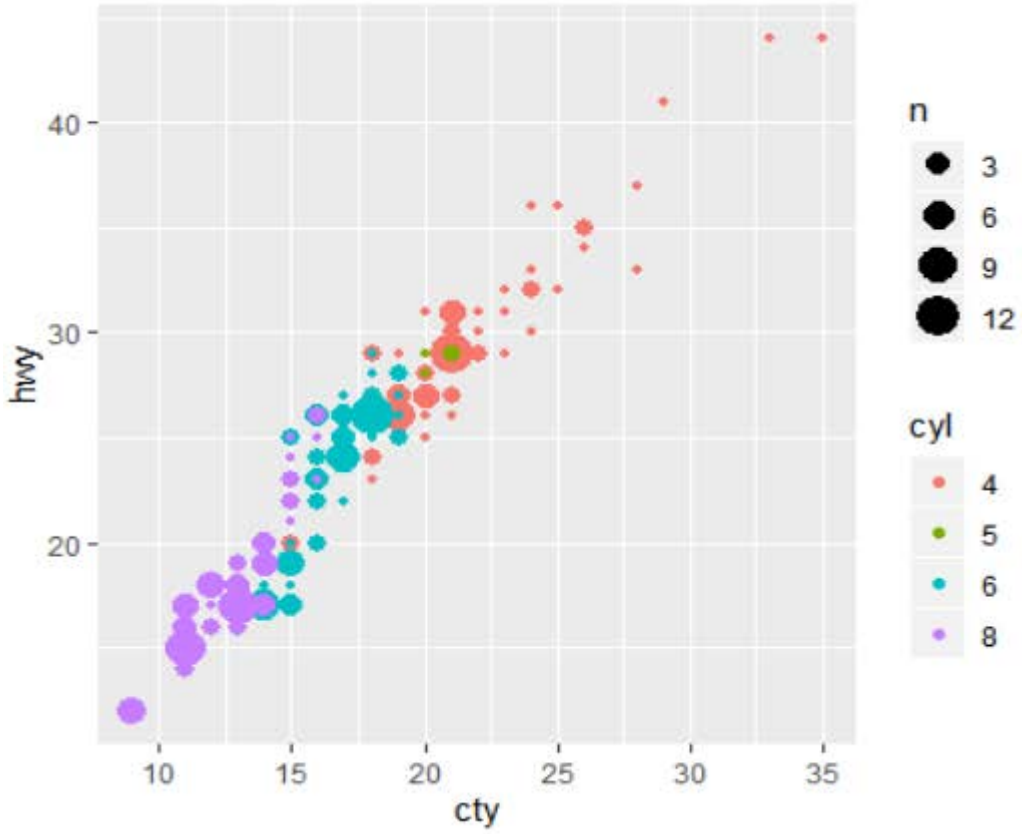
```
geom_point(aes(color = class, size = n), ...)
geom_point(aes(...), color = "blue", size = 2)
```

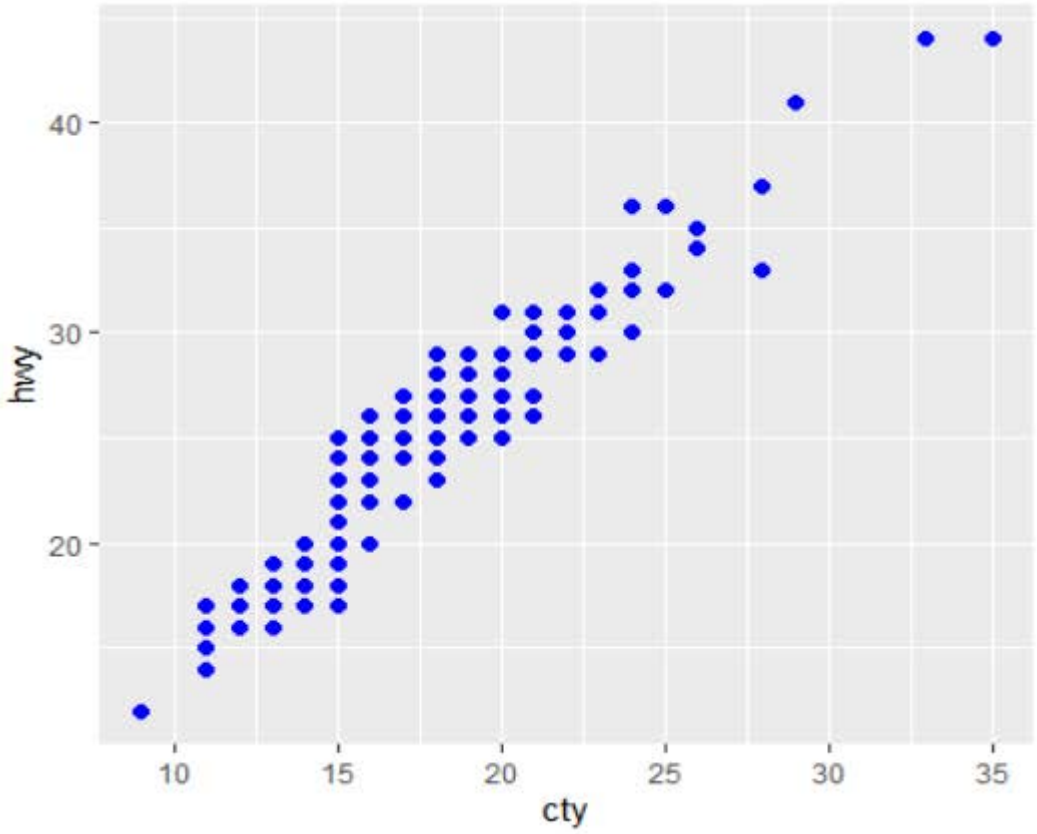


Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

NOTE THE DIFFERENCE

- `geom_point(aes(color = class, size = n , ...)`
- `geom_point(aes(...), color = "blue", size = 2)`



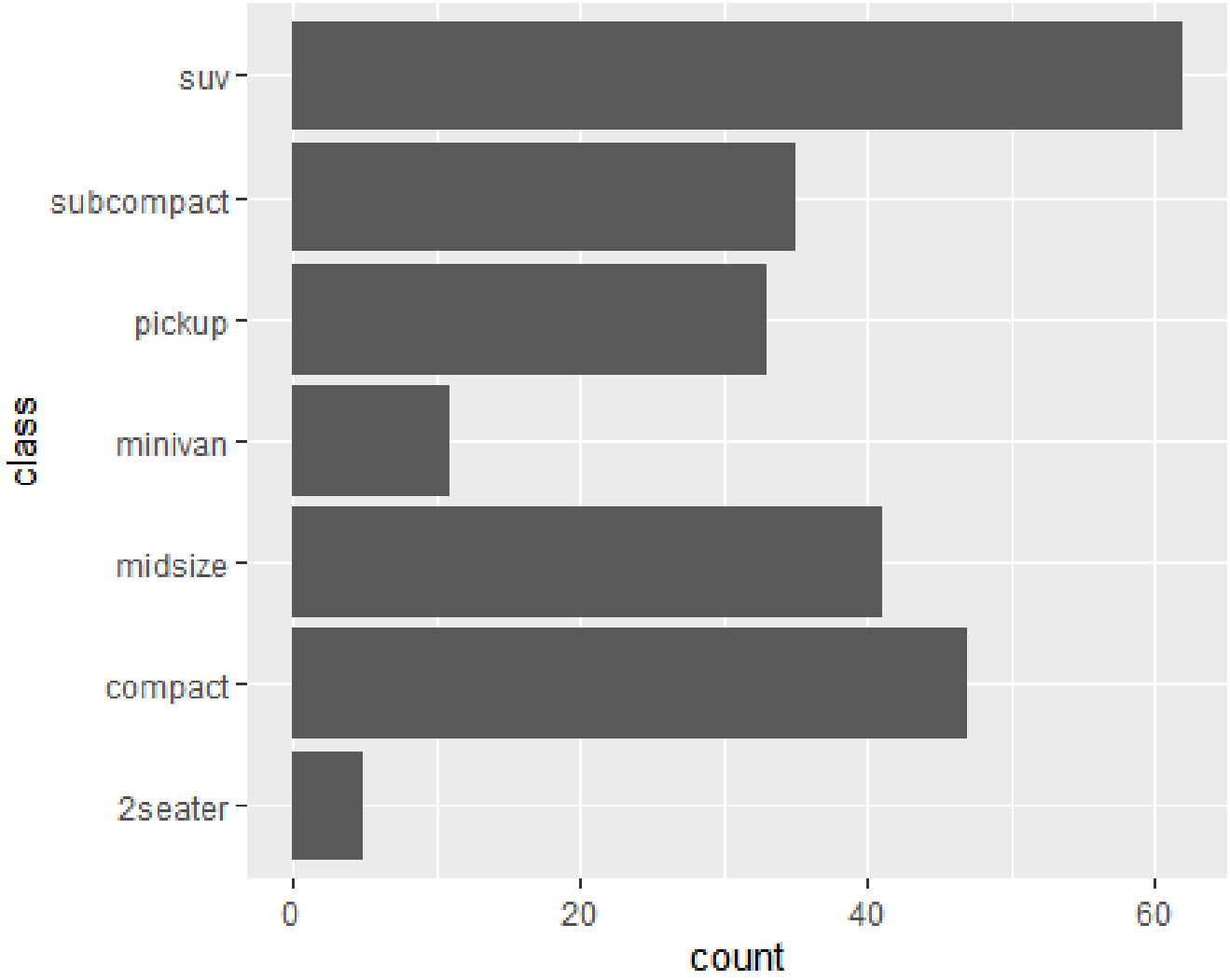


Tips & Tricks

DESCENDING BAR CHARTS

The number one things I get asked is how to make a bar chart in descending order.

```
ggplot(mpg, aes(y = class)) +
  geom_bar()
```



AGGREGATED DATA: `fct_reorder()`

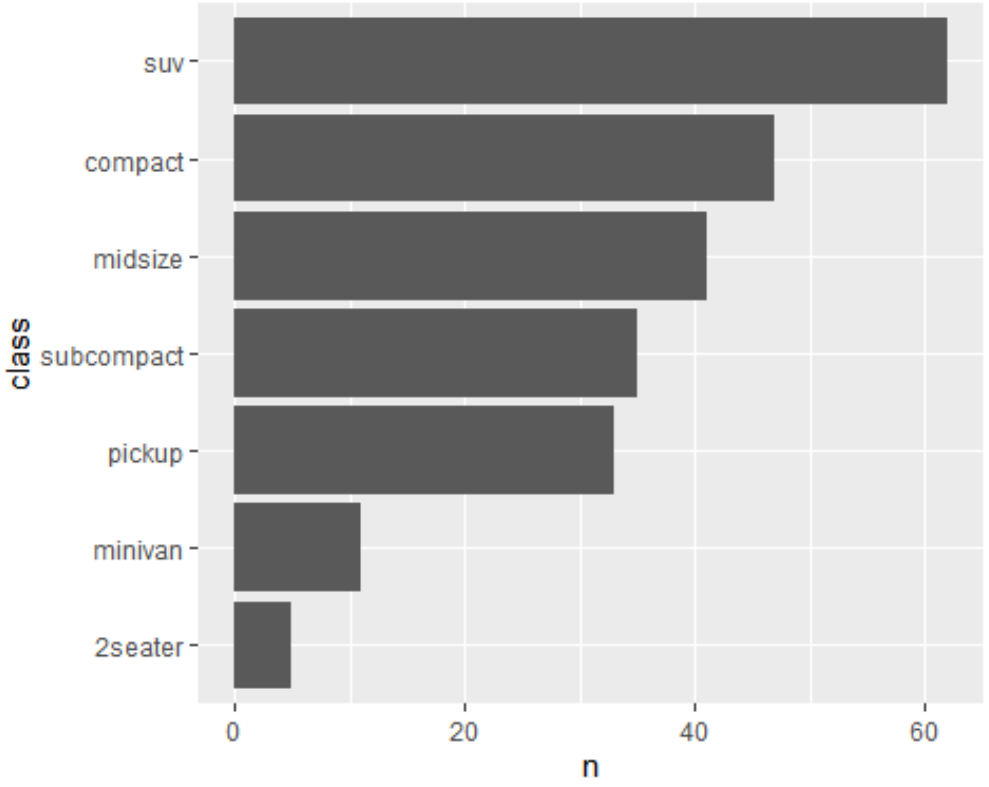
```r
library(tidyverse)

mpg %>%
  count(class) %>%
  mutate(
    class = fct_reorder(class, .x = n),
    class_int = as.integer(class)
  ) %>%
  arrange(class)
```

```
#  class          n  class_int
#  2seater        5          1
#  minivan       11          2
#  pickup        33          3
#  subcompact    35          4
#  midsize       41          5
#  compact       47          6
#  suv           62          7
```

Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

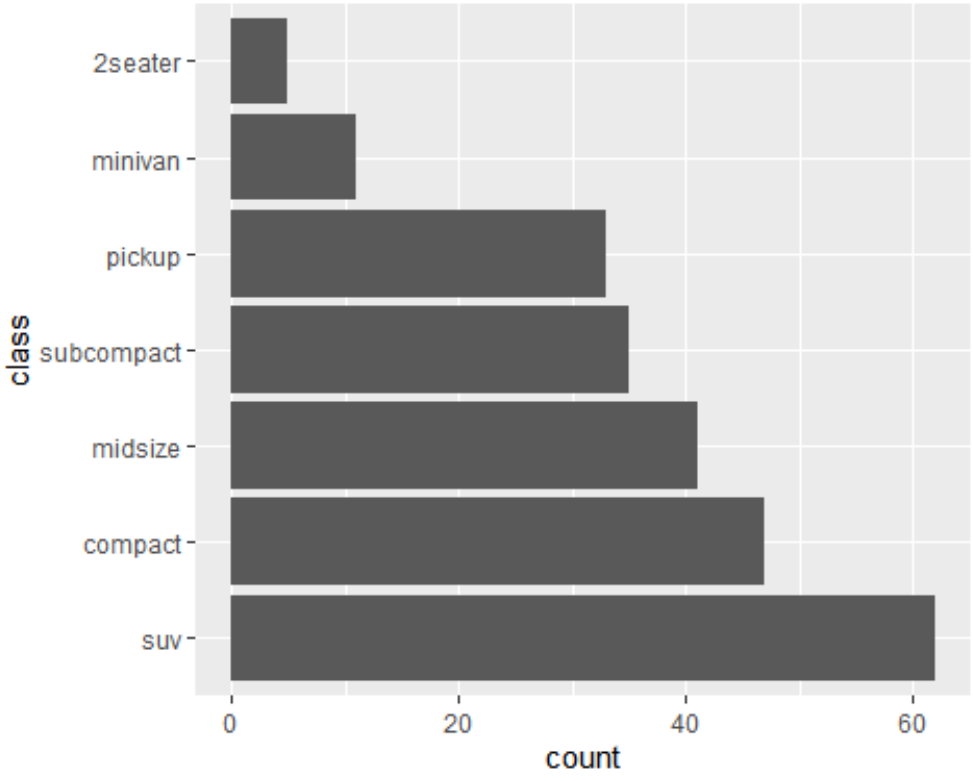
AGGREGATED DATA: geom_col()

```
mpg %>%
  count(class) %>%
  mutate(class = fct_reorder(class, .x = n)) %>%
  ggplot(aes(x = n, y = class)) +
  geom_col()
```



