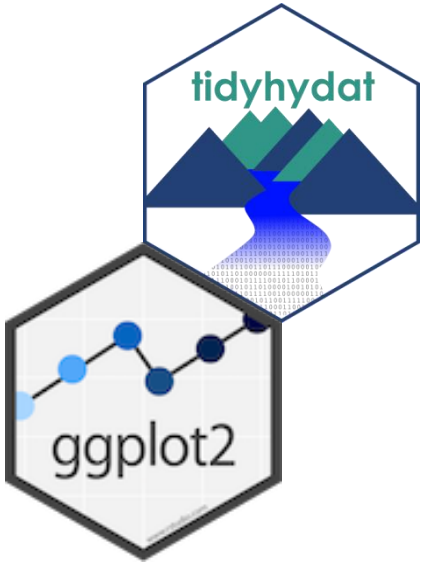
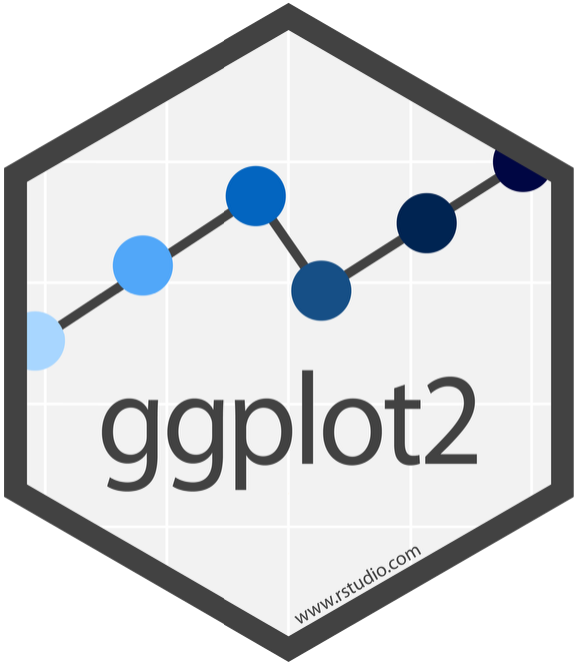
Visualize Data with



"The simple graph has brought more information to the data analyst's mind than any other device. "

- John Tukey

ggplot2



One of the earliest members of the tidyverse.

Complicated plots, come from combining simple components.

Data for this section

The number of stations and average length of record by real time and activity status

```
data_range <- hy_stn_data_range() %>%
  group_by(STATION_NUMBER) %>%
  summarise(total_record_length = sum(RECORD_LENGTH, na.rm =
TRUE))

station_meta <- hy_stations() %>%
  group_by(PROV_TERR_STATE_LOC, HYD_STATUS, REAL_TIME) %>%
  left_join(data_range, by = c("STATION_NUMBER")) %>%
  summarise(number_stns = n(),
            mean_record_length = mean(total_record_length,
                                      na.rm = TRUE))
```

station_meta

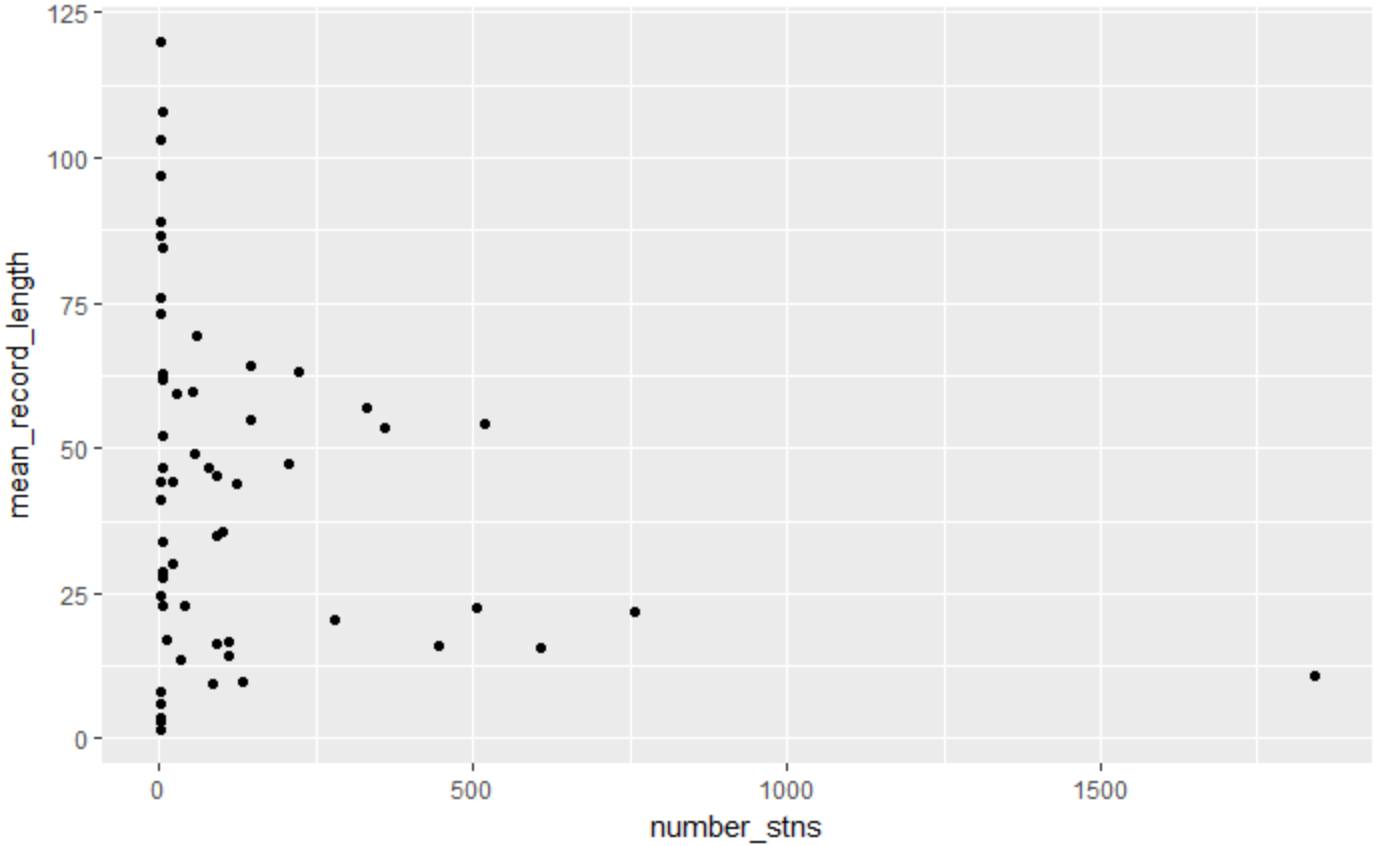
```
# A tibble: 58 x 5
# Groups:   PROV_TERR_STATE_LOC, HYD_STATUS [?]
   PROV_TERR_STATE_LOC HYD_STATUS   REAL_TIME number_stns mean_record_length
   <chr>               <chr>        <lgl>           <int>              <dbl>
 1 AB                  ACTIVE       FALSE             123               43.9
 2 AB                  ACTIVE       TRUE              331               57
 3 AB                  DISCONTINUED FALSE             609               15.8
 4 AB                  NA           FALSE               2                3
 5 AK                  ACTIVE       TRUE                4               28.8
 6 BC                  ACTIVE       FALSE              78               46.6
 7 BC                  ACTIVE       TRUE              360               53.6
 8 BC                  DISCONTINUED FALSE            1842               10.9
 9 BC                  NA           FALSE               1                8
10 ID                  ACTIVE       FALSE               3               86.7
# ... with 48 more rows
```

Your turn

Confer with your neighbours.

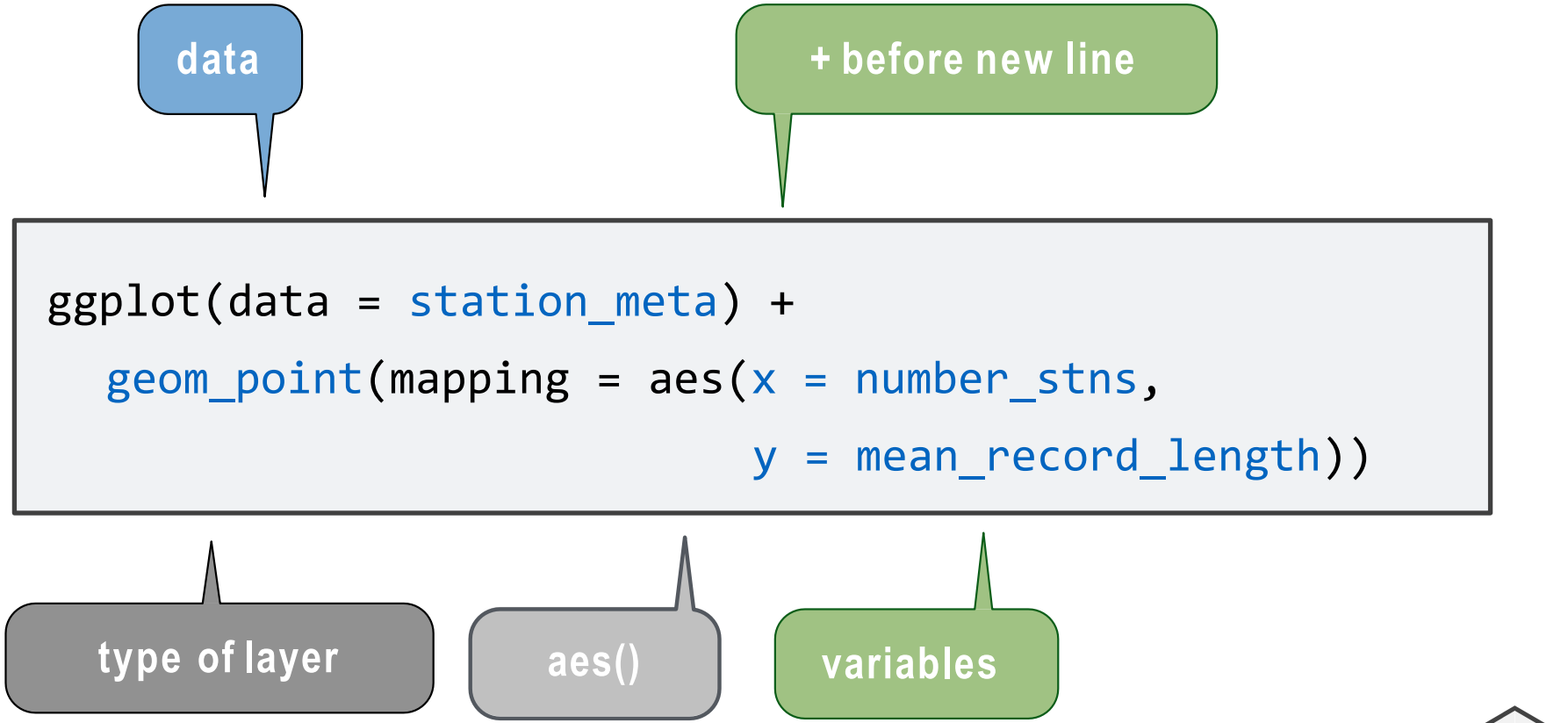
What relationships might interest you with this data? Let's plot it:

```
ggplot(data = station_meta, mapping = aes(x = number_stns, y = mean_record_length)) +
  geom_point()
```



1. "Initialize" a plotwith ggplot()
2. Add layers with geom_functions

```
ggplot(data = station_meta) +
  geom_point(mapping = aes(x = number_stns,
                           y = mean_record_length))
```



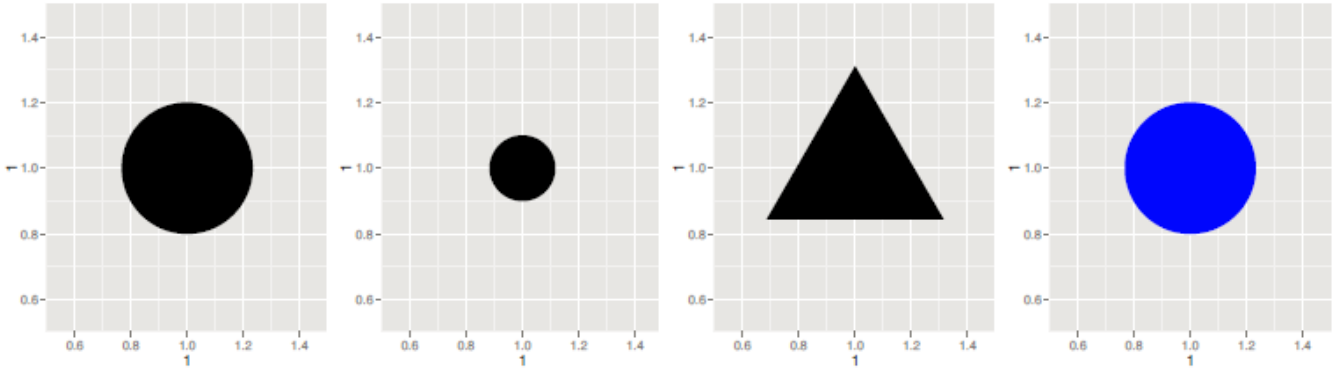
A Template

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

Mappings

Aesthetics

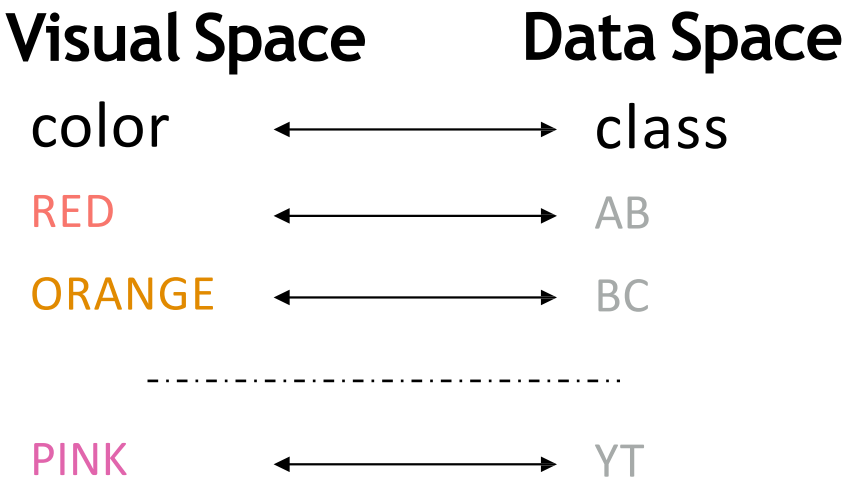
Visual properties of a geometric object



How do the appearance of these points vary?

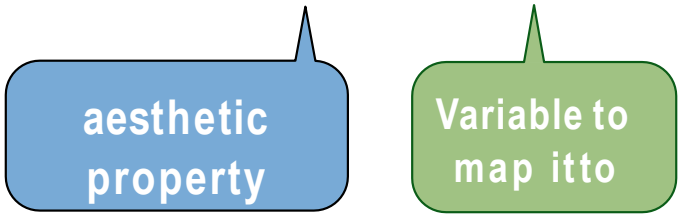
ggplot2

Mappings describe how aesthetics should relate to variables in the data.



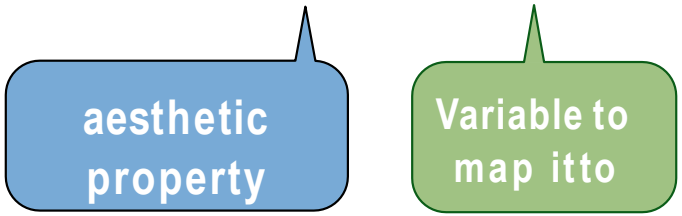
Aesthetic

```
ggplot(data = station_meta) +
  geom_point(mapping = aes(x = number_stns,
                           y = mean_record_length,
                           colour = PROV_TERR_STATE_LOC))
```



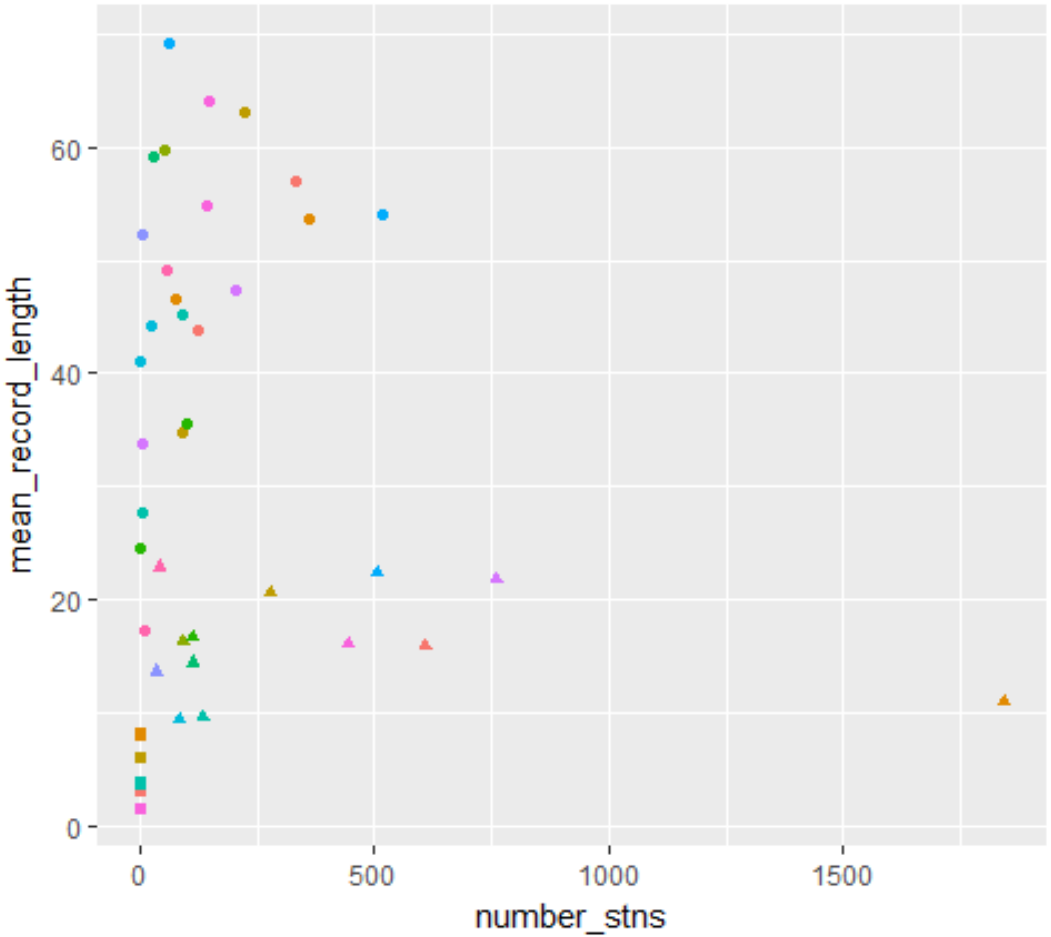
Aesthetic

```
ggplot(data = station_meta) +
  geom_point(mapping = aes(x = number_stns,
                           y = mean_record_length,
                           shape = HYD_STATUS))
```



Aesthetic

```
ggplot(data = station_meta) +
  geom_point(mapping = aes(x = number_stns,
                           y = mean_record_length,
                           colour = PROV_TERR_STATE_LOC,
                           shape = HYD_STATUS))
```