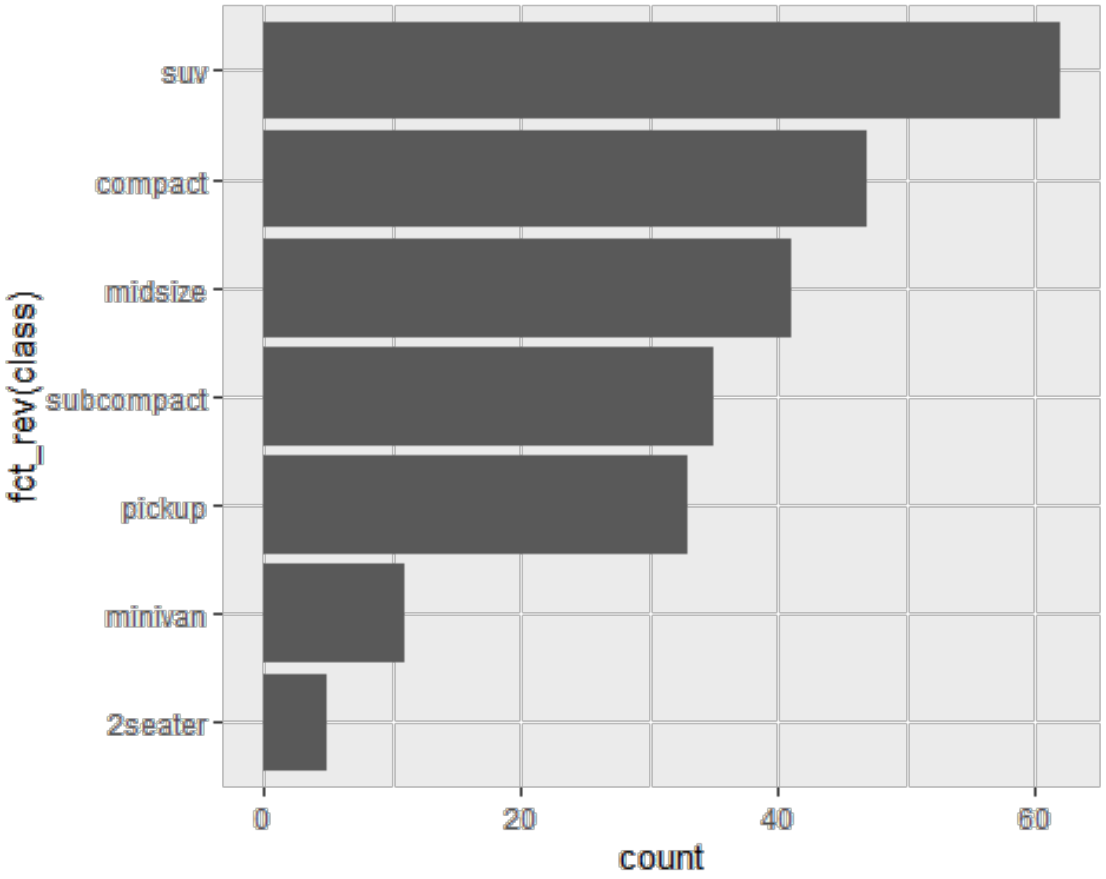
ARRANGE BY VOLUME: `fct_infreq()`

```
mpg %>%
  mutate(class = fct_infreq(class)) %>%
  ggplot(aes(y = class)) +
  geom_bar()
```



ARRANGE IN DESCENDING ORDER: `fct_rev()`

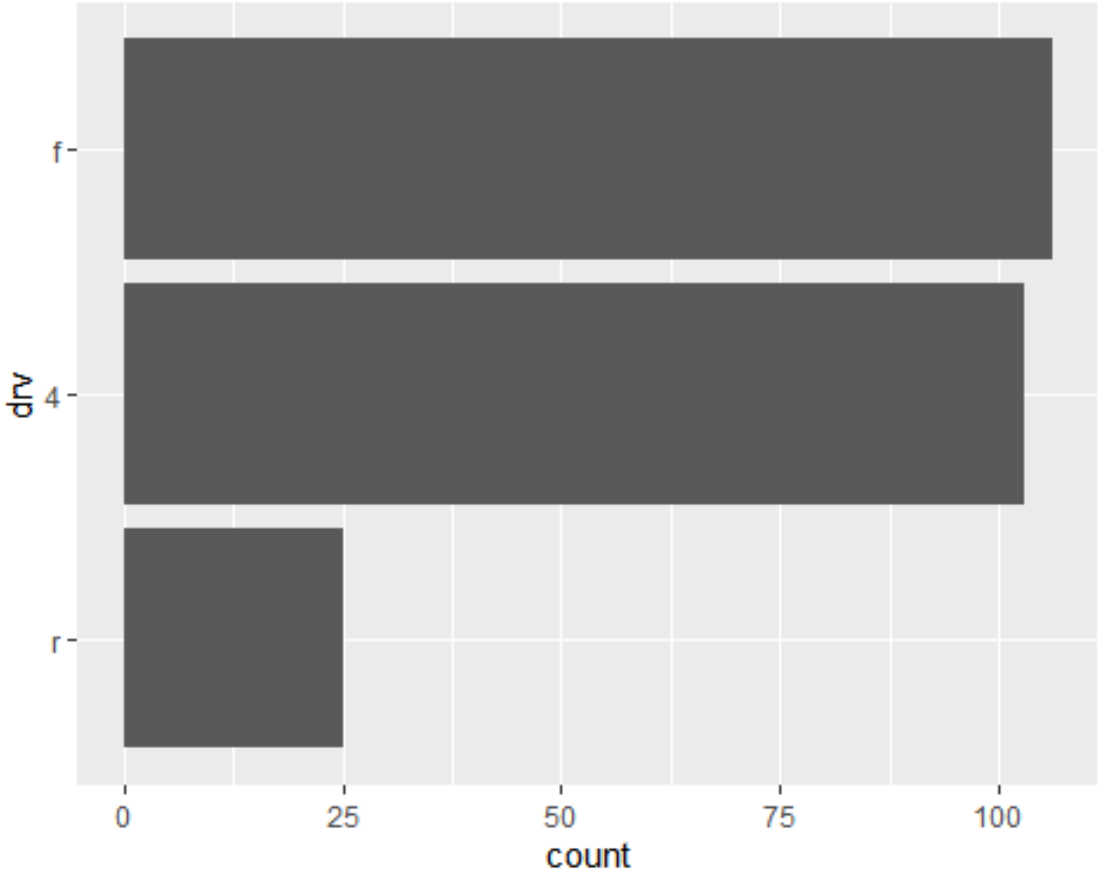
```
mpg %>%
  mutate(class = fct_infreq(class)) %>%
  ggplot(aes(y = fct_rev(class))) +
  geom_bar()
```



Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

YOU CAN ALSO PIPE THE `fct_*()` STEPS

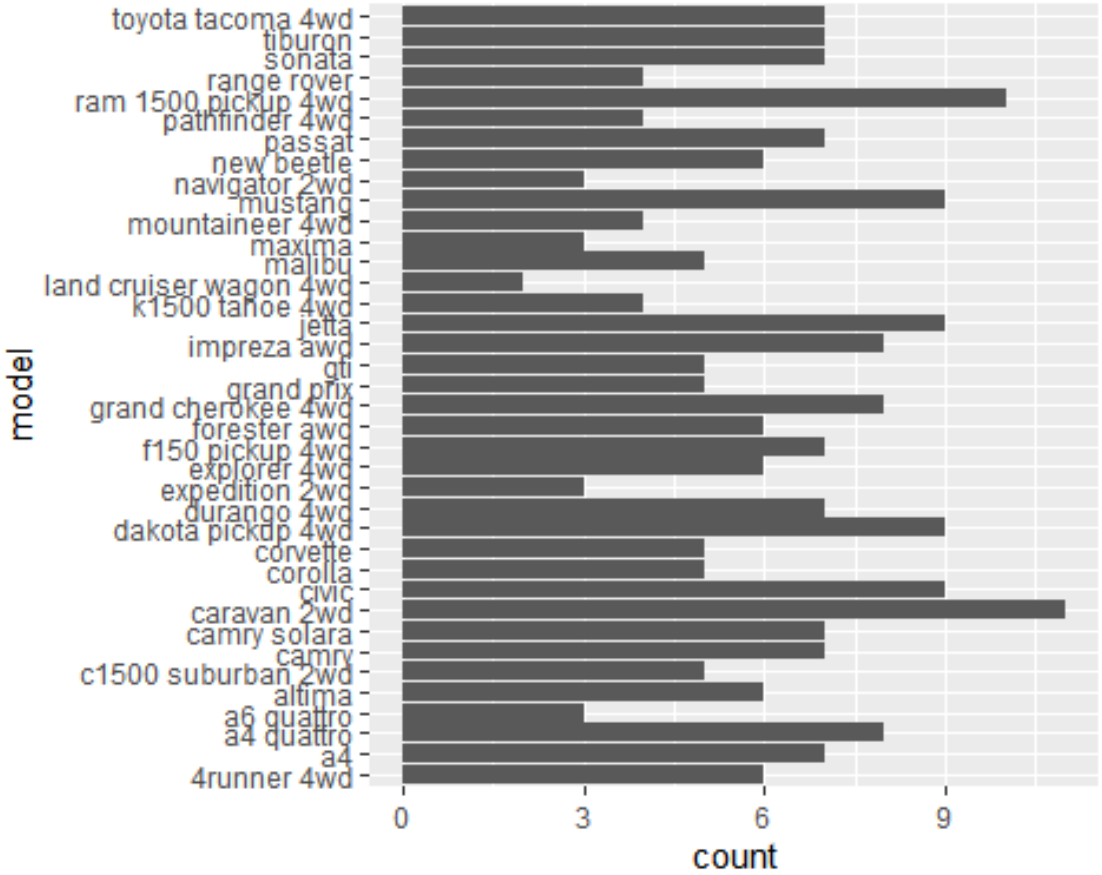
```
mpg %>%
  mutate(
    drv =
      fct_infreq(drv) %>%
      fct_rev()
  ) %>%
  ggplot(aes(y = drv)) +
  geom_bar()
```



Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

TOO MANY BARS

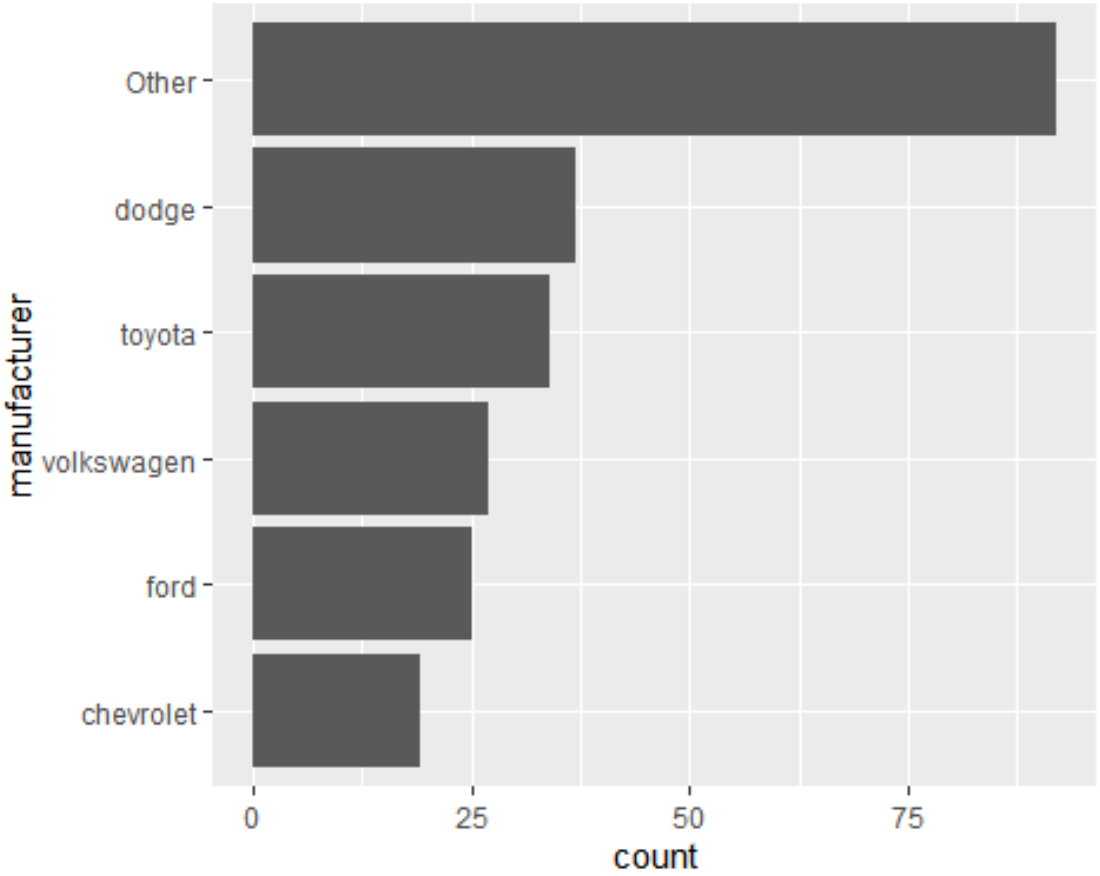
```
ggplot(mpg, aes(y = model)) +
  geom_bar()
```



Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

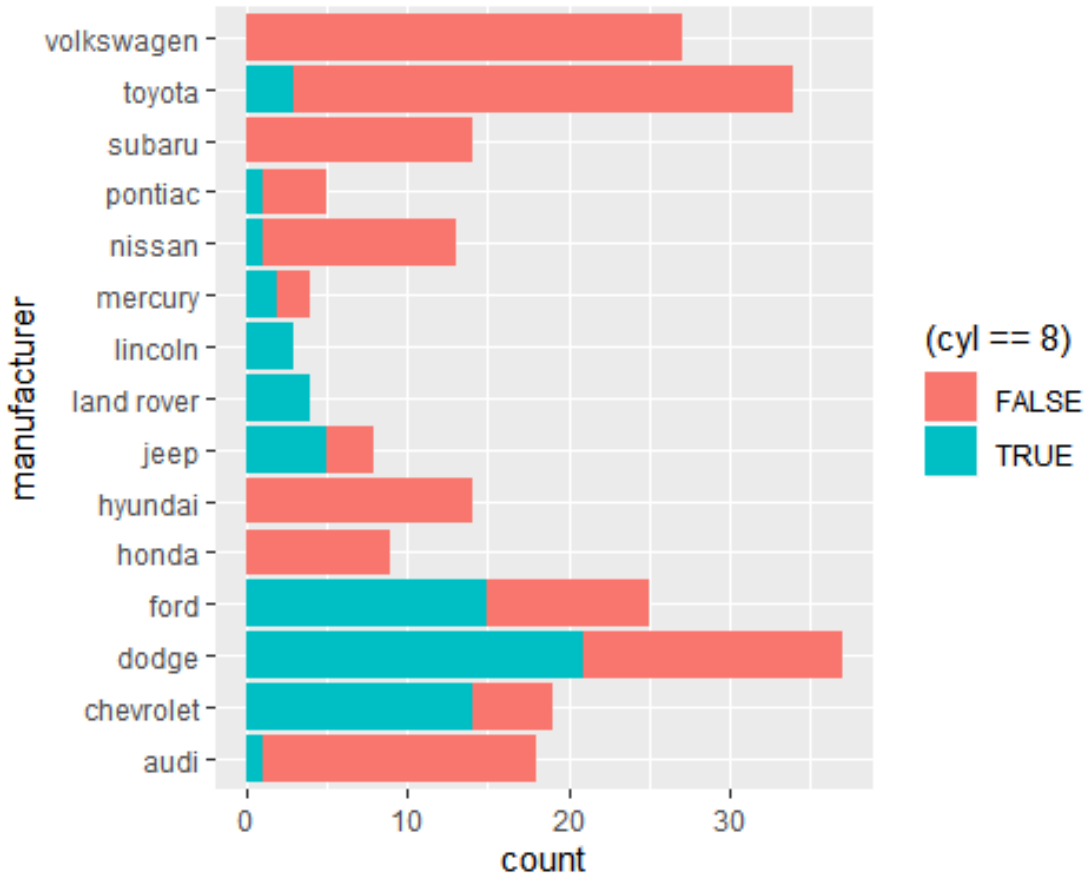
TOO MANY BARS: `fct_lump()`

```
mpg %>%
  mutate(
    manufacturer =
      fct_lump(manufacturer, n = 5) %>%
      fct_infreq() %>%
      fct_rev()
  ) %>%
  ggplot(aes(y = manufacturer)) +
  geom_bar()
```