ACTIVE
DISCONTINUED
NA

PROV_TERR_STATE_LOC
AB
BC
MB
NB
NL
NS
NT
NU
ON
PE
QC
SK
YT

Legends added automatically

ggplot2

Your turn

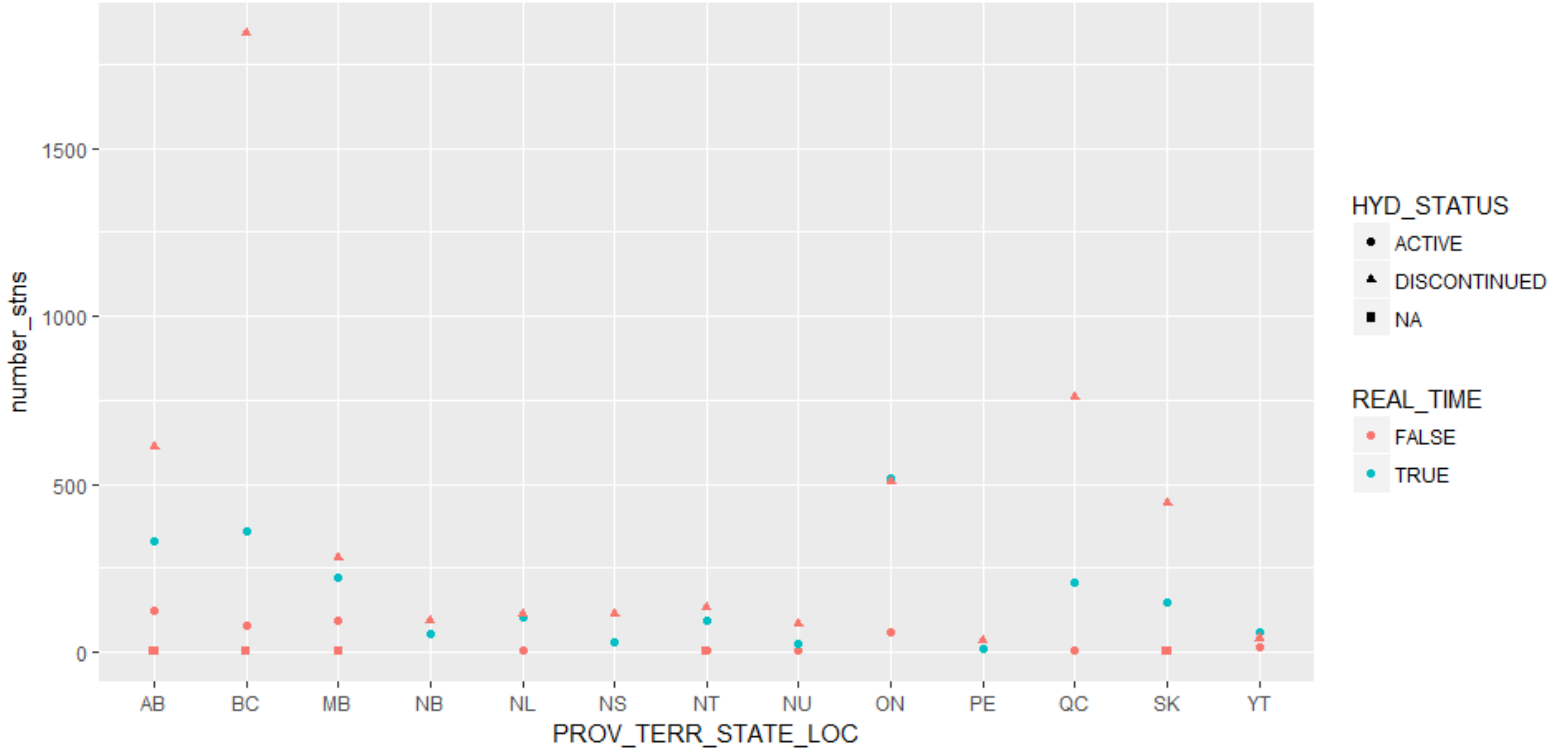
In the next chunk, modify which data is mapped to x, y, color, size, alpha, and shape aesthetics to your graph. Experiment.

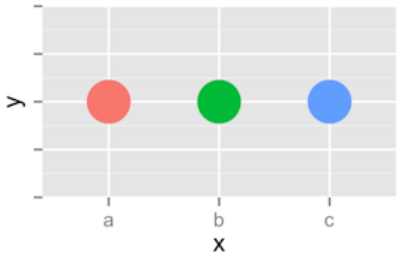
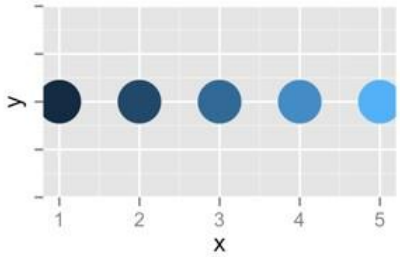
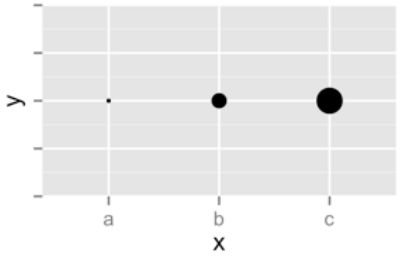
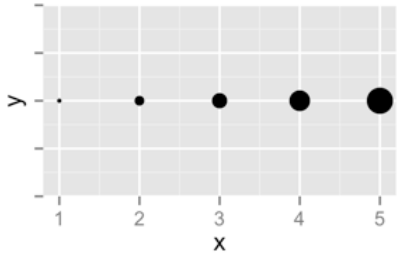
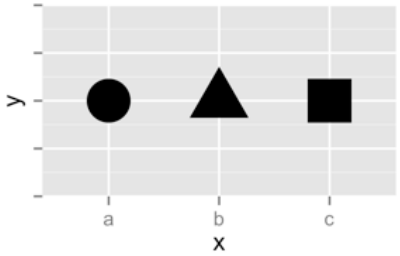
Generate two different examples?

Experiments

```
ggplot(data = station_meta, mapping = aes(x = PROV_TERR_STATE_LOC,
                                          y = number_stns,
                                          shape = HYD_STATUS,
                                          colour = REAL_TIME)) +
  geom_point()
```

Experiments

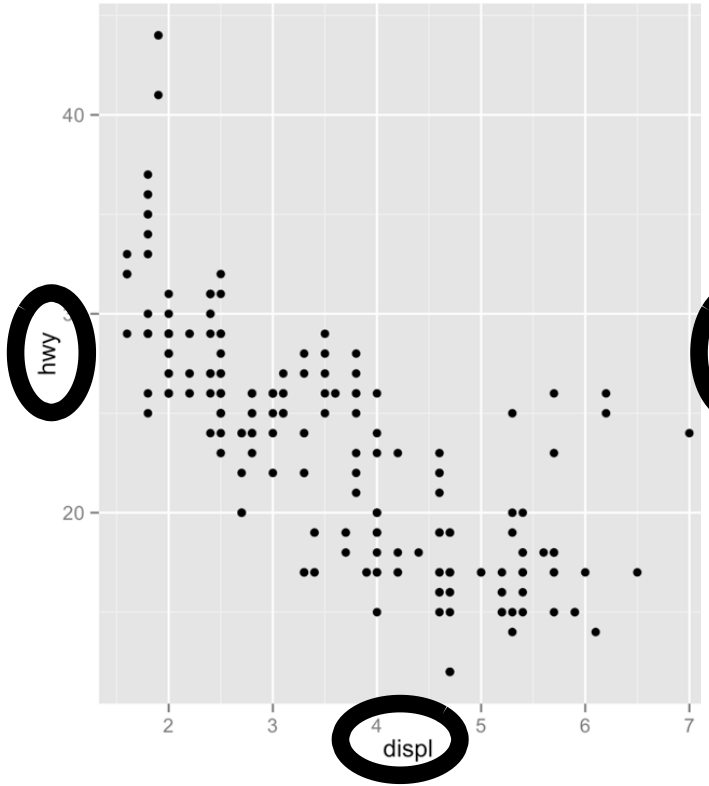


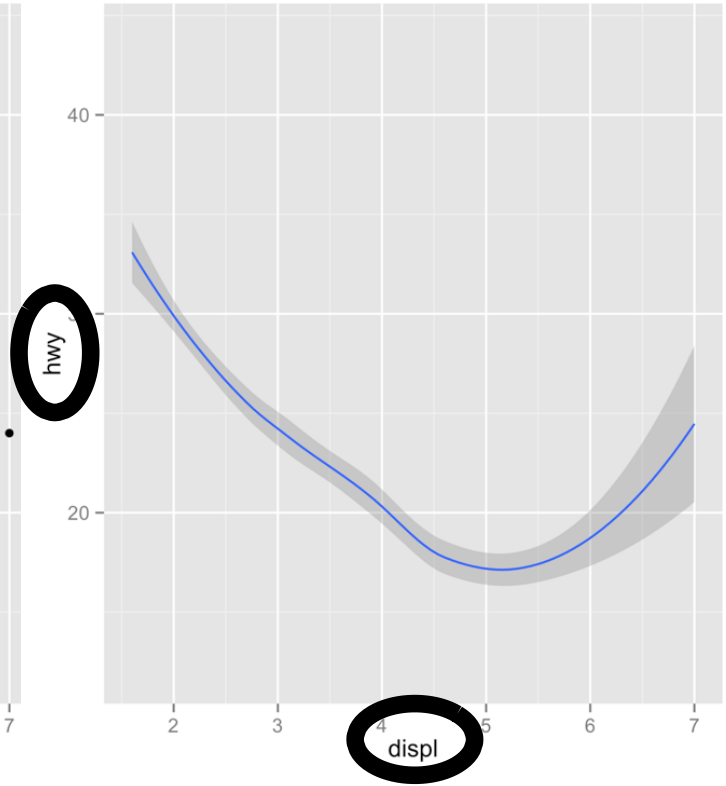
	Discrete	Continuous
Color		
Size		
Shape		

Geoms

How are these plots similar?