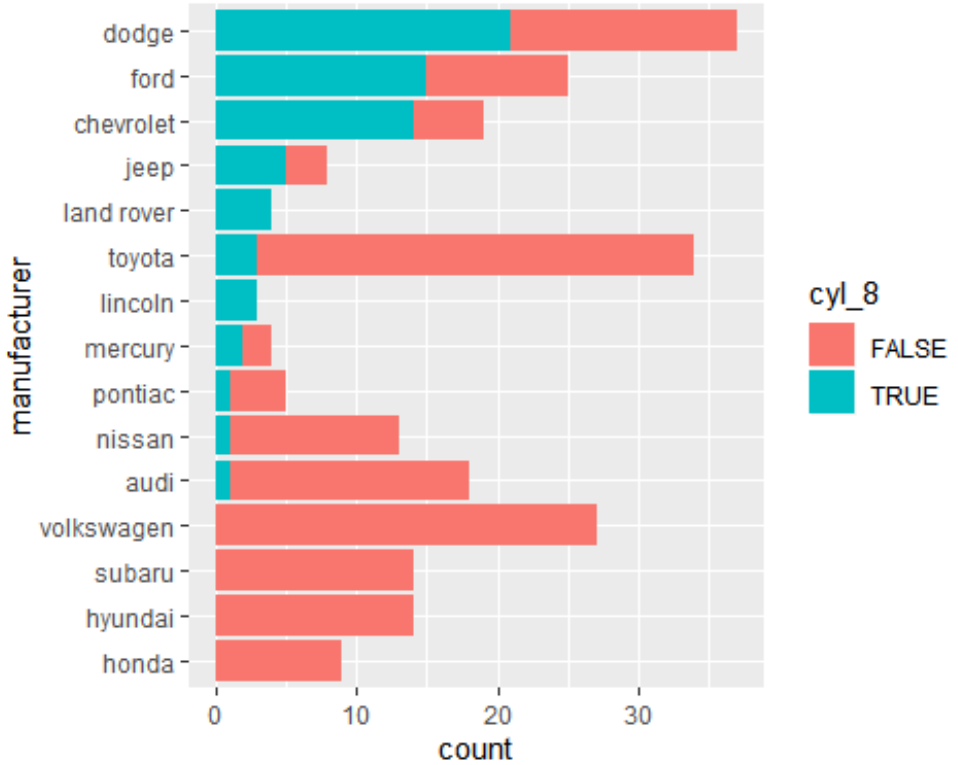
ORDER OF FILL

```
ggplot(mpg, aes(y = manufacturer, fill = (cyl == 8))) +
  geom_bar()
```



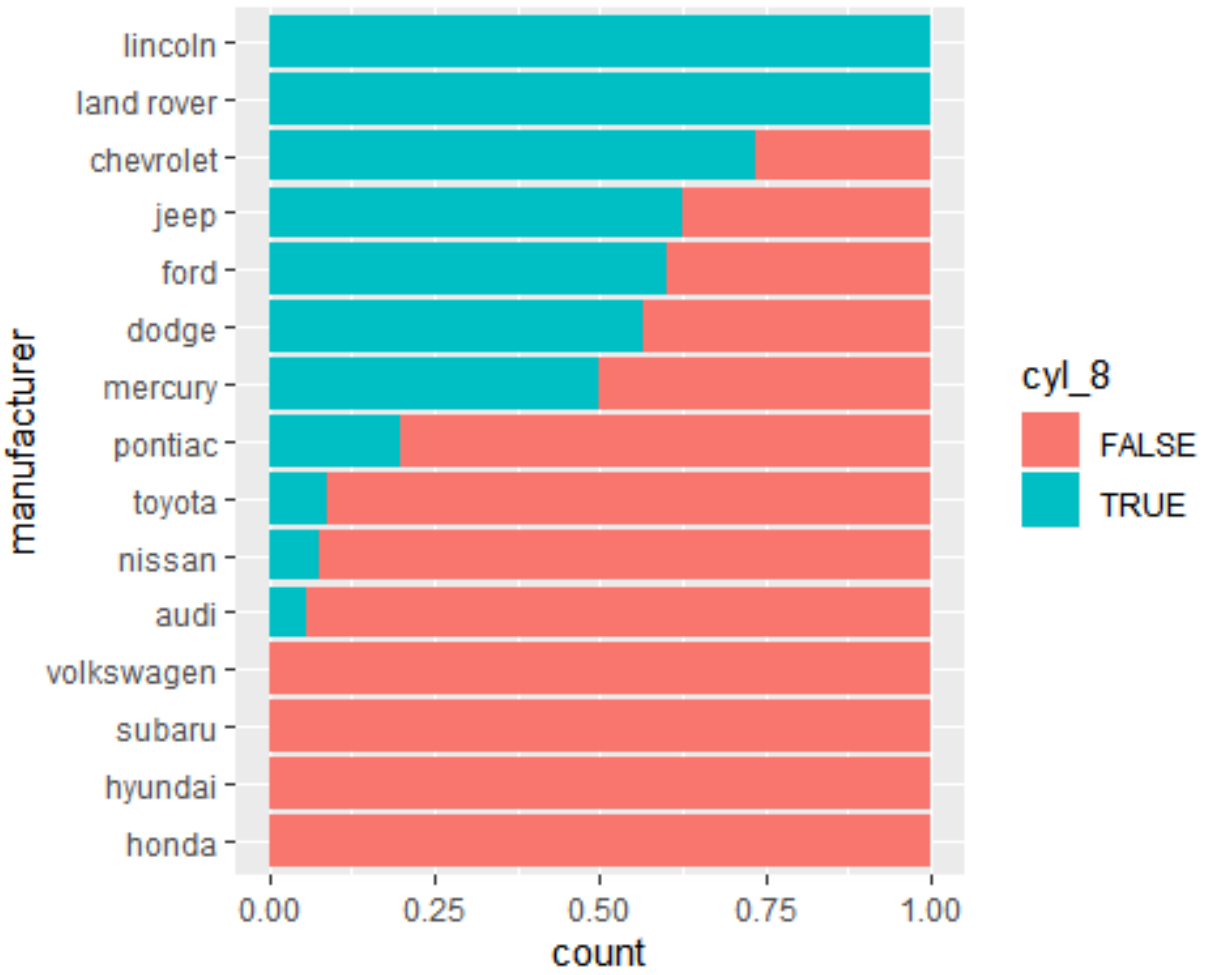
ORDER OF FILL

```
mpg %>%
  mutate(
    cyl_8 = (cyl == 8),
    manufacturer = fct_reorder(manufacturer, .x = cyl_8, .fun = sum)
  ) %>%
  ggplot(aes(y = manufacturer, fill = cyl_8)) +
  geom_bar()
```



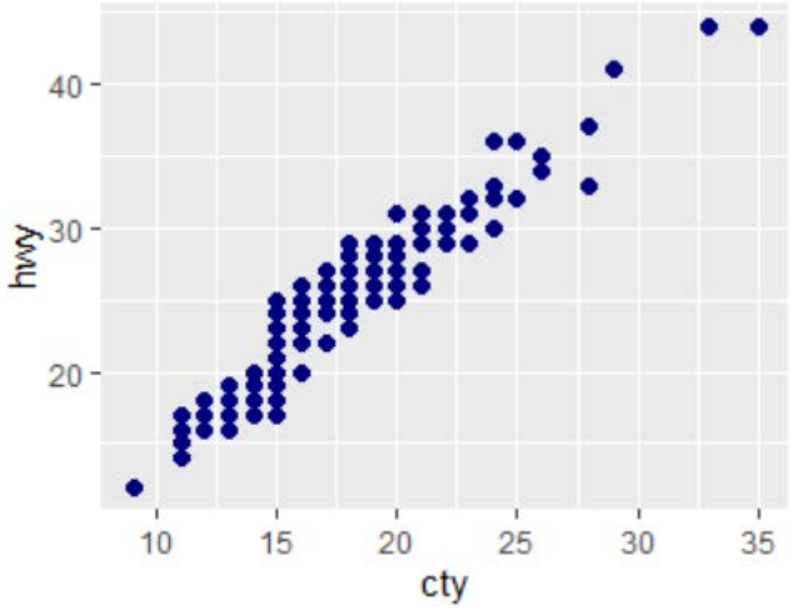
Q2 ANSWER

```
mpg %>%
  mutate(
    cyl_8 = (cyl == 8),
    manufacturer =
      fct_reorder(
        manufacturer,
        .x = cyl_8,
        .fun = mean
    )
  ) %>%
  ggplot(aes(manufacturer, fill = cyl_8))+
  geom_bar(position = "fill") +
  coord_flip()
```



SETTING DEFAULTS

```
ggplot(mpg) +
  geom_point(aes(cty, hwy), color = "navyblue", size = 2)

update_geom_defaults("point", list(color = "navyblue", size = 2))

ggplot(mpg) +
  geom_count(aes(cty, hwy))
```



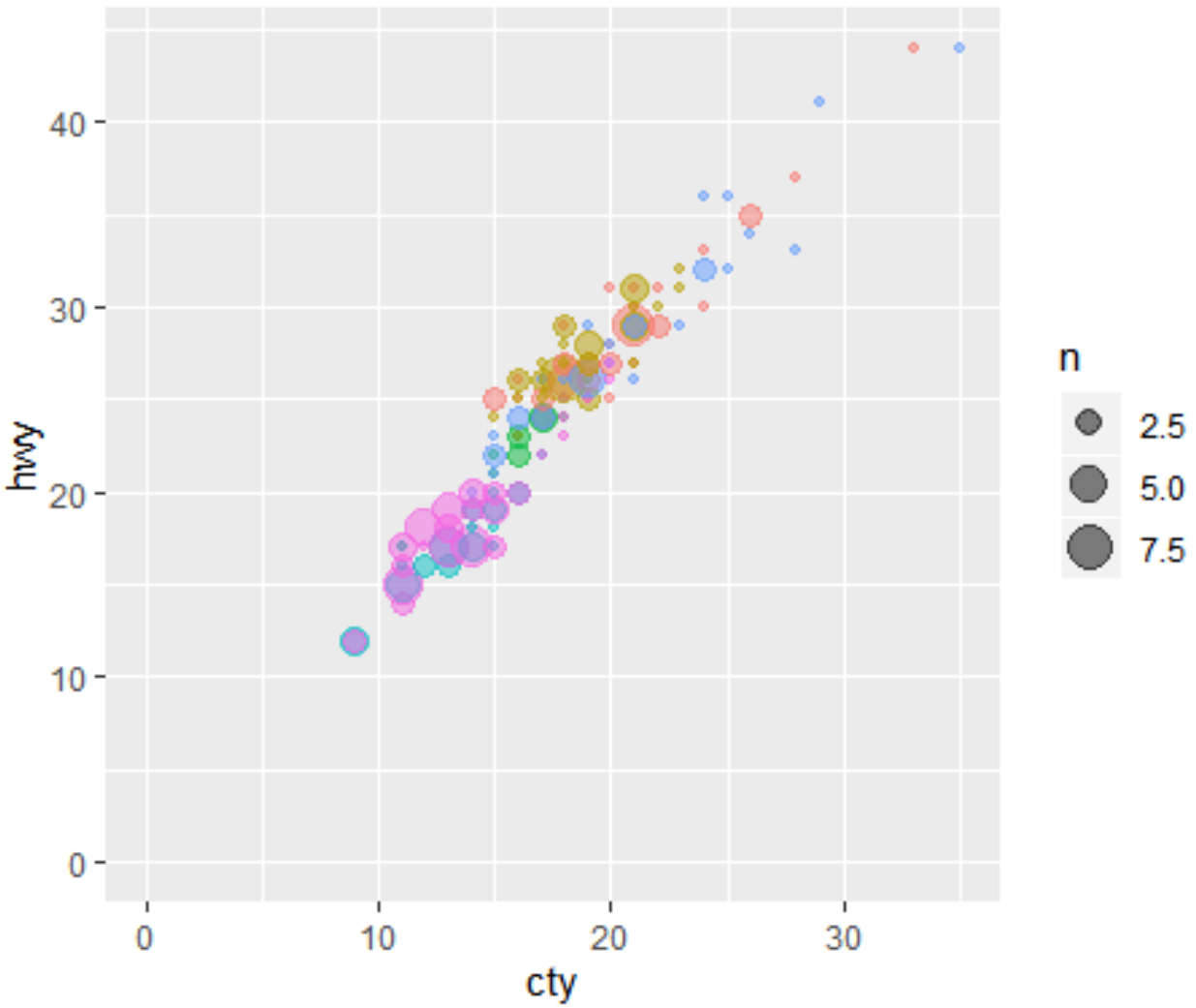
You have to do this for each geom

```
update_geom_defaults("col", list(fill = "navyblue"))
update_geom_defaults("bar", list(fill = "navyblue"))
```

Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

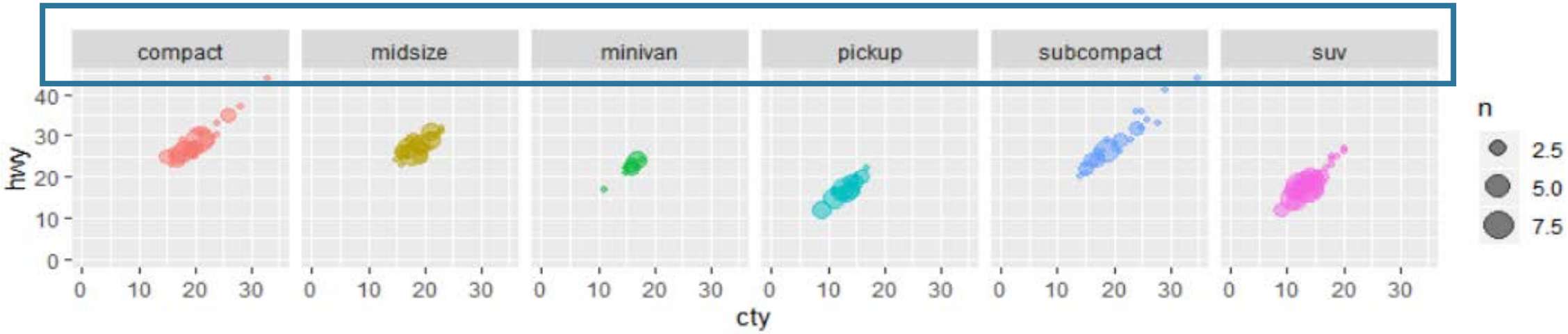
facet_grid() vs facet_wrap()

```
p <-
  mpg %>%
  filter(
    class != "2seater",
    cyl != 5
  ) %>%
  ggplot(aes(cty, hwy, color = class)) +
  geom_count(alpha = 0.5) +
  lims(x = c(0, NA), y = c(0, NA)) +
  guides(color = FALSE) +
  theme(aspect.ratio = 1)
```



facet_grid(): NEW SYNTAX

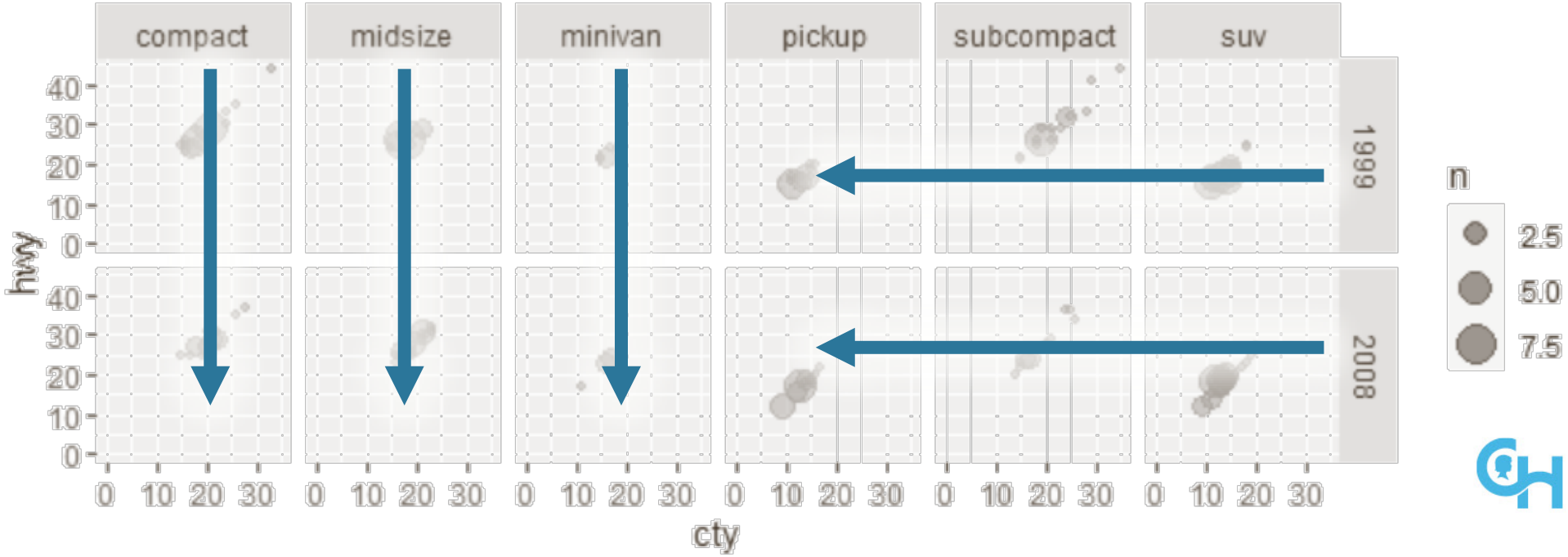
```
# this is the new syntax, replaces `facet_grid(~class)`
p +
  facet_grid(cols = vars(class))
```



Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

FACETS: SCALES

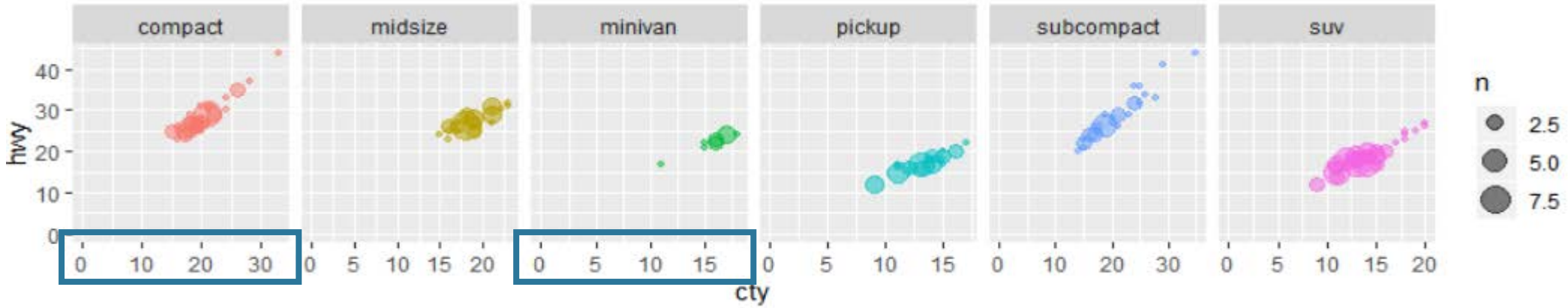
```
p +
  facet_grid(
    rows = vars(year),
    cols = vars(class)
  )
```



Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

FACETS: SCALES

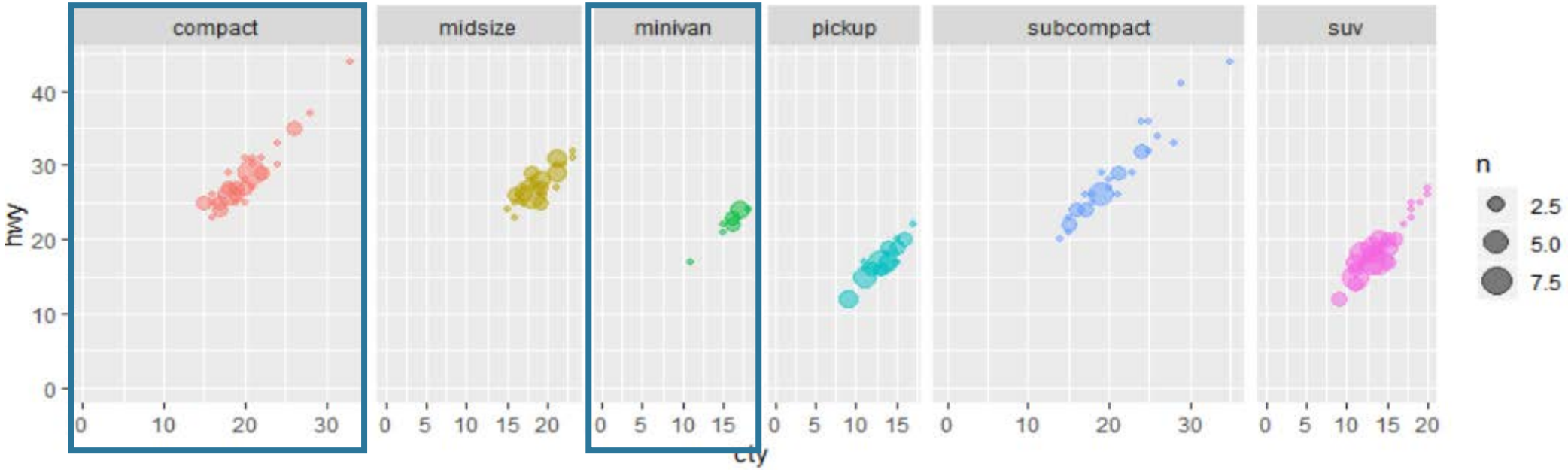
```
# scales allows the x & y to vary
# also "free_x", "free_y"
p +
  facet_grid(cols = vars(class), scales = "free")
```



Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

FACETS: SCALES & SPACE

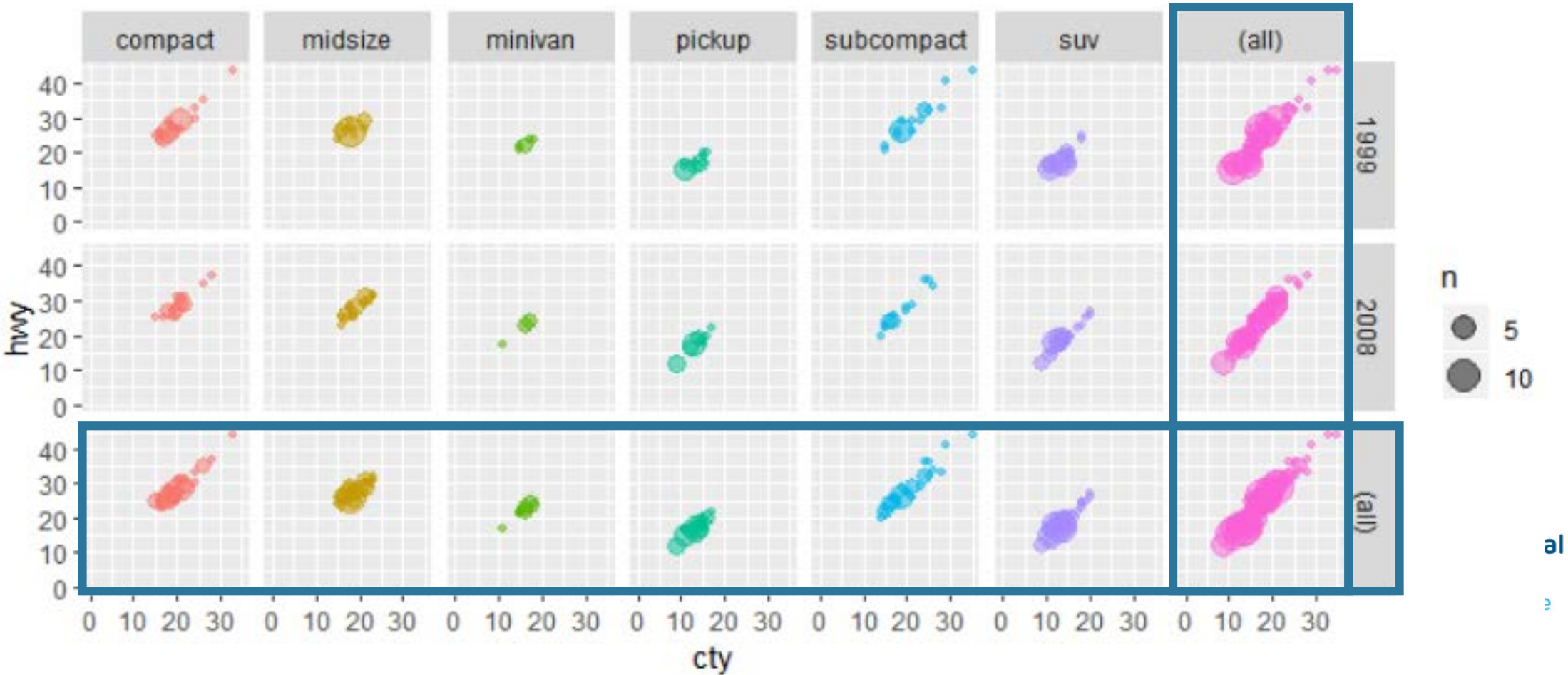
```
p +
  facet_grid(cols = vars(class), scales = "free", space = "free")
```



Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

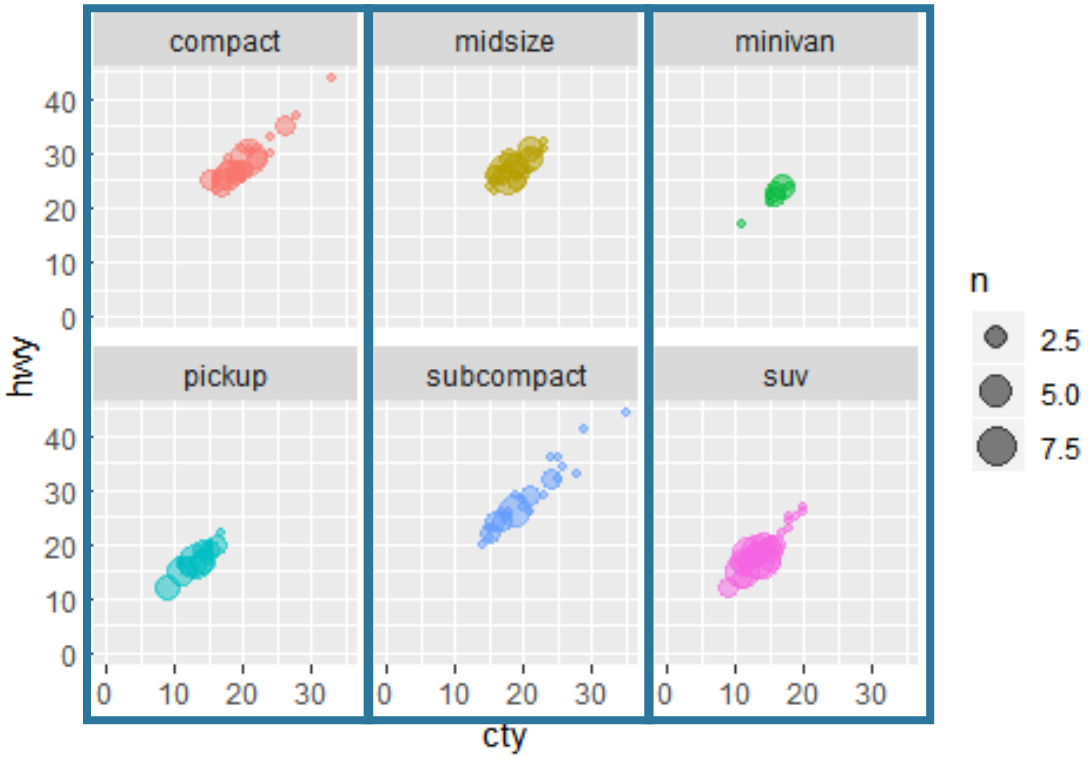
FACETS: MARGINS

```
p +
  facet_grid(rows = vars(year), cols = vars(class), margins = TRUE)
```



facet_wrap(): # OF COLUMNS/ROWS

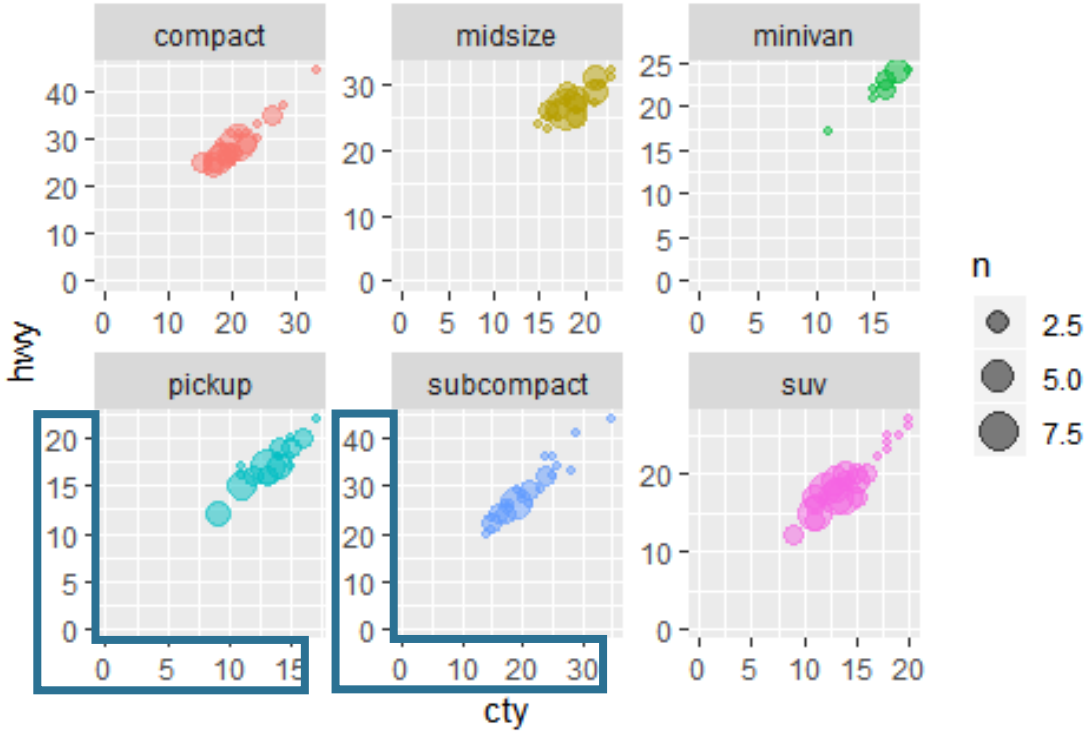
```
# also nrow
p +
  facet_wrap(~class, ncol = 3)
```



Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

FACET_WRAP: SCALES

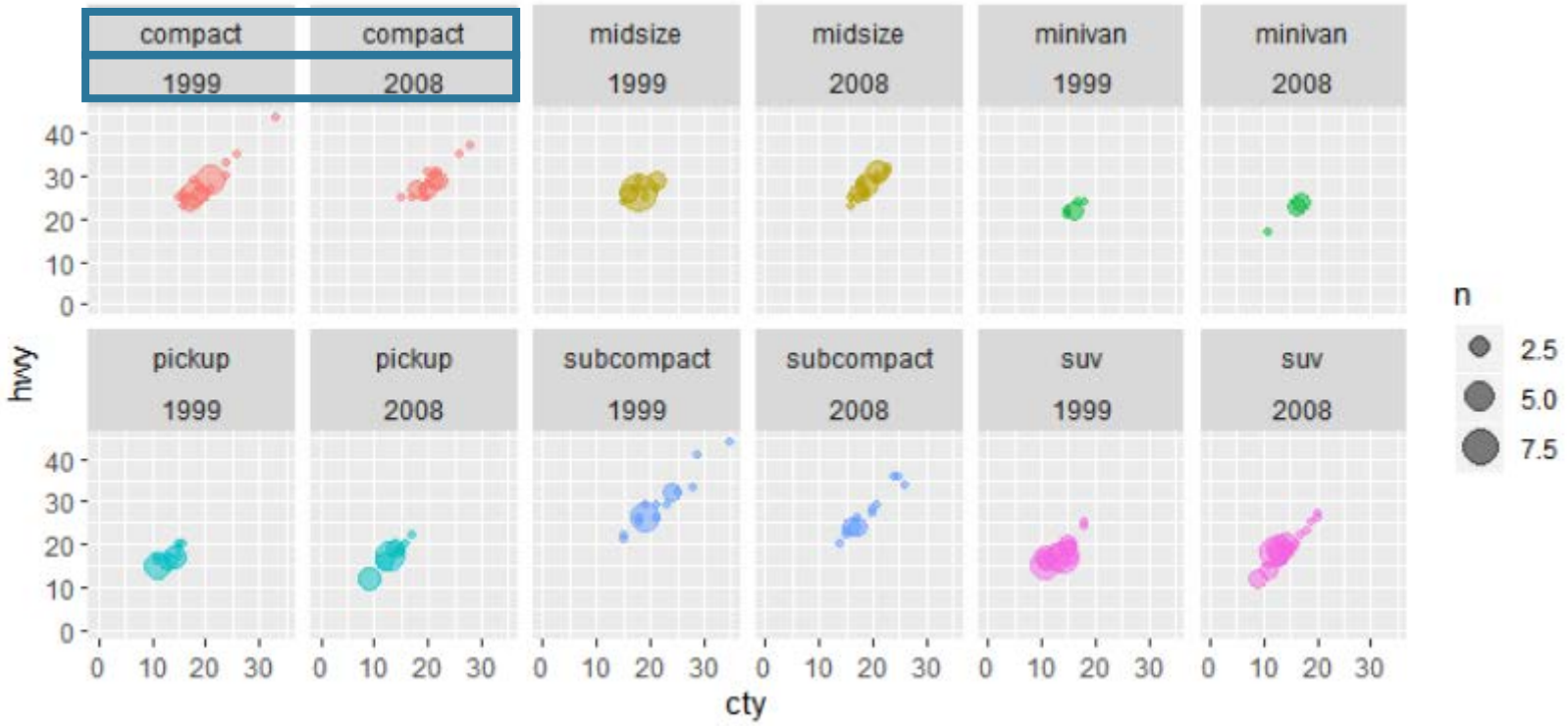
```
# space does not work with facet_wrap()
p + facet_wrap(~class, ncol = 3, scales = "free")
```



Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

FACETS: (A + B)

```
# also works with facet_grid
p +
  facet_wrap(~class + year, nrow = 2)
```



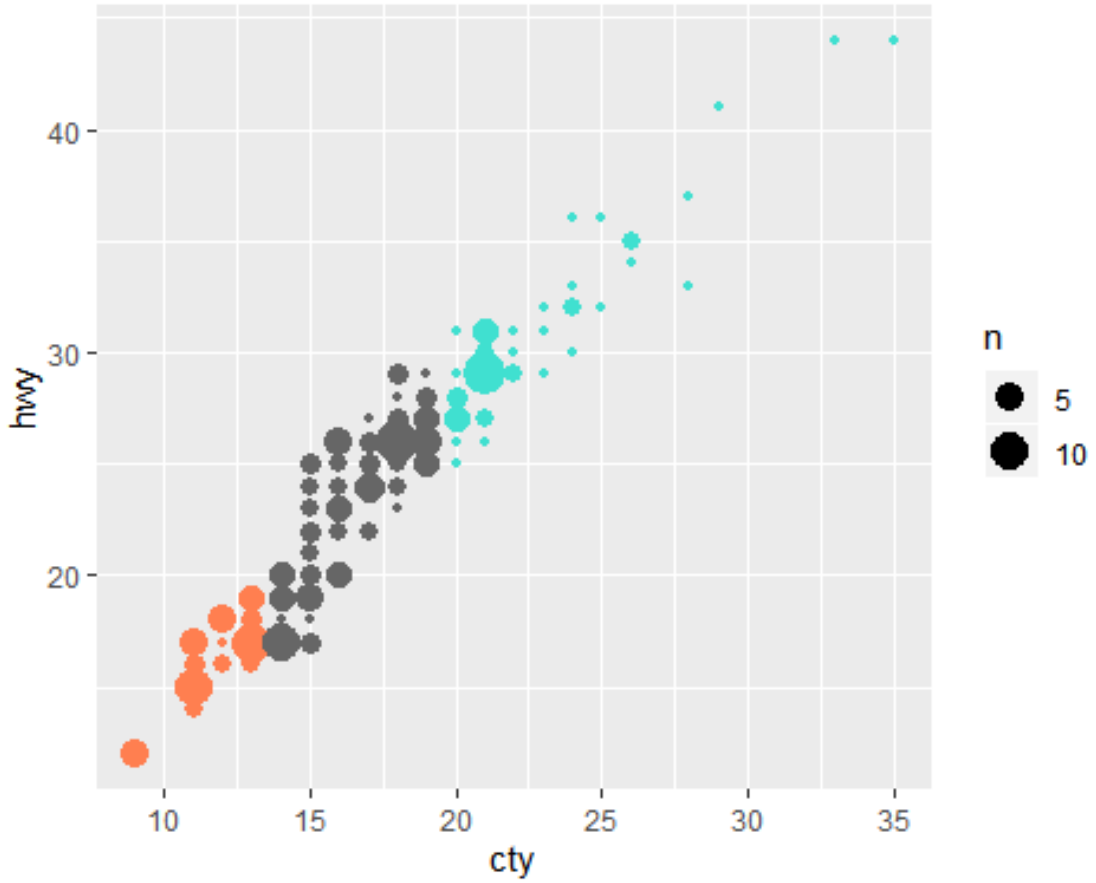
Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

scale_*_identity()

Sometimes I want to have better control over colors & sizes.

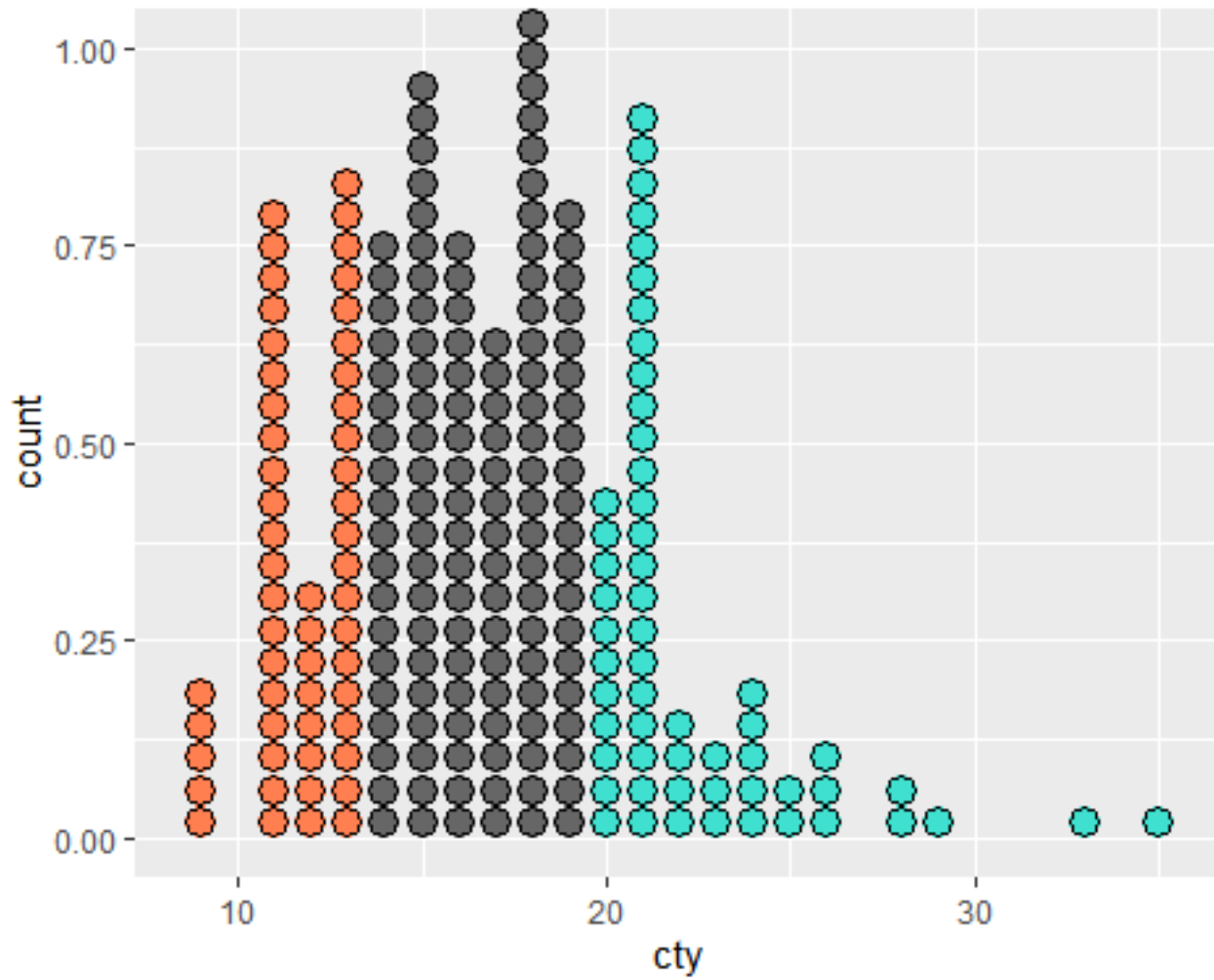
Here, I am hard coding the colors

```
df <-
  mpg %>%
  mutate(
    category =
      case_when(
        cty < 14 ~ "coral",
        cty > 19 ~ "turquoise",
        TRUE ~ "grey40"
      )
  )


ggplot(df, aes(cty, hwy, color = category)) +
  geom_count() +
  scale_color_identity()
```



SCALE_FILL_IDENTITY()

```
ggplot(df, aes(cty, fill = category)) +
  geom_dotplot() +
  scale_fill_identity()
```

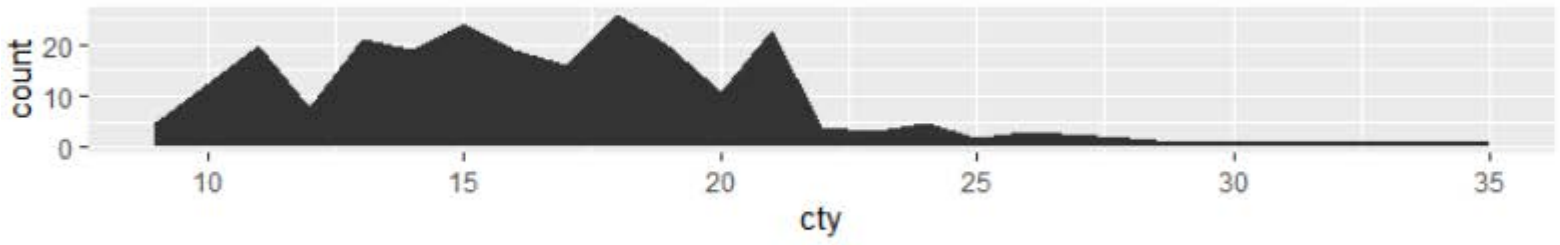


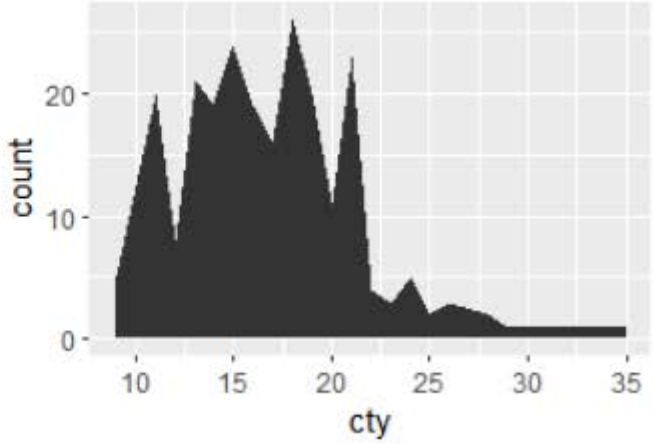
Best practices

THE GOLDEN RATIO **1:1.6**

- Try to give your charts the **proportion of a credit card**
- Also look this up

```
p <-
  ggplot(mpg, aes(cty)) +
  geom_area()

p + coord_fixed(1/10)

p + theme(aspect.ratio = 1/1.6) # ratio depends on the units
```





DEALING WITH SPAGHETTI CHARTS

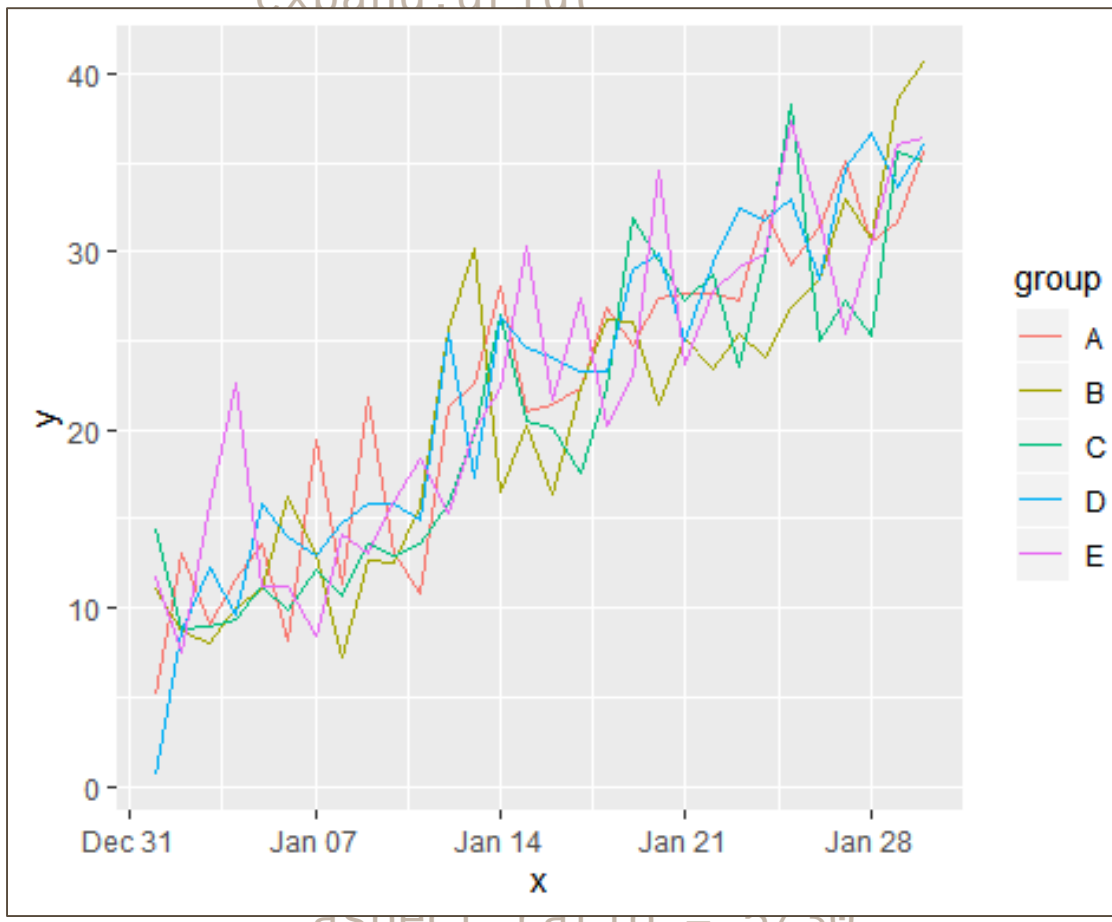
This is one of the most common questions:

multiple categories over **time**

This often results in a chart like the one here. It is hard to read but there are some ways you can **help your audience**

```
ggplot(df, aes(x, y, color = group)) +
  geom_line()
```

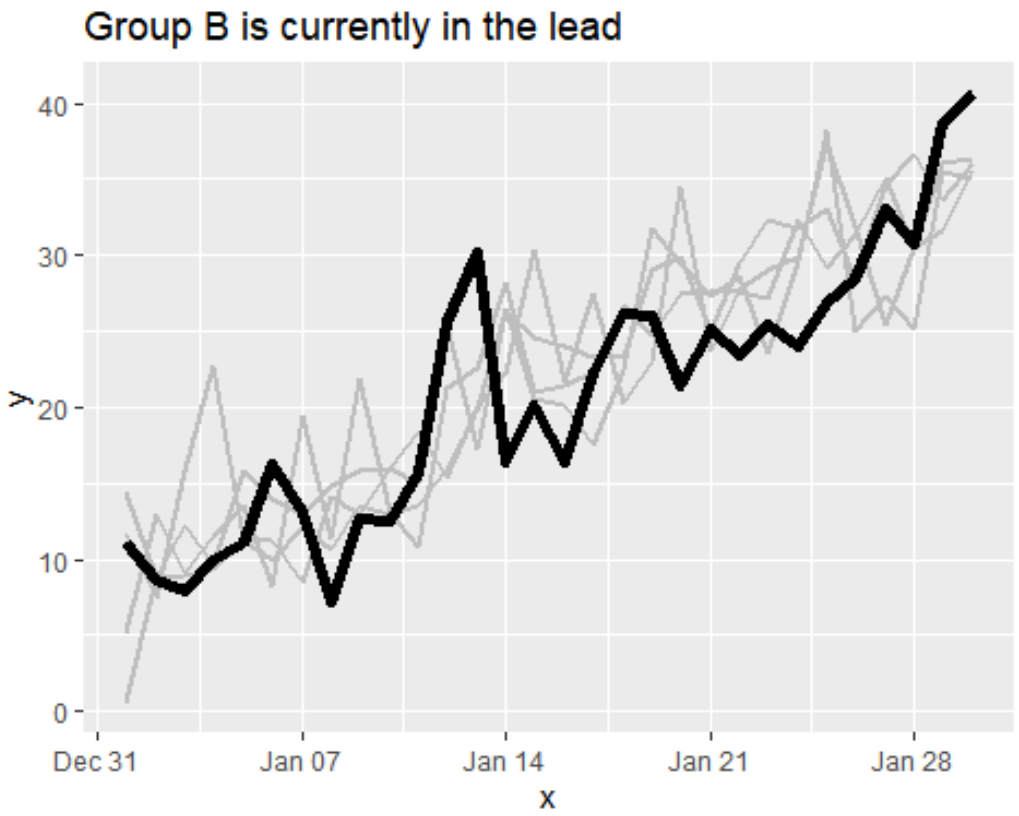
```
set.seed(1234)
df <-
  expand.grid(
```




```
  aspect.ratio = 5/30,
  legend.position = "bottom"
)
```

HIGHLIGHT THE FOCUS & USE AN INFORMATIVE TITLE

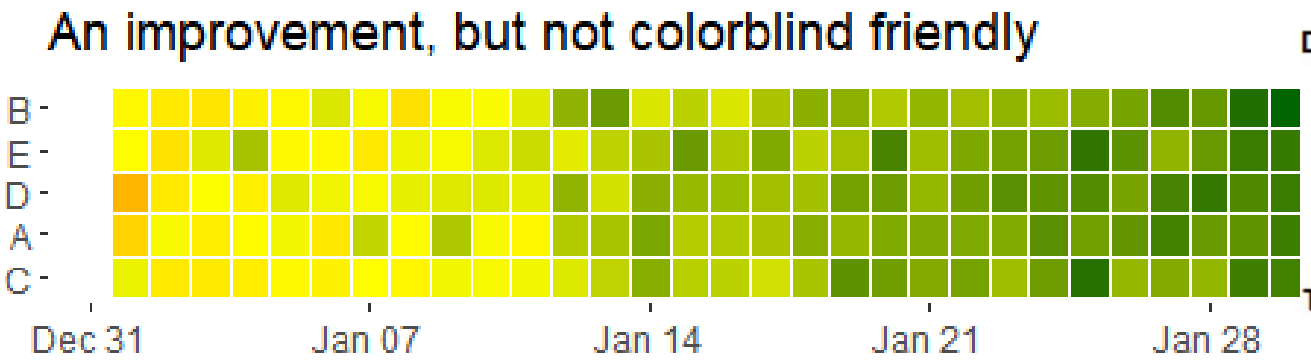
```
ggplot(df, aes(x, y, group = group)) +
  geom_line(data = filter(df, group != "B"), color = "grey", size = 1) +
  geom_line(data = filter(df, group == "B"), color = "black", size = 2) +
  labs(title = "Group B is currently in the lead")
```

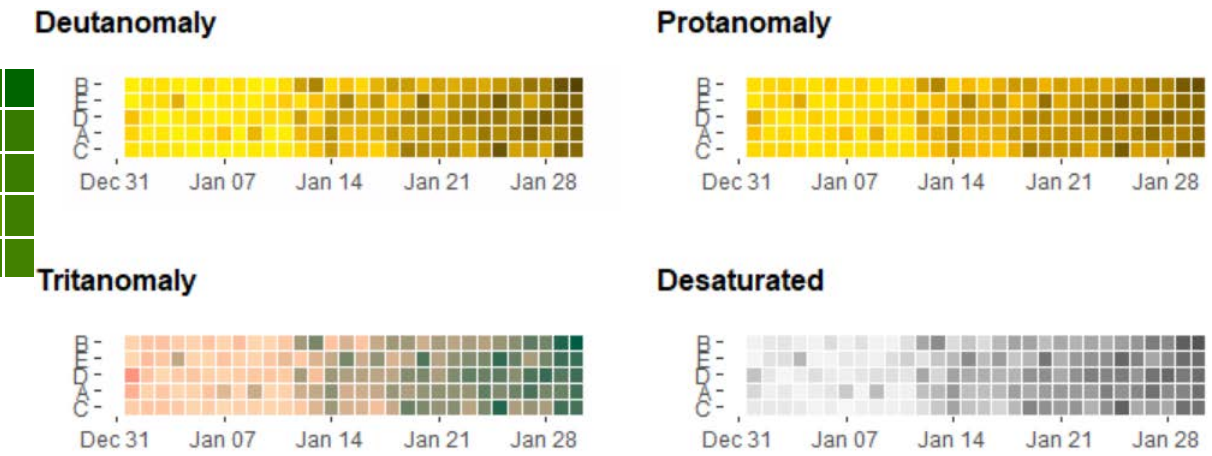


Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

TRY A HEATMAP BUT BEWARE

```
ggplot(df, aes(x, fct_reorder(group, y, last), fill = y)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(
    low = "red",
    mid = "yellow",
    high = "darkgreen",
    midpoint = 12
  ) +
  my_theme +
  labs(title = "An improvement, but not colorblind friendly")
```

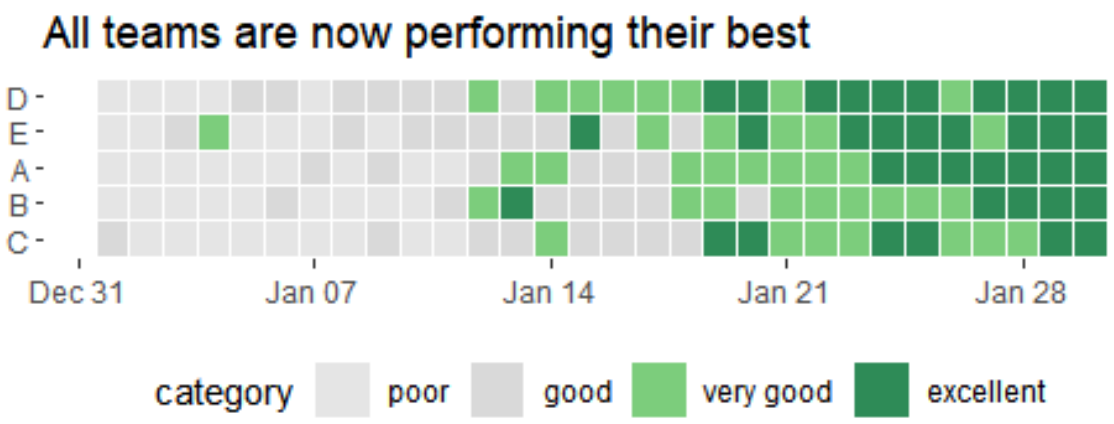


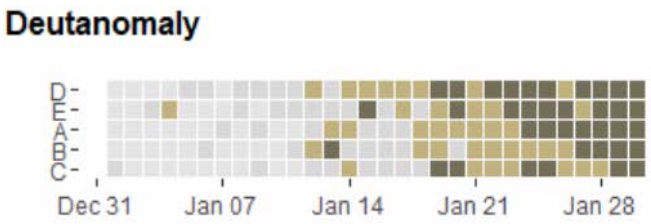


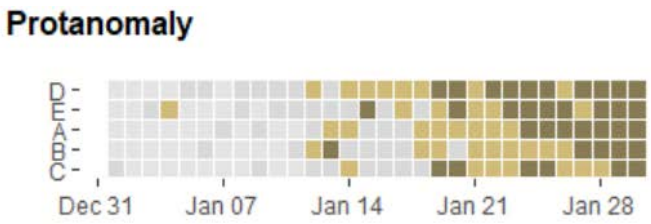

```
colorblindr::cvd_grid()
```

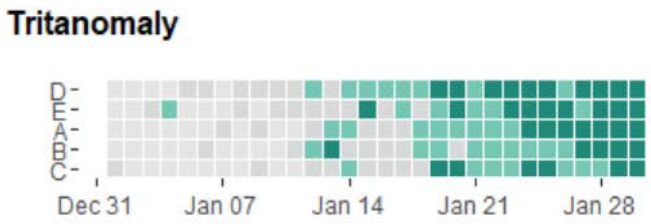
NOT EVERY POINT NEEDS A COLOR

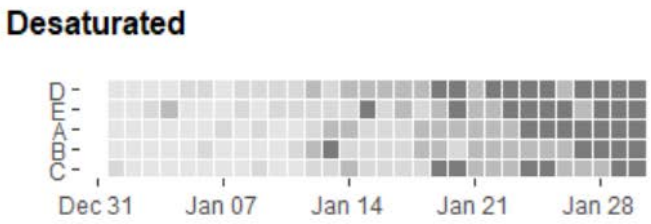
```
ggplot(df, aes(x, fct_reorder(group, y), fill = category)) +
  geom_tile(color = "white", size = 0.1) +
  scale_fill_manual(
    values = c(
      "poor"= "grey90",
      "good"= "grey85",
      "very good" = "palegreen3",
      "excellent" = "seagreen4"
    )
  ) +
  my_theme +
  labs(title = "All teams are now performing their best")
```





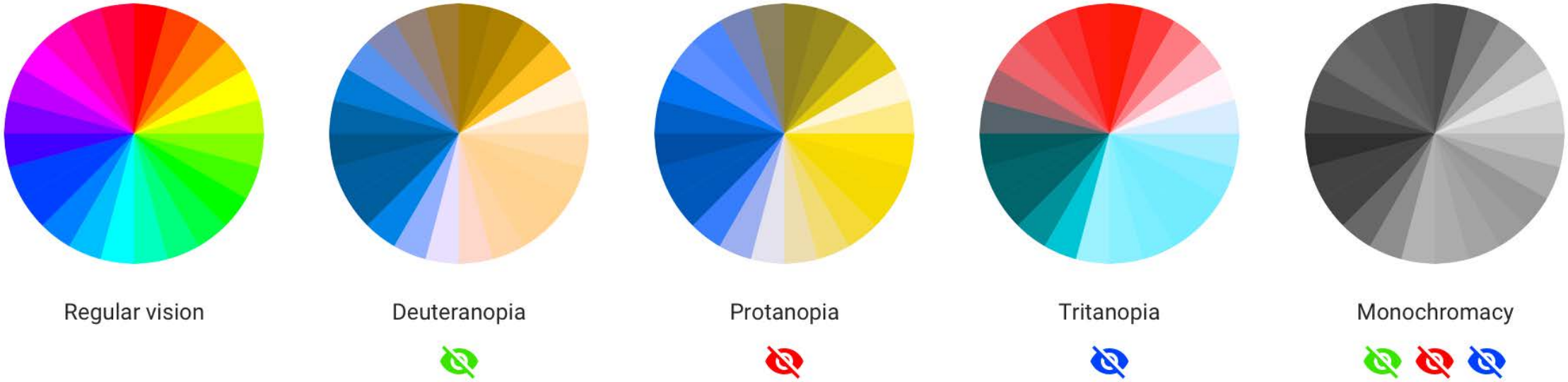






UNIVERSAL COLORS

- Keep in mind: nearly everyone can see **orange** and **blue**



accessibility

Children's Hospital of Philadelphia

TAKE CARE WHEN CROPPING DATA

The usual methods to “zoom in” can yield unexpected results when `stat_*()` geoms are used.