Same X and Y

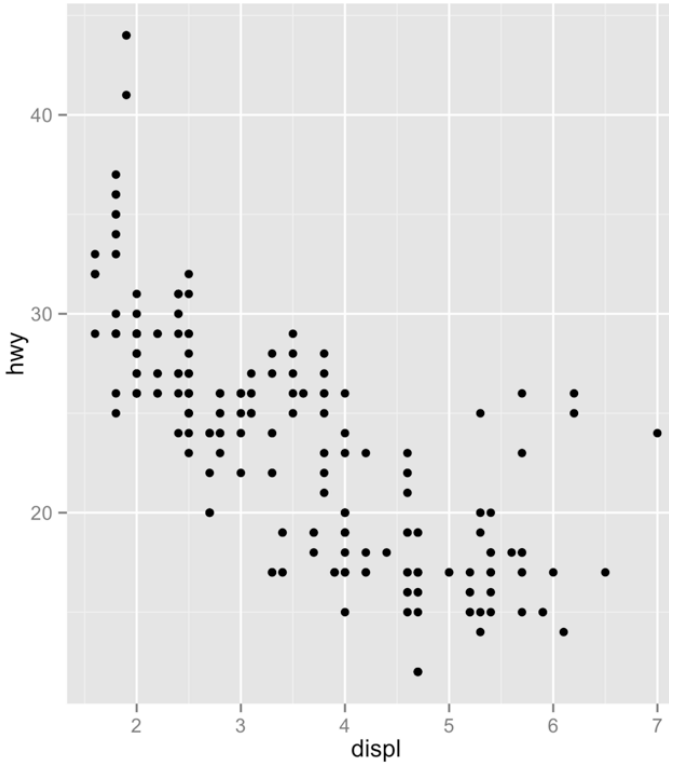


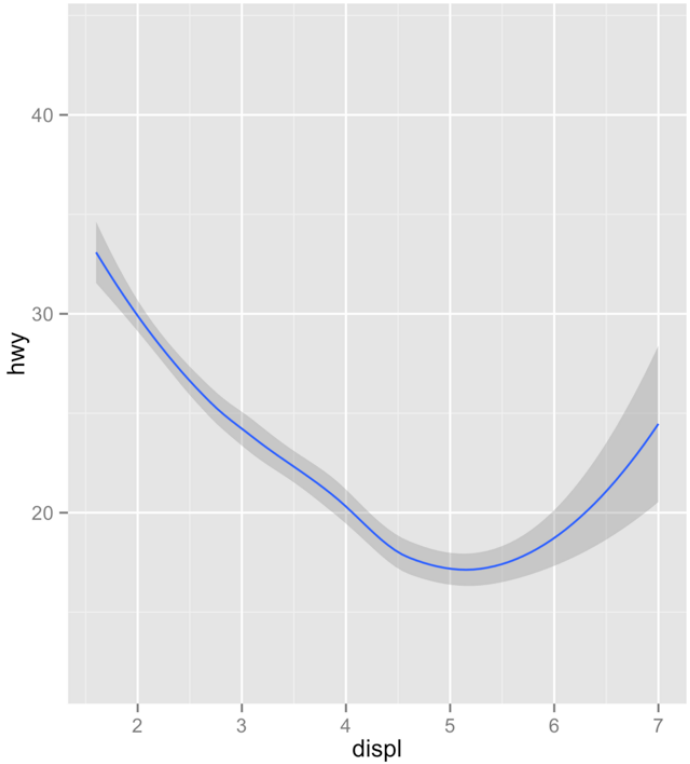


ggplot2

How are these plots different?

Different: geometric object (geom)
e.g. the visual object used to represent the data





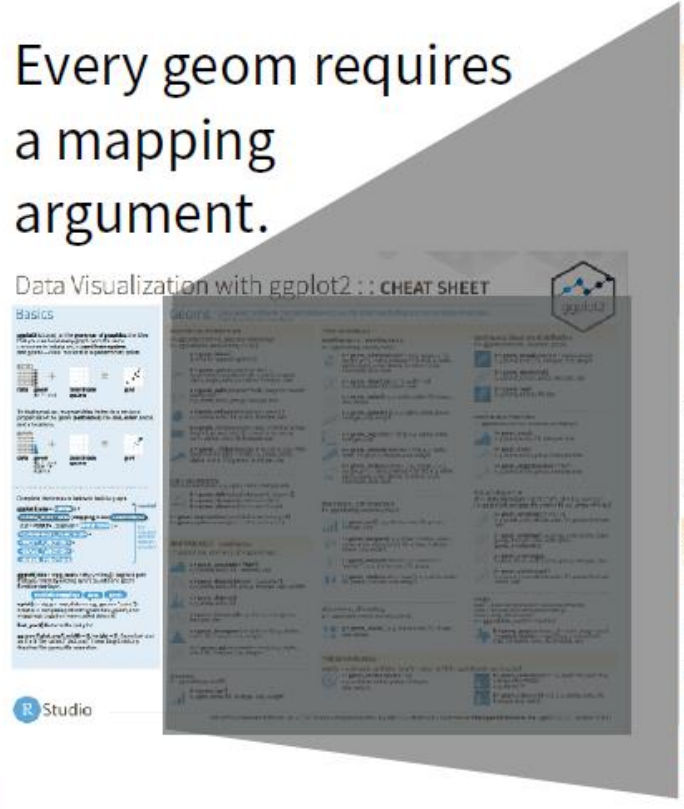
ggplot2

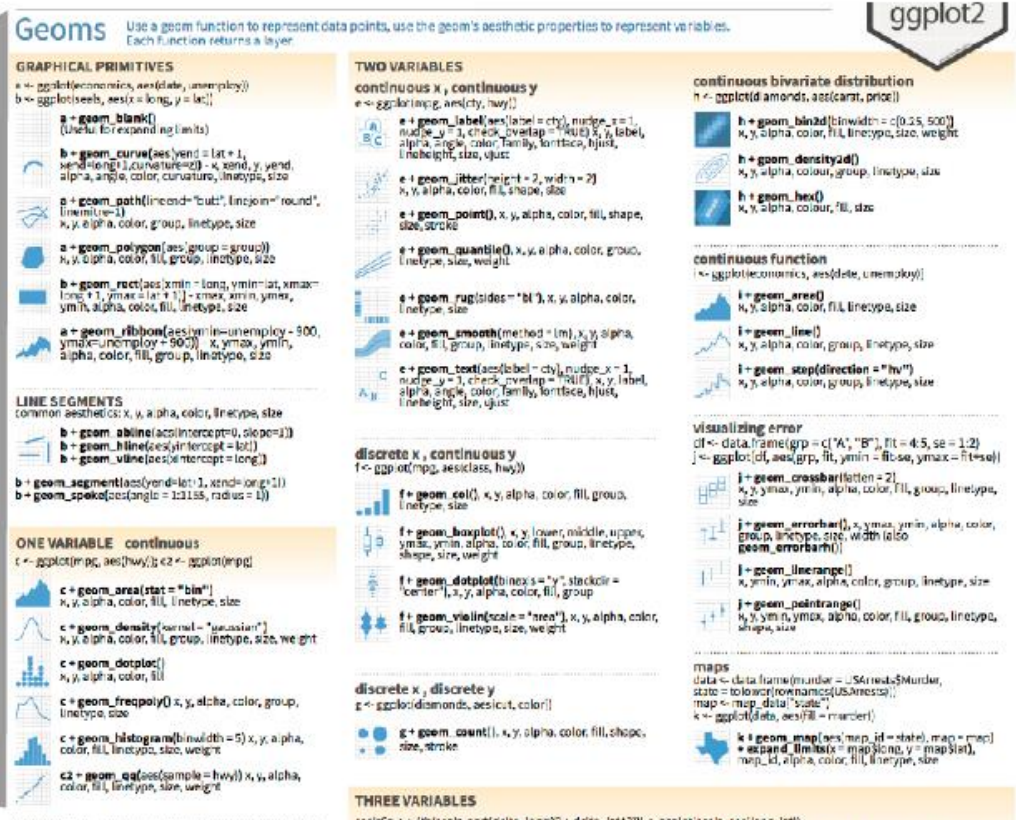
geoms

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

geom_functions

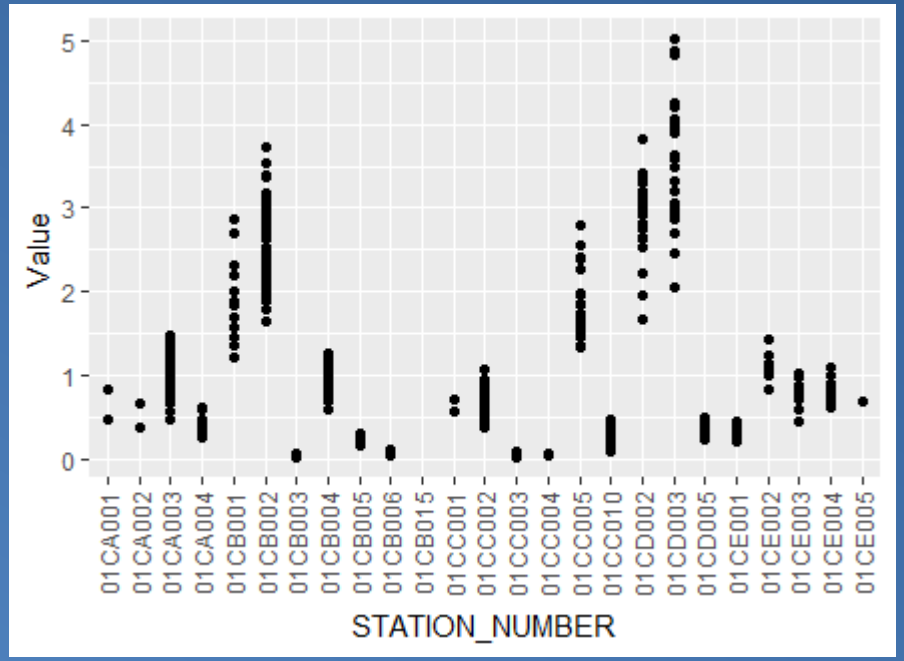
Every geom requires a mapping argument.