For example, `geom_boxplot()` calls `stat_boxplot()` and filters out data **before** doing the stats and your boxplot will keep readjusting the quartiles.

To zoom in, use `coord_cartesian(xlim = c(…), ylim = c(…))`

Children’s Hospital of Philadelphia®
Center for Healthcare Quality & Analytics

USE `coord_cartesian()` TO ZOOM IN

Do not use `ylim()` or `scale_*_continuous`

```
# find_limits() is a custom function
bind_rows(
  find_limits(p),
  find_limits(p + ylim(0, 12000)),
  find_limits(p + scale_y_continuous(limits = c(0, 12000))),
  find_limits(p + coord_cartesian(ylim = c(0, 12000)))
)
```

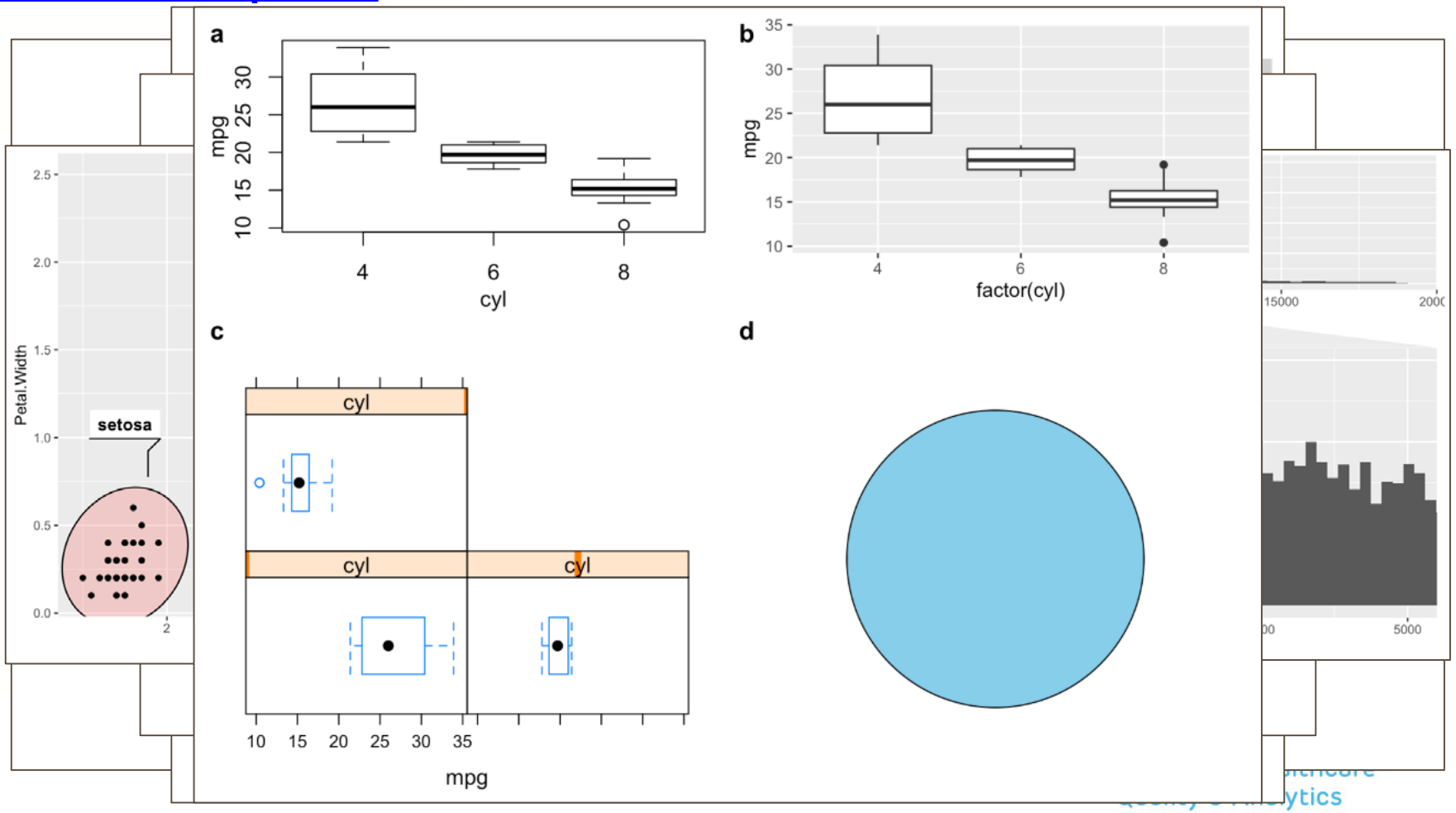
```
# lower middle upper
#   950   2401  5324
#   911   2161  4679
#   911   2161  4679
#   950   2401  5324
```

Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

Extensions & Add-ins

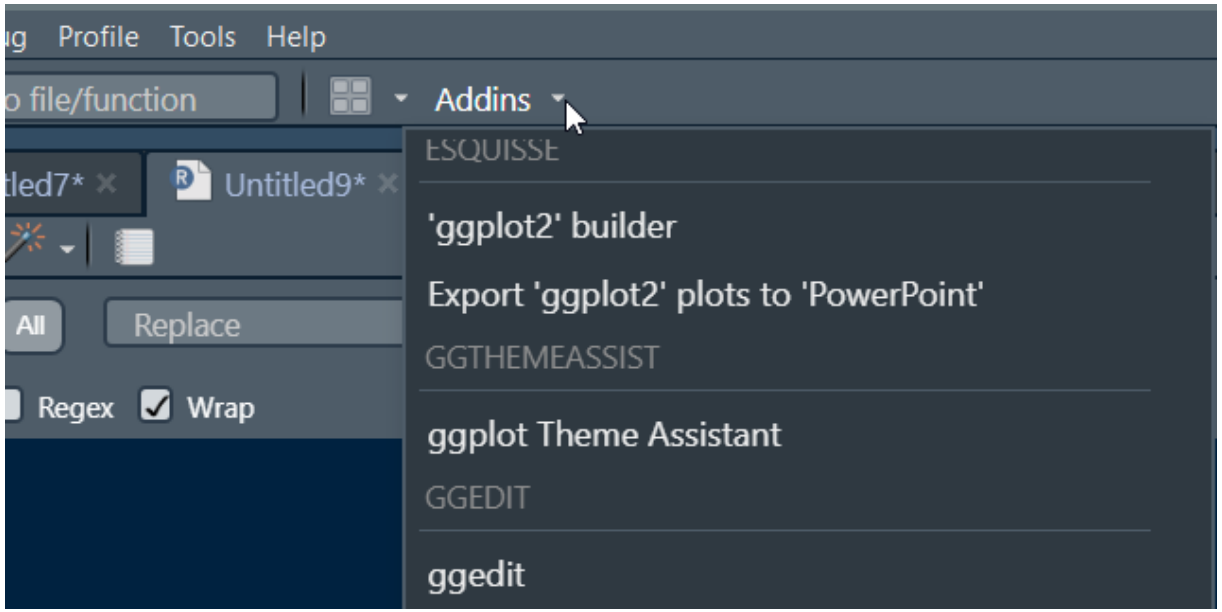
EXTENSIONS

- ggradar - spider/radar plots
- gganimate
- ggrepel
- ggforce
- ggtext
- cowplot
- more



ADD-INS

```
p <-
  ggplot(mpg, aes(cty, hwy)) +
  geom_point()


# install.packages("ggThemeAssist")
ggThemeAssist::ggThemeAssistGadget(p)

# install.packages("ggedit")
ggedit::ggedit(p)

# install.packages("esquisse")
esquisse:::esquisser()
esquisse:::esquisser(mpg)

# install.packages("colourpicker")
colourpicker::colourPicker()

# install.packages("addinslist")
addinslist::addinslistAddin()
```



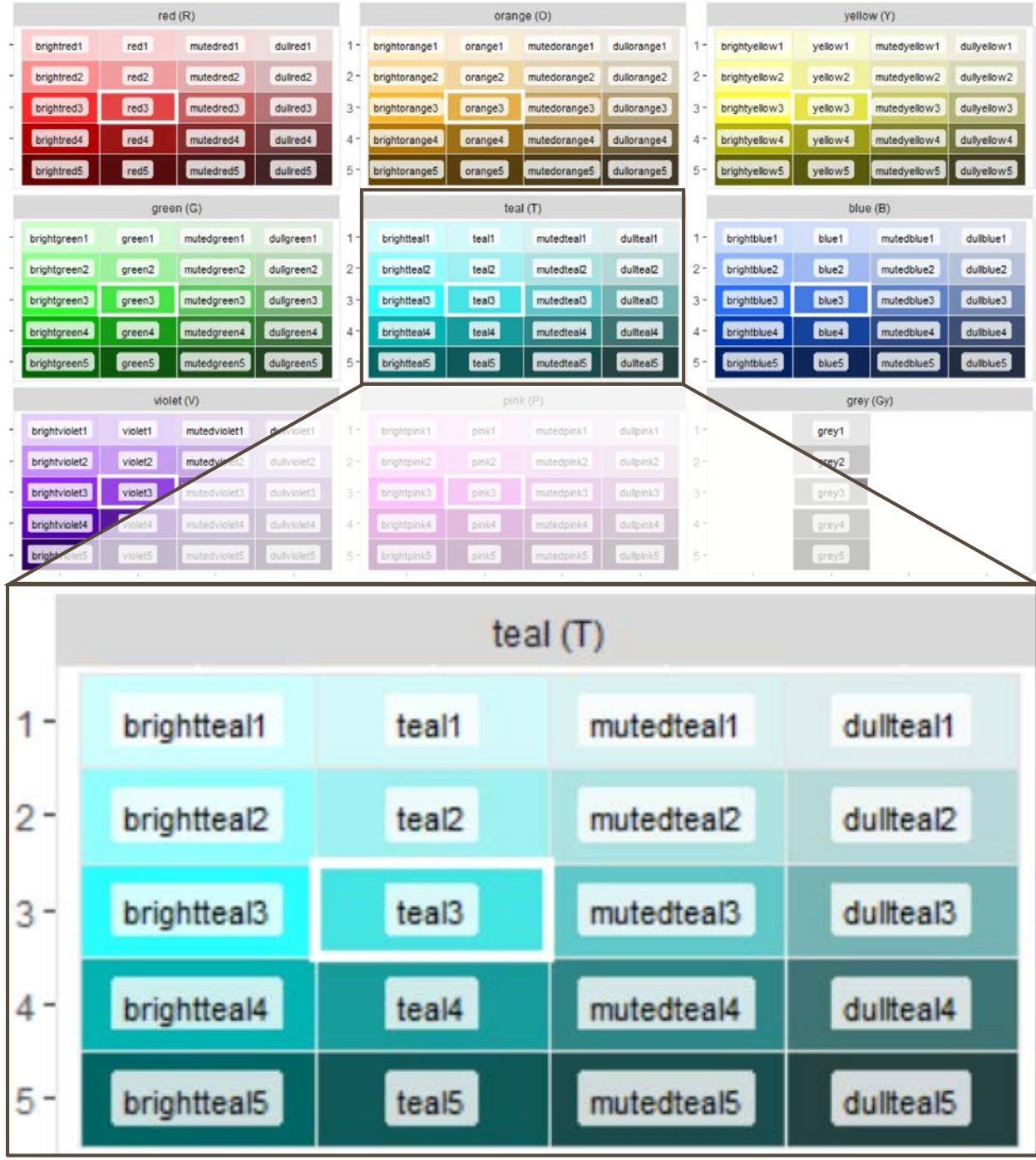
SIMPLECOLORS

uniformly named colors

rjake.github.io/simplecolors/articles/intro.html

```
library(simplecolors)
show_colors()
```

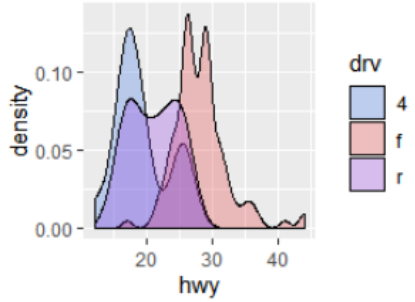
```
sc(
  "brightblue4",
  "mutedorange5",
  "grey3"
)
```

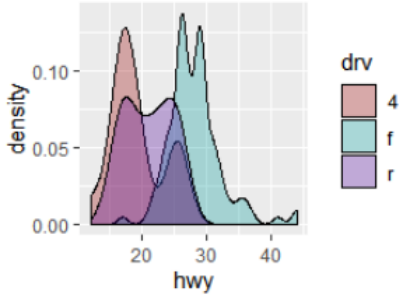


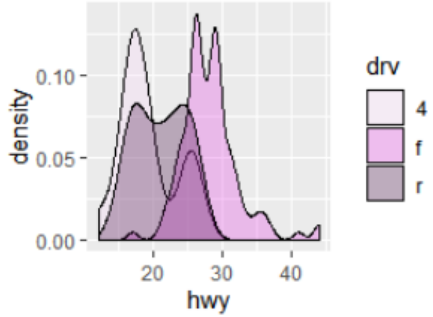
SIMPLECOLORS

Three main functions: `sc()` `sc_across()` `sc_*()`

```
p <-
  ggplot(mpg, aes(hwy, fill = drv)) +
  geom_density(alpha = 0.3)

p +
  scale_fill_manual(values = sc("blue3", "red3", "violet3"))
                          # or sc_across("BRV", 3)

p +
  scale_fill_manual(
    values = sc_across("RTV", light = 4, sat = "bright")
  )

p +
  scale_fill_manual(values = sc_pink(light = c(1,3,5)))
```



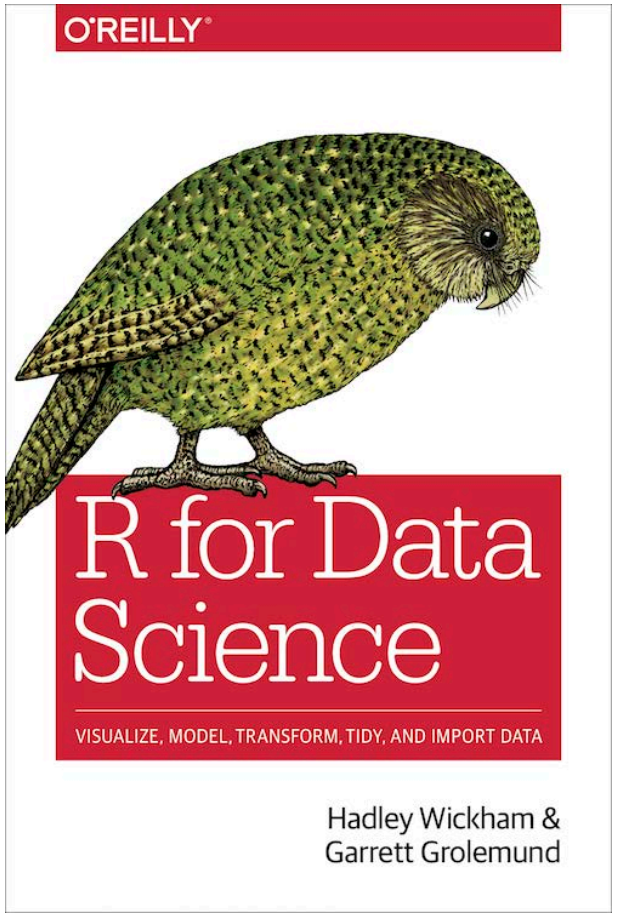




EXTRA HELP

R4DS

R for Data Science is a book all about the `tidyverse`. It is less “data science-y” and more about data manipulation and visualization. It is free online here as well as available for sale.