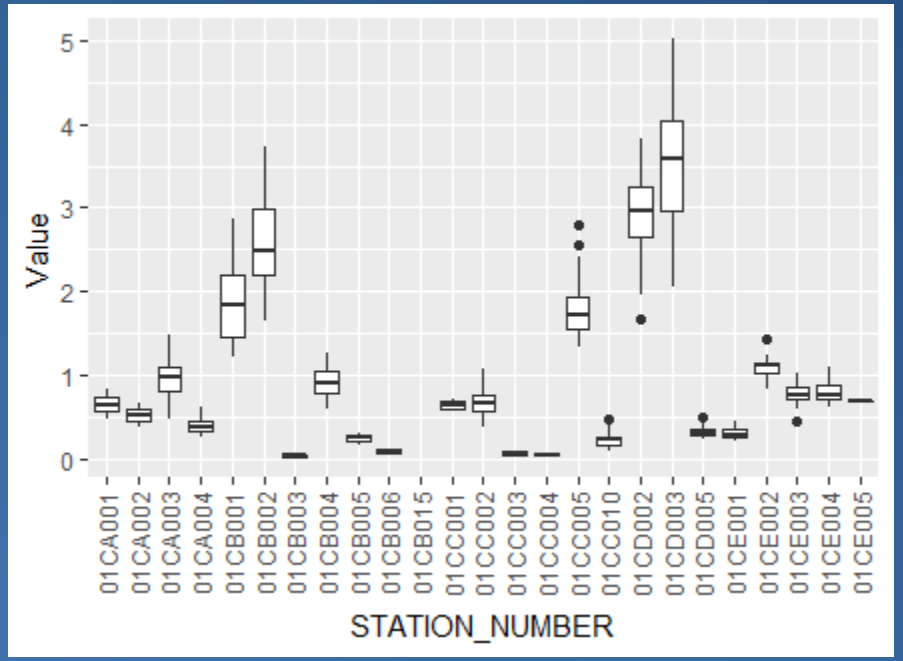


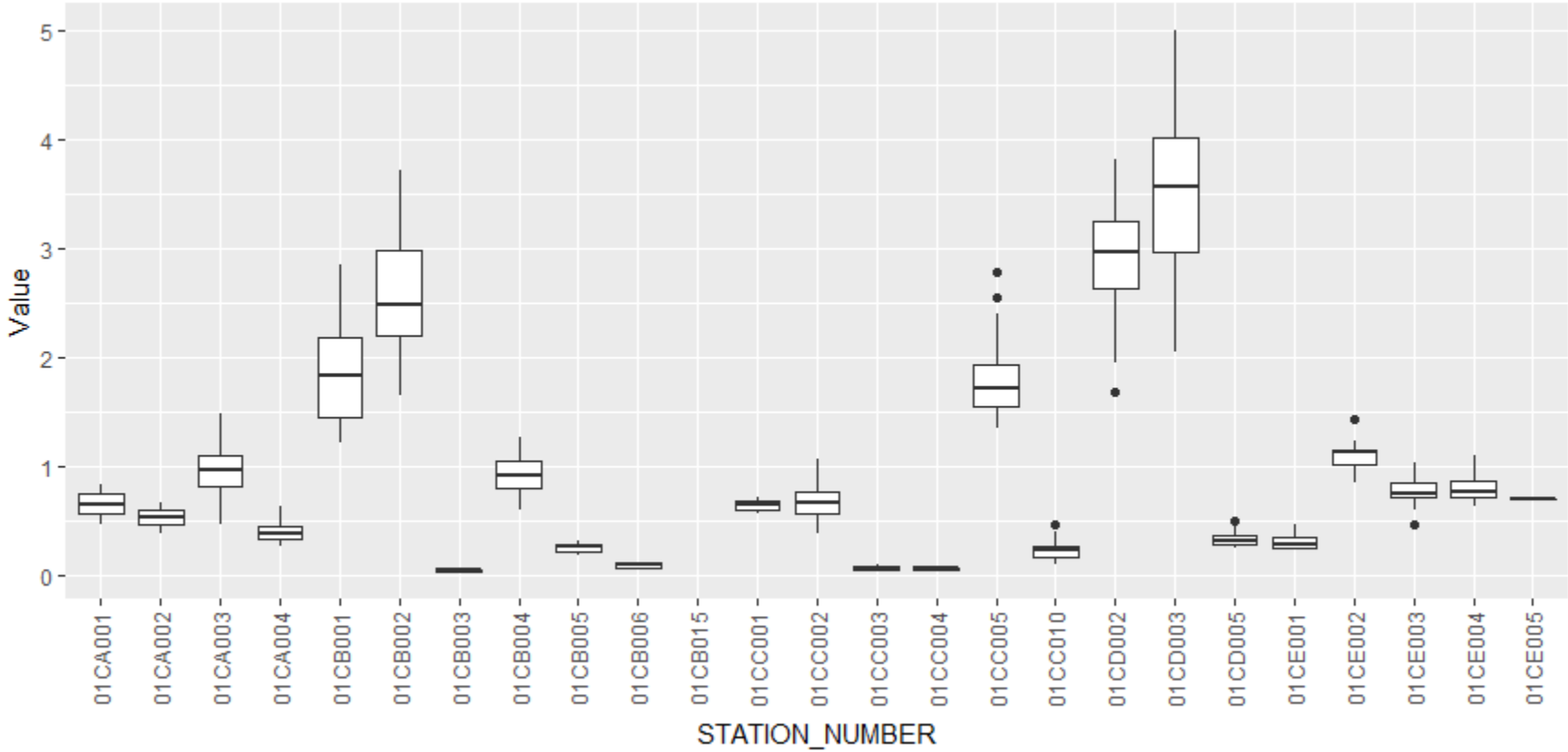


Your turn

With your neighbour, discuss how to convert the plot of left to the plot on the right. (Use the cheatsheet)





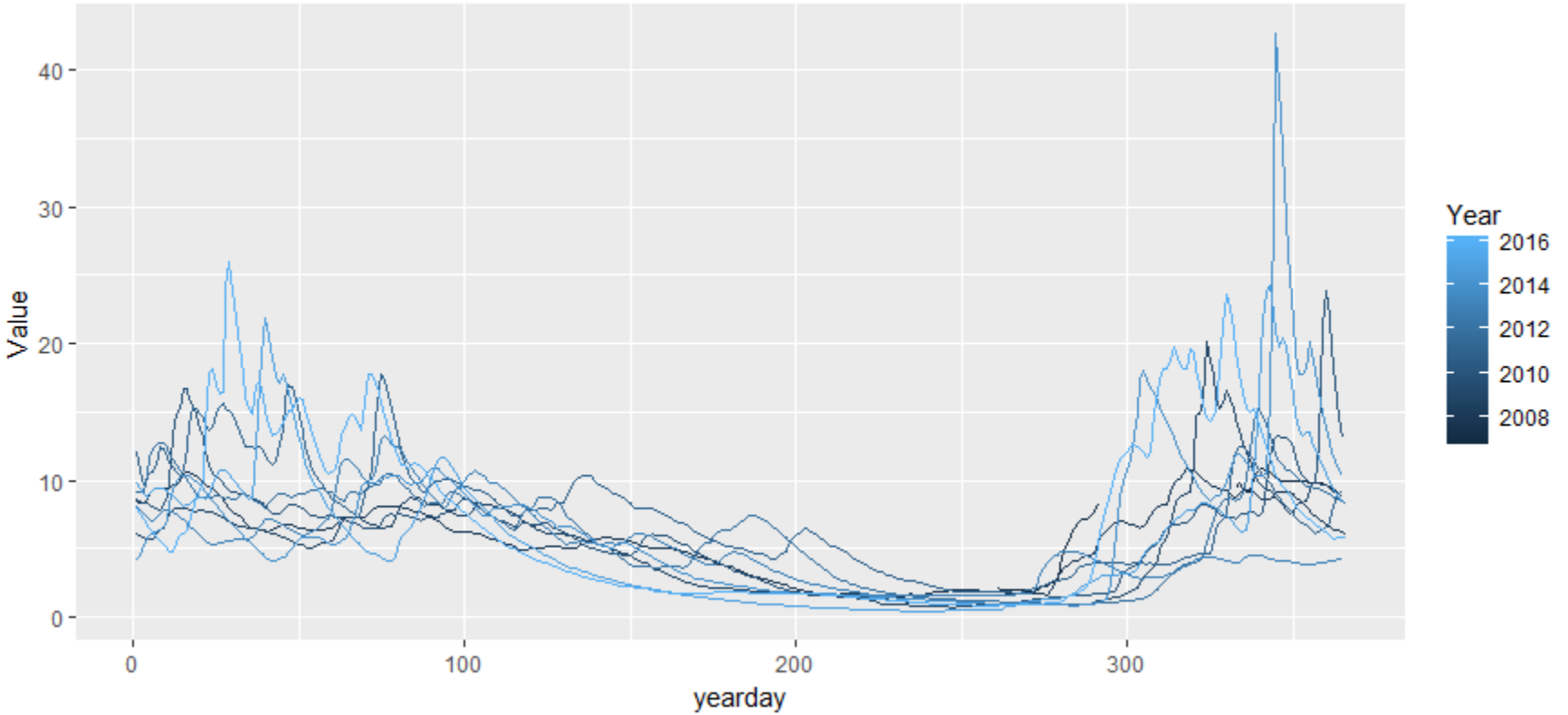



```
ggplot(data = pei_mean) +
  geom_boxplot(mapping = aes(x = STATION_NUMBER, y = Value)) +
  theme(axis.text.x  = element_text(angle=90, vjust=0.5))
```

Your turn

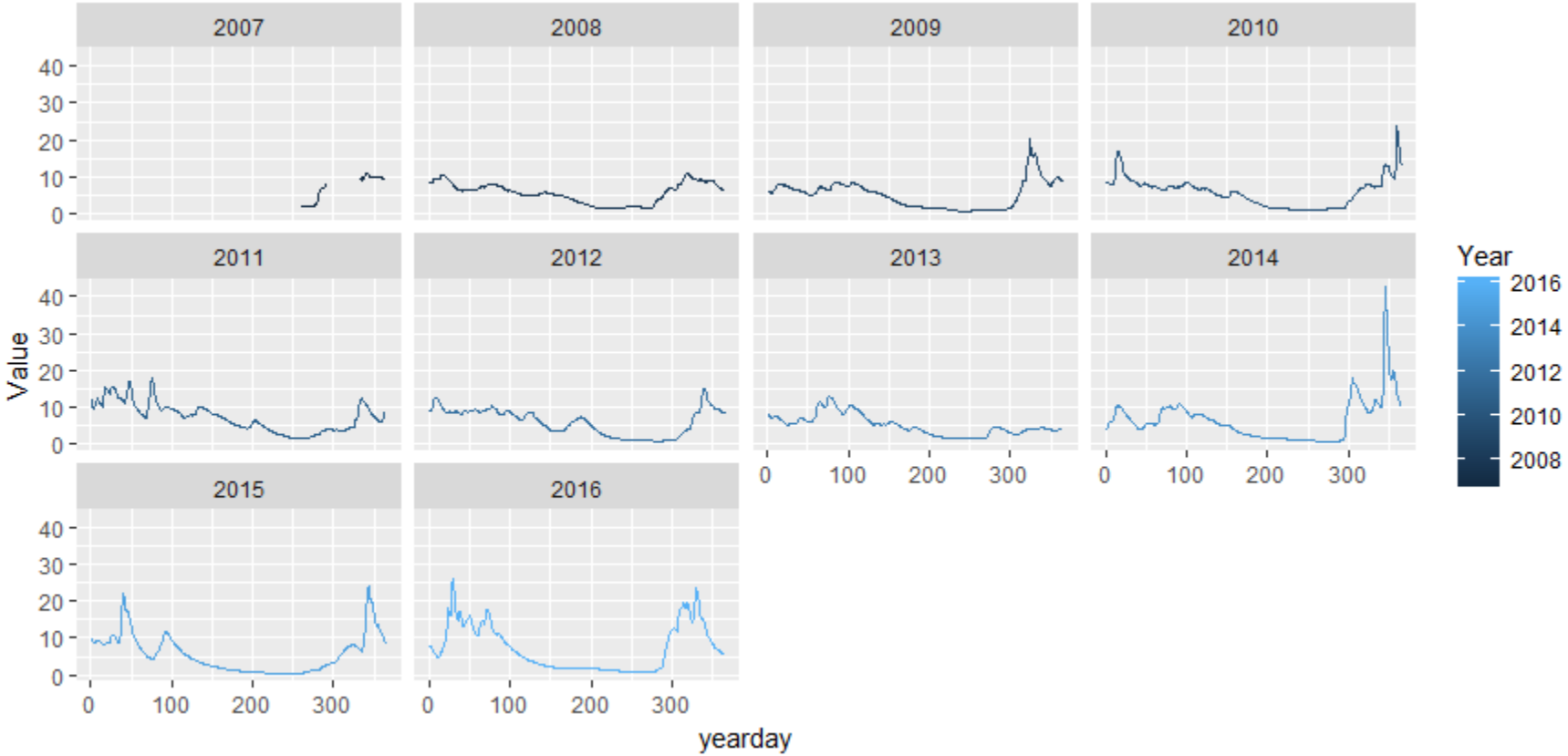
Make a hydrograph of the 08GB014 HORSESHOE RIVER ABOVE LOIS LAKE BC station data for each year using the Year and yearday columns created.

Hint checkout ?yday for an explanation of what it does




```
ggplot(data = horseshoe, mapping = aes(x = yearday, y = Value,
colour = Year)) +
  geom_line(mapping = aes(group = Year))
```

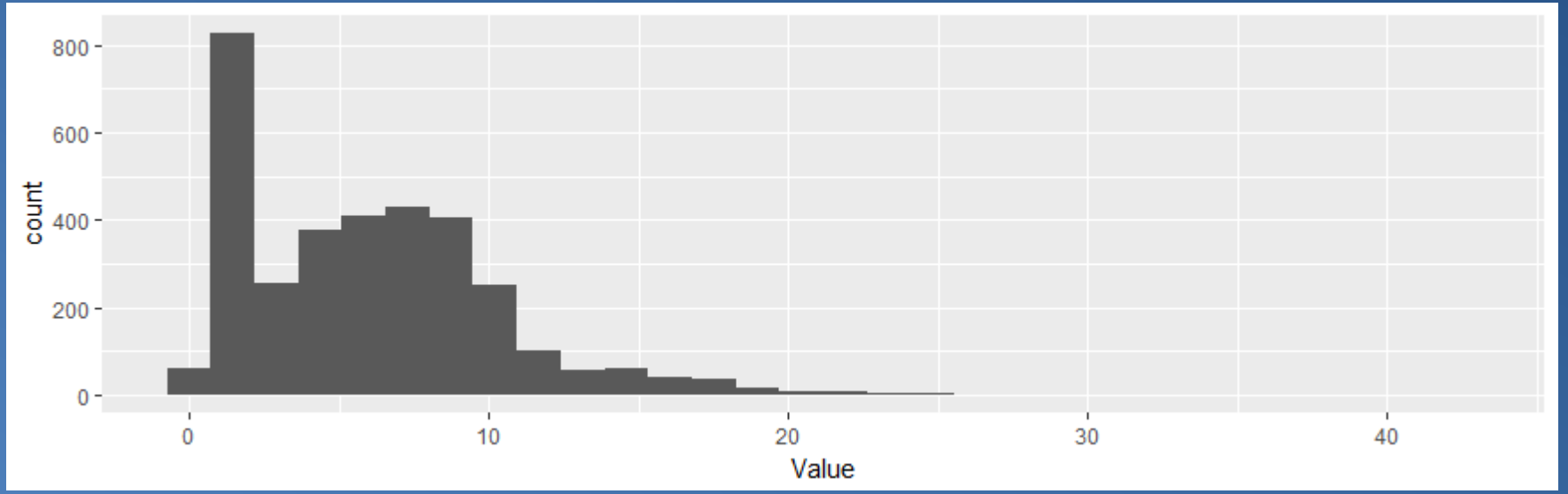

ggplot2

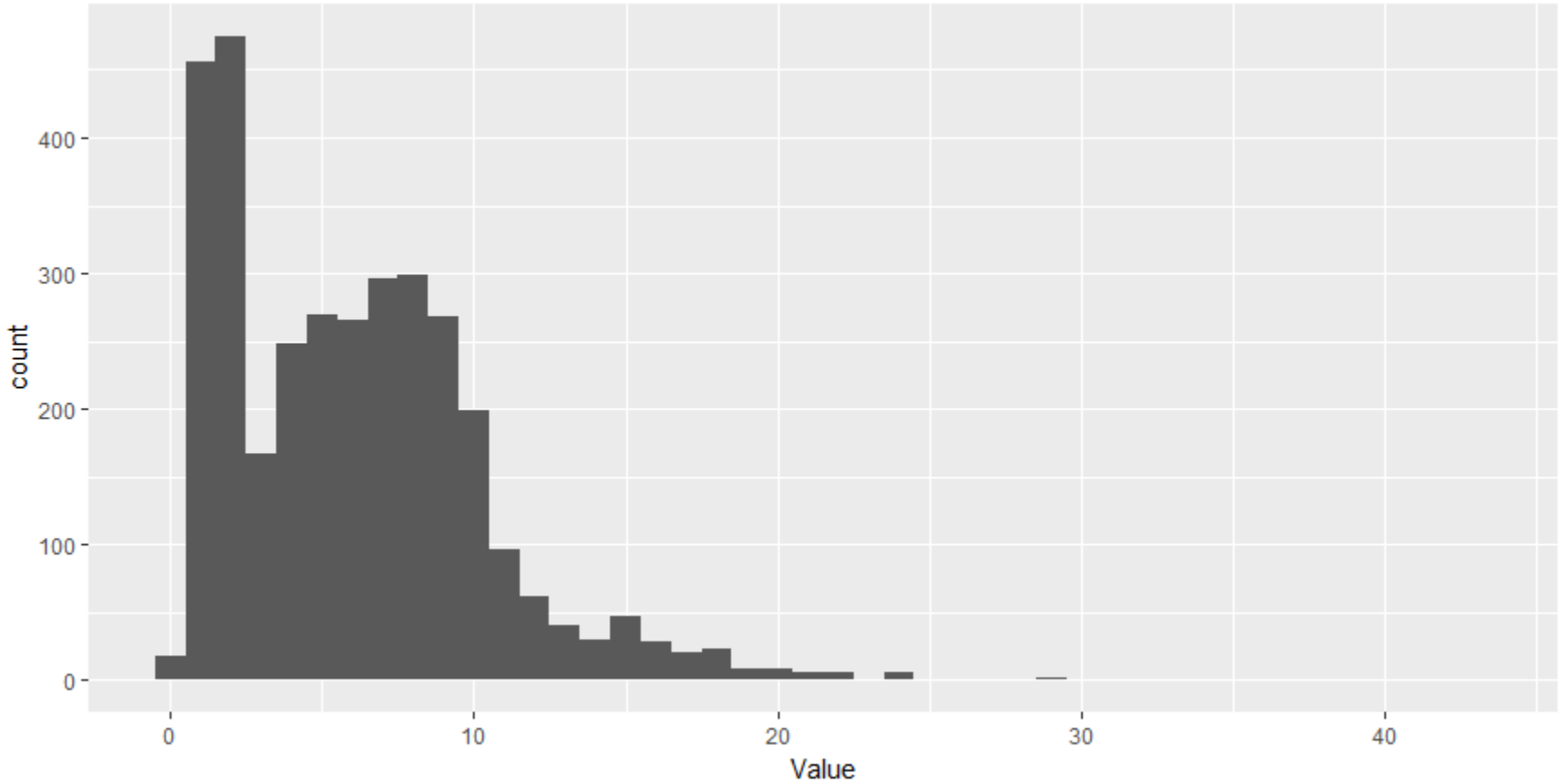



```
ggplot(data = horseshoe, mapping = aes(x = yearday, y = Value, colour = Year)) +
  geom_line(mapping = aes(group = Year)) +
  facet_wrap(~Year)
```