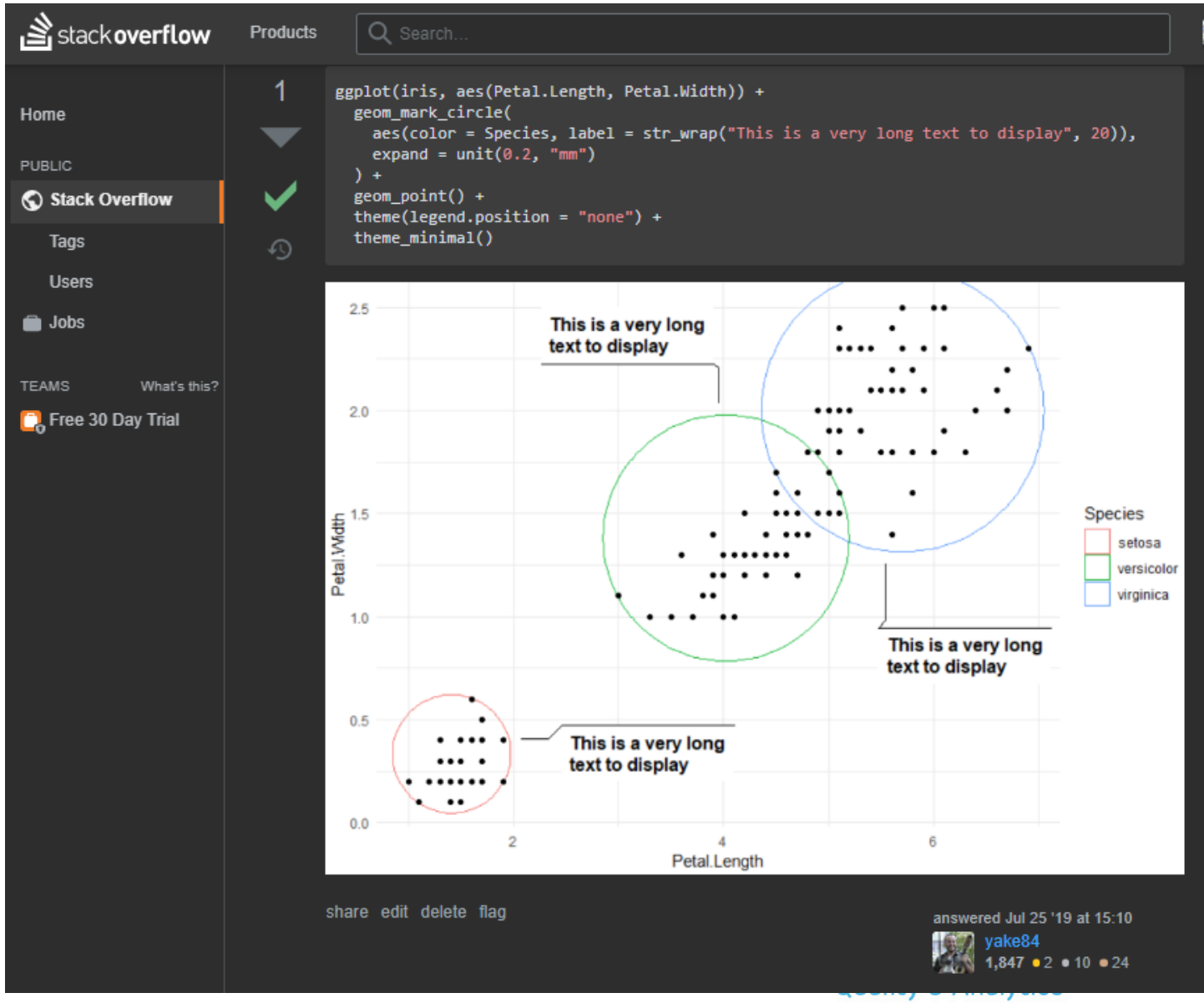


Children's Hospital of Philadelphia
Center for Healthcare Quality & Analytics

STACKOVERFLOW

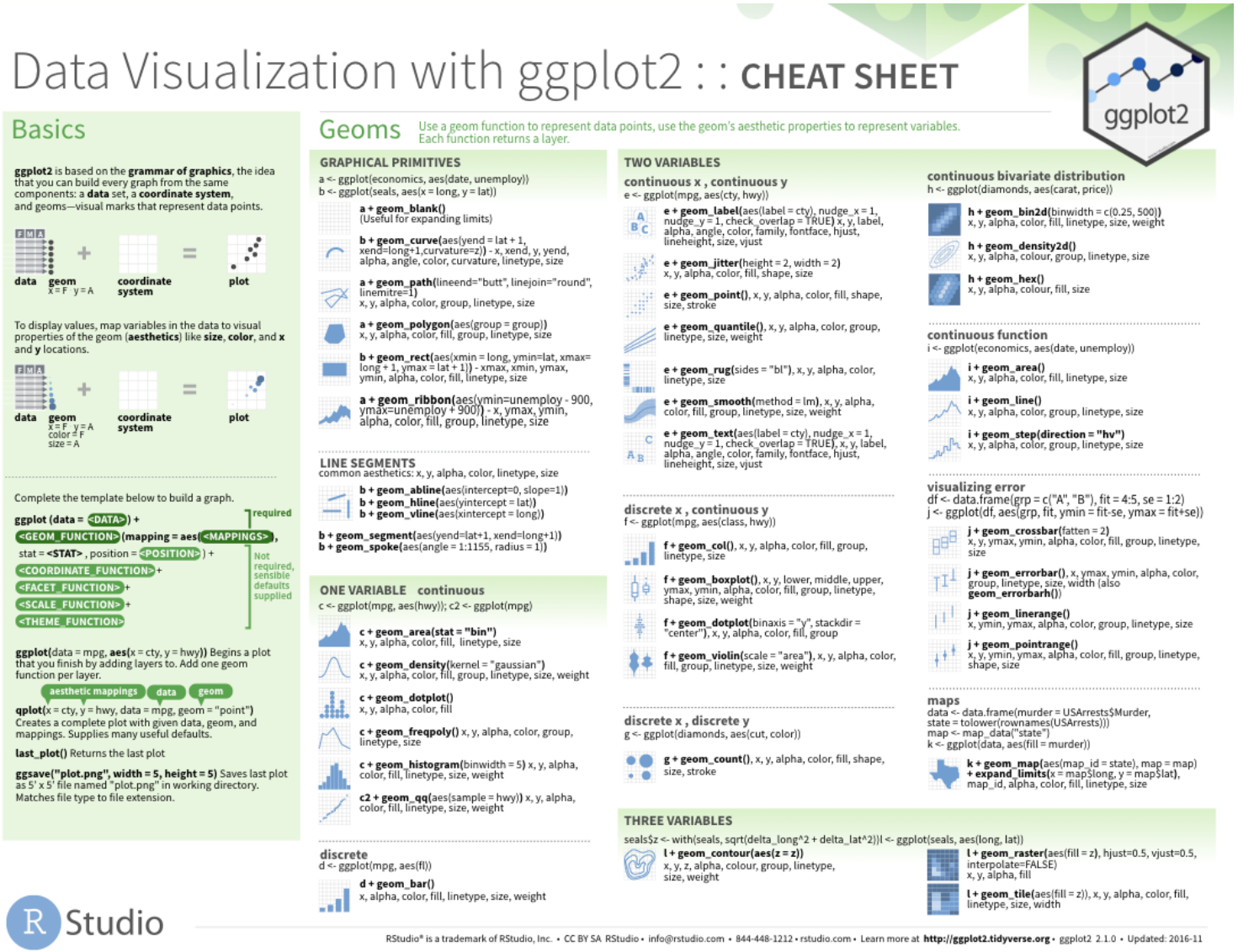
- try `datapasta` and `reprex` for a minimal **rep**roducible **ex**ample

- include images rather than links



CHEATSHEET

https://github.com/rstudio/cheatsheets/raw/master/data-visualization-2.1.pdf



Data Visualization with ggplot2 : : CHEAT SHEET

Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and geoms—visual marks that represent data points.

data | geom (x = F, y = A) | coordinate system | plot

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

data | geom (x = F, y = A, color = F, size = A) | coordinate system | plot

Complete the template below to build a graph.

```
ggplot (data = <DATA>) +
<GEOM_FUNCTION> (mapping = aes(<MAPPINGS>),
stat = <STAT>, position = <POSITION>) +
<COORDINATE_FUNCTION> +
<FACET_FUNCTION> +
<SCALE_FUNCTION> +
<THEME_FUNCTION>
```

required

Not required, sensible defaults supplied

ggplot(data = mpg, **aes**(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

aesthetic mappings | data | geom

qplot(x = cty, y = hwy, data = mpg, geom = "point") Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

last_plot() Returns the last plot

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

GRAPHICAL PRIMITIVES

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

a + geom_blank() (Useful for expanding limits)

b + geom_curve(aes(yend = lat + 1, xend=long+1,curvature=z)) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + geom_path(lineend="butt", linejoin="round", linemitre=1) x, y, alpha, color, group, linetype, size

a + geom_polygon(aes(group = group)) x, y, alpha, color, fill, group, linetype, size

b + geom_rect(aes(xmin = long, ymin=lat, xmax= long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900)) - x, ymax, ymin, alpha, color, fill, group, linetype, size

LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

b + geom_abline(aes(intercept=0, slope=1))
b + geom_hline(aes(yintercept = lat))
b + geom_vline(aes(xintercept = long))

b + geom_segment(aes(yend=lat+1, xend=long+1))
b + geom_spoke(aes(angle = 1:1155, radius = 1))

ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + geom_area(stat = "bin") x, y, alpha, color, fill, linetype, size

c + geom_density(kernel = "gaussian") x, y, alpha, color, fill, group, linetype, size, weight

c + geom_dotplot() x, y, alpha, color, fill

c + geom_freqpoly() x, y, alpha, color, group, linetype, size

c + geom_histogram(binwidth = 5) x, y, alpha, color, fill, linetype, size, weight

c2 + geom_qq(aes(sample = hwy)) x, y, alpha, color, fill, linetype, size, weight

discrete

d <- ggplot(mpg, aes(fl))

d + geom_bar() x, alpha, color, fill, linetype, size, weight

TWO VARIABLES

continuous x , continuous y

e <- ggplot(mpg, aes(cty, hwy))

e + geom_label(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE) x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + geom_jitter(height = 2, width = 2) x, y, alpha, color, fill, shape, size

e + geom_point(), x, y, alpha, color, fill, shape, size, stroke

e + geom_quantile(), x, y, alpha, color, group, linetype, size, weight

e + geom_rug(sides = "bl"), x, y, alpha, color, linetype, size

e + geom_smooth(method = lm), x, y, alpha, color, fill, group, linetype, size, weight

e + geom_text(aes(label = cty), nudge_x = 1, nudge_y = 1, check_overlap = TRUE), x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

discrete x , continuous y

f <- ggplot(mpg, aes(class, hwy))

f + geom_col(), x, y, alpha, color, fill, group, linetype, size

f + geom_boxplot(), x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + geom_dotplot(binaxis = "y", stackdir = "center"), x, y, alpha, color, fill, group

f + geom_violin(scale = "area"), x, y, alpha, color, fill, group, linetype, size, weight

discrete x , discrete y

g <- ggplot(diamonds, aes(cut, color))

g + geom_count(), x, y, alpha, color, fill, shape, size, stroke

continuous bivariate distribution

h <- ggplot(diamonds, aes(carat, price))

h + geom_bin2d(binwidth = c(0.25, 500)) x, y, alpha, color, fill, linetype, size, weight

h + geom_density2d() x, y, alpha, colour, group, linetype, size

h + geom_hex() x, y, alpha, colour, fill, size

continuous function

i <- ggplot(economics, aes(date, unemploy))

i + geom_area() x, y, alpha, color, fill, linetype, size

i + geom_line() x, y, alpha, color, group, linetype, size

i + geom_step(direction = "hv") x, y, alpha, color, group, linetype, size

visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

j + geom_crossbar(fatten = 2) x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + geom_errorbar(), x, ymax, ymin, alpha, color, group, linetype, size, width (also **geom_errorbarh()**)

j + geom_linerange() x, ymin, ymax, alpha, color, group, linetype, size

j + geom_pointrange() x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps

data <- data.frame(murder = USArrests$Murder, state = tolower(rownames(USArrests)))
map <- map_data("state")
k <- ggplot(data, aes(fill = murder))

k + geom_map(aes(map_id = state), map = map) **+ expand_limits**(x = map$long, y = map$lat), map_id, alpha, color, fill, linetype, size

THREE VARIABLES

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))l <- ggplot(seals, aes(long, lat))

l + geom_contour(aes(z = z)) x, y, z, alpha, colour, group, linetype, size, weight

l + geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE) x, y, alpha, fill

l + geom_tile(aes(fill = z)), x, y, alpha, color, fill, linetype, size, width

RStudio

RStudio® is a trademark of RStudio, Inc. • CC BY SA RStudio • info@rstudio.com • 844-448-1212 • rstudio.com • Learn more at **http://ggplot2.tidyverse.org** • ggplot2 2.1.0 • Updated: 2016-11

Children's Hospital of Philadelphia®
Center for Healthcare Quality & Analytics