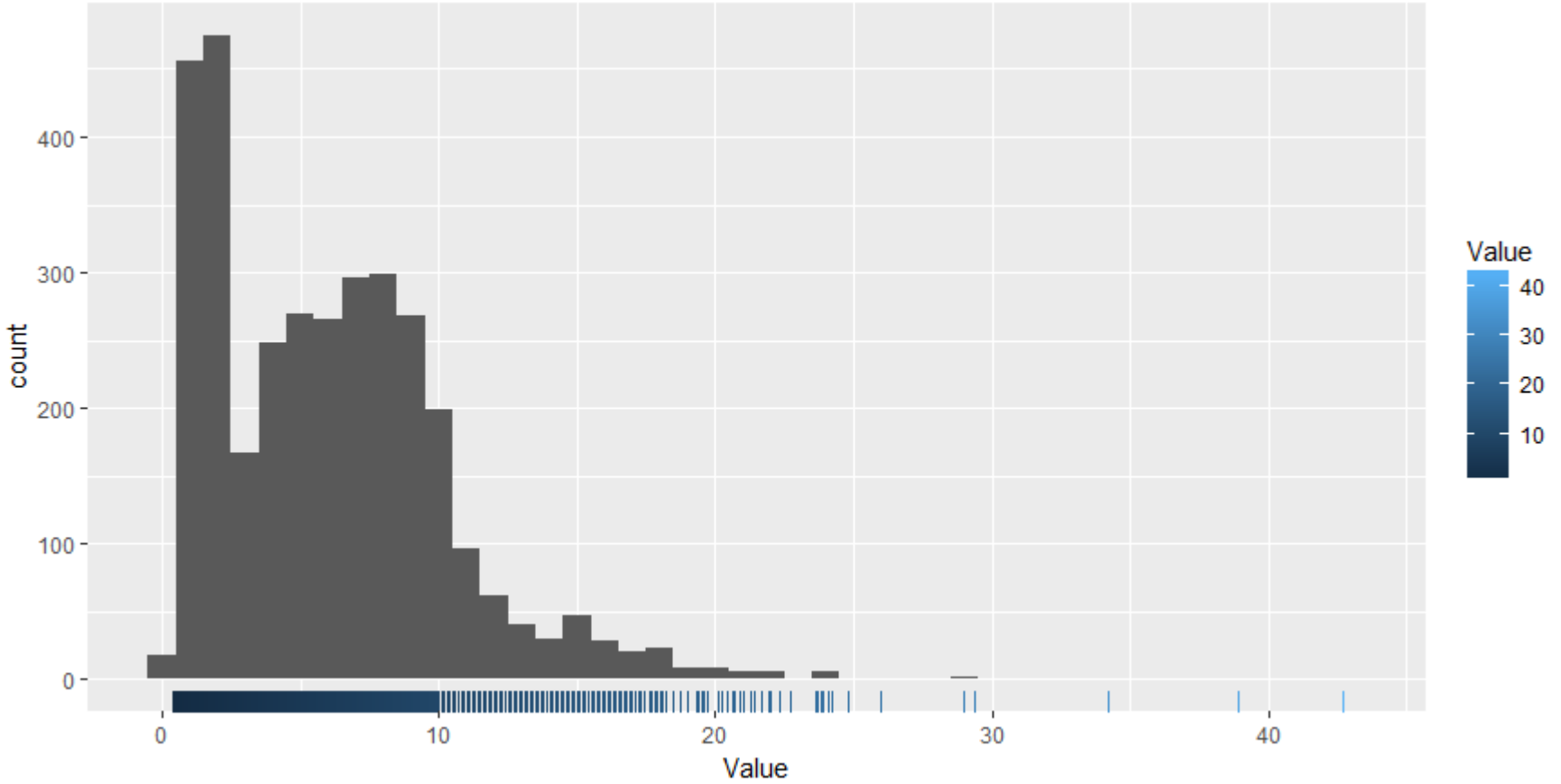
Your turn

With your partner, make the histogram of flow (i.e. Value) from the horseshoe data. Use the cheatsheet. Hint: do not supply a y variable.






```
ggplot(data = horseshoe, mapping = aes(x = Value)) +
  geom_histogram(binwidth = 1)
```

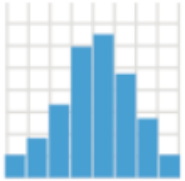



```
ggplot(data = horseshoe, mapping = aes(x = Value)) +
  geom_histogram(binwidth = 1) +
  geom_rug(mapping = aes(colour = Value))
```


ggplot2

On the cheatsheat:

Arguments inside (), are geom specific options



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

Outside the (), are aesthetics that can be mapped or set.

ggplot2

Road map

- Make histogram
- Move all the way dwon to the Grammar of Graphics part
- Then show the meta_stn data in all its glory

Saving graphs

Your turn

What does this command return?

```
getwd()
```

Working Directory

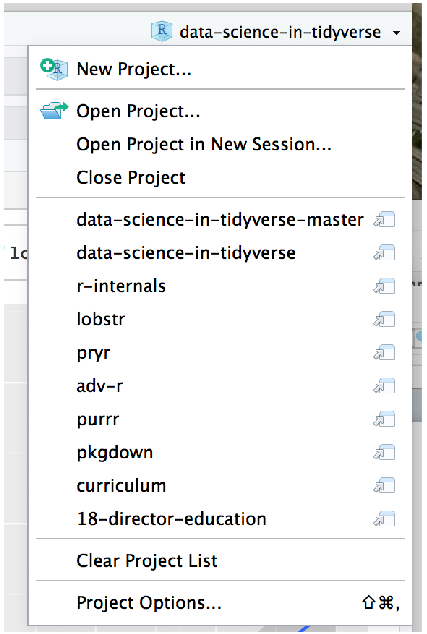
R associates itself with a folder (i.e. directory) on your computer.

- This folder is known as your "working directory"
- When you save files, R will save them here
- When you load files, R will look for them here

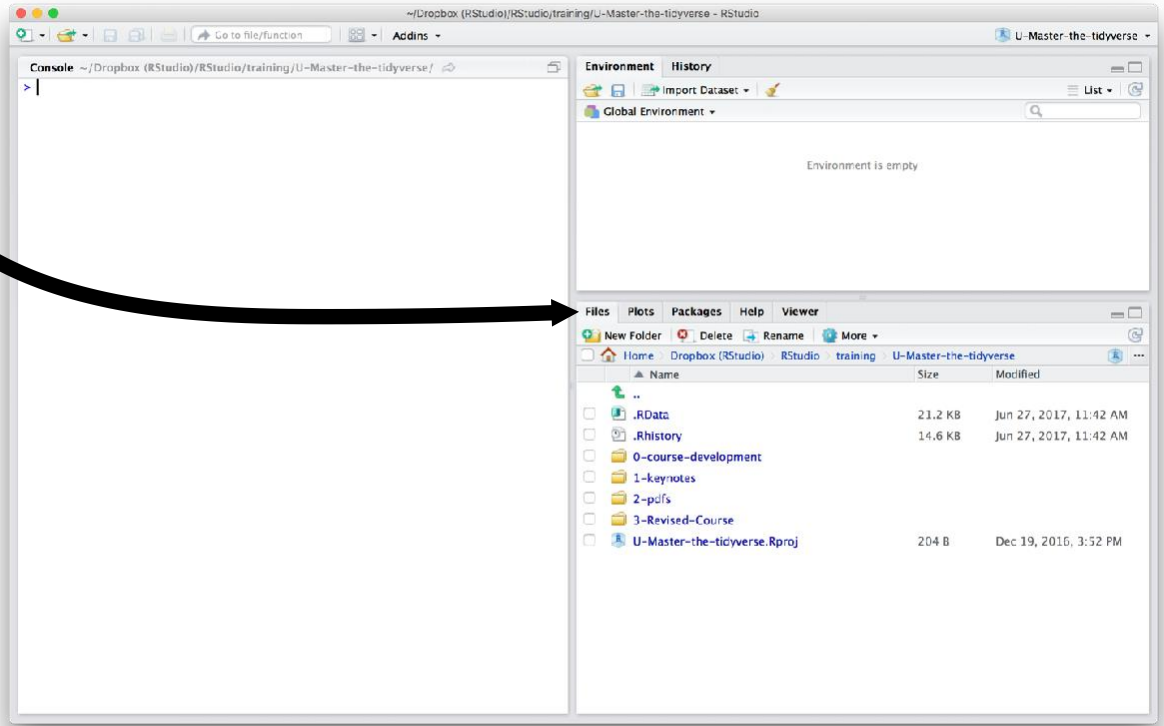
Projects

The best way of managing your working directory is with RStudio Projects.

One RStudio project = one real life project One RStudio project = one directory



The files pane of the IDE displays the contents of your working directory



Saving plots

ggsave() saves the last plot.

Uses size on screen:

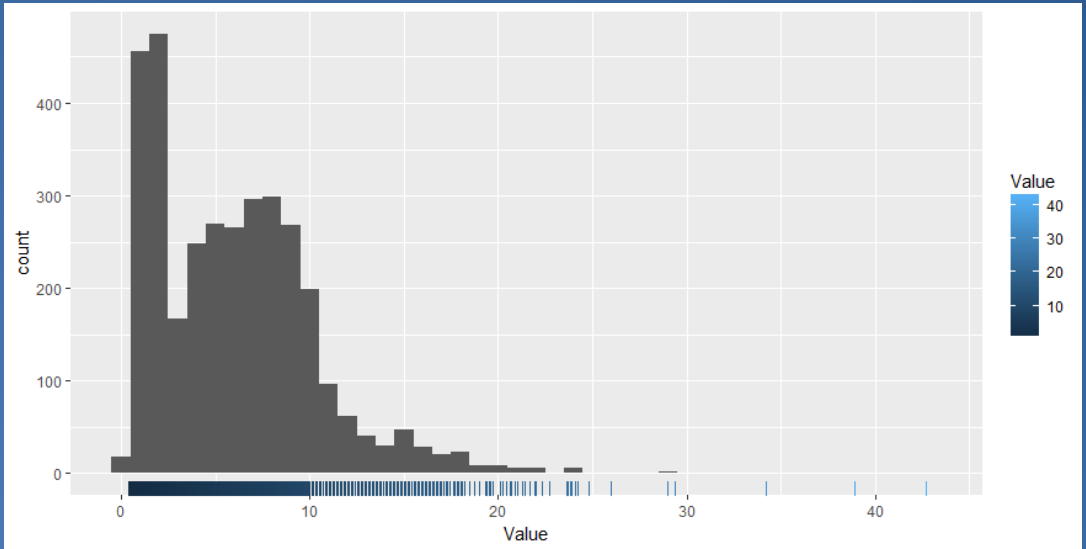
```
ggsave("my-plot.pdf")
ggsave("my-plot.png")
```

Specify size in inches

```
ggsave("my-plot.pdf", width = 6, height = 6)
```


Your turn

Save your last plot and then locate it in your files pane. (You may have to refresh the files list).



Grammar of Graphics

To make a graph

[template]

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

To make a graph

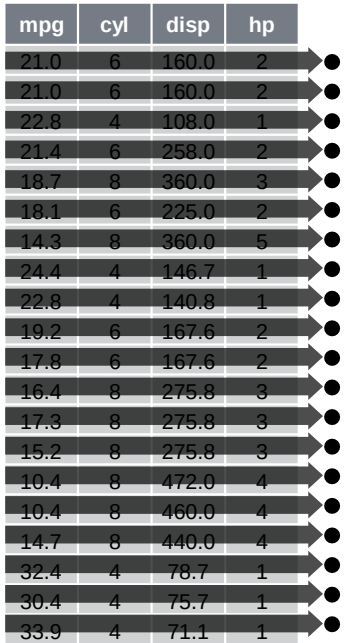
mpg	cyl	disp	hp
21.0	6	160.0	2
21.0	6	160.0	2
22.8	4	108.0	1
21.4	6	258.0	2
18.7	8	360.0	3
18.1	6	225.0	2
14.3	8	360.0	5
24.4	4	146.7	1
22.8	4	140.8	1
19.2	6	167.6	2
17.8	6	167.6	2
16.4	8	275.8	3
17.3	8	275.8	3
15.2	8	275.8	3
10.4	8	472.0	4
10.4	8	460.0	4
14.7	8	440.0	4
32.4	4	78.7	1
30.4	4	75.7	1
33.9	4	71.1	1

data

1. Pick a **data**set

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```


To make a graph



mpg	cyl	disp	hp
21.0	6	160.0	2
21.0	6	160.0	2
22.8	4	108.0	1
21.4	6	258.0	2
18.7	8	360.0	3
18.1	6	225.0	2
14.3	8	360.0	5
24.4	4	146.7	1
22.8	4	140.8	1
19.2	6	167.6	2
17.8	6	167.6	2
16.4	8	275.8	3
17.3	8	275.8	3
15.2	8	275.8	3
10.4	8	472.0	4
10.4	8	460.0	4
14.7	8	440.0	4
32.4	4	78.7	1
30.4	4	75.7	1
33.9	4	71.1	1

data **geom**

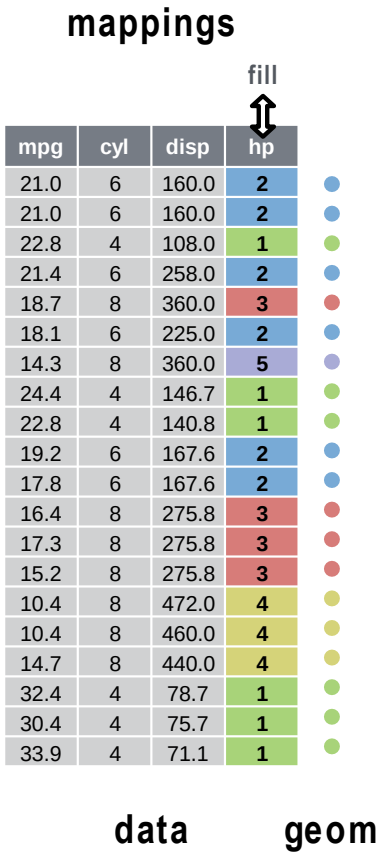
1. Pick a **data**set

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

2. Choose a **geom** to display cases

ggplot2

To make a graph



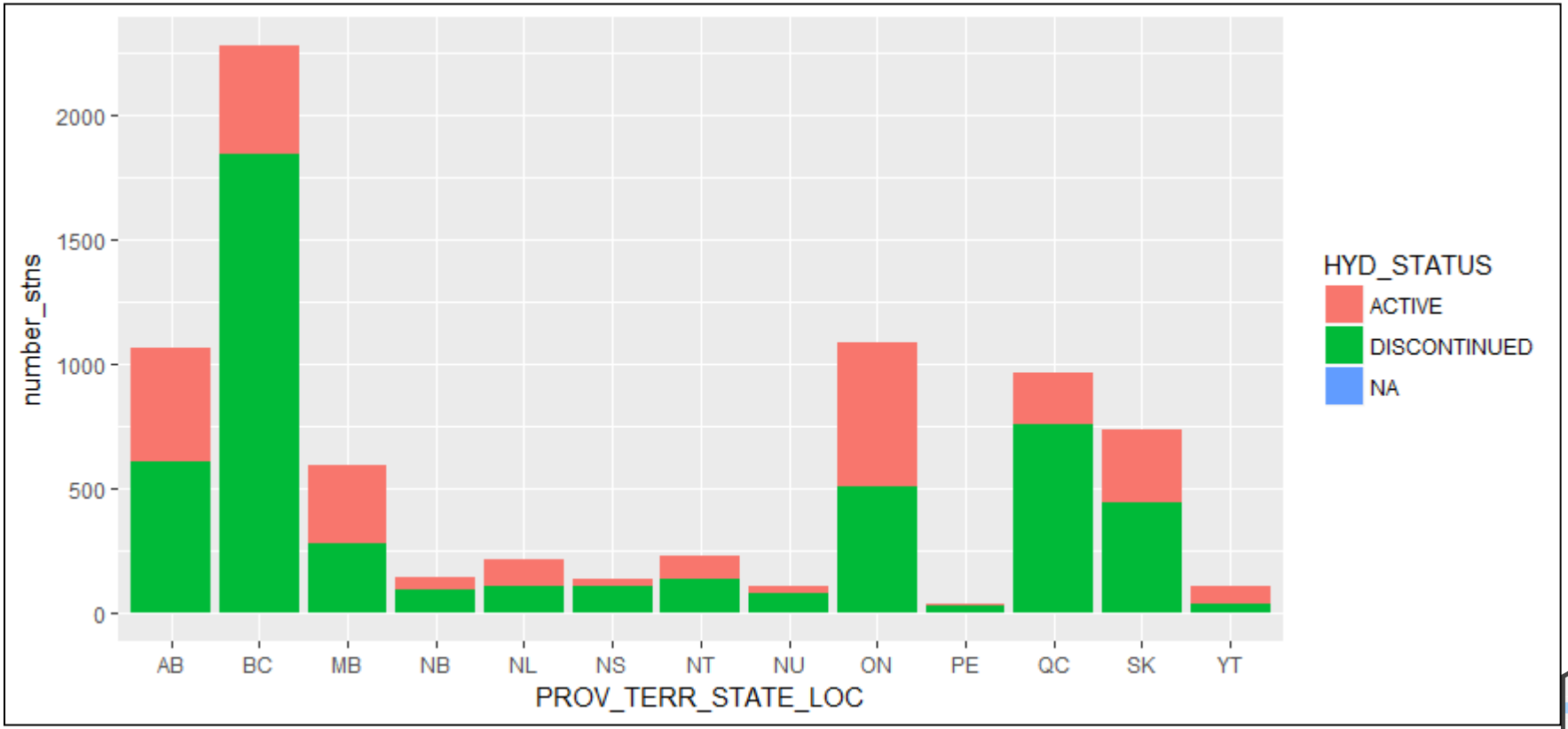
1. Pick a **data**set

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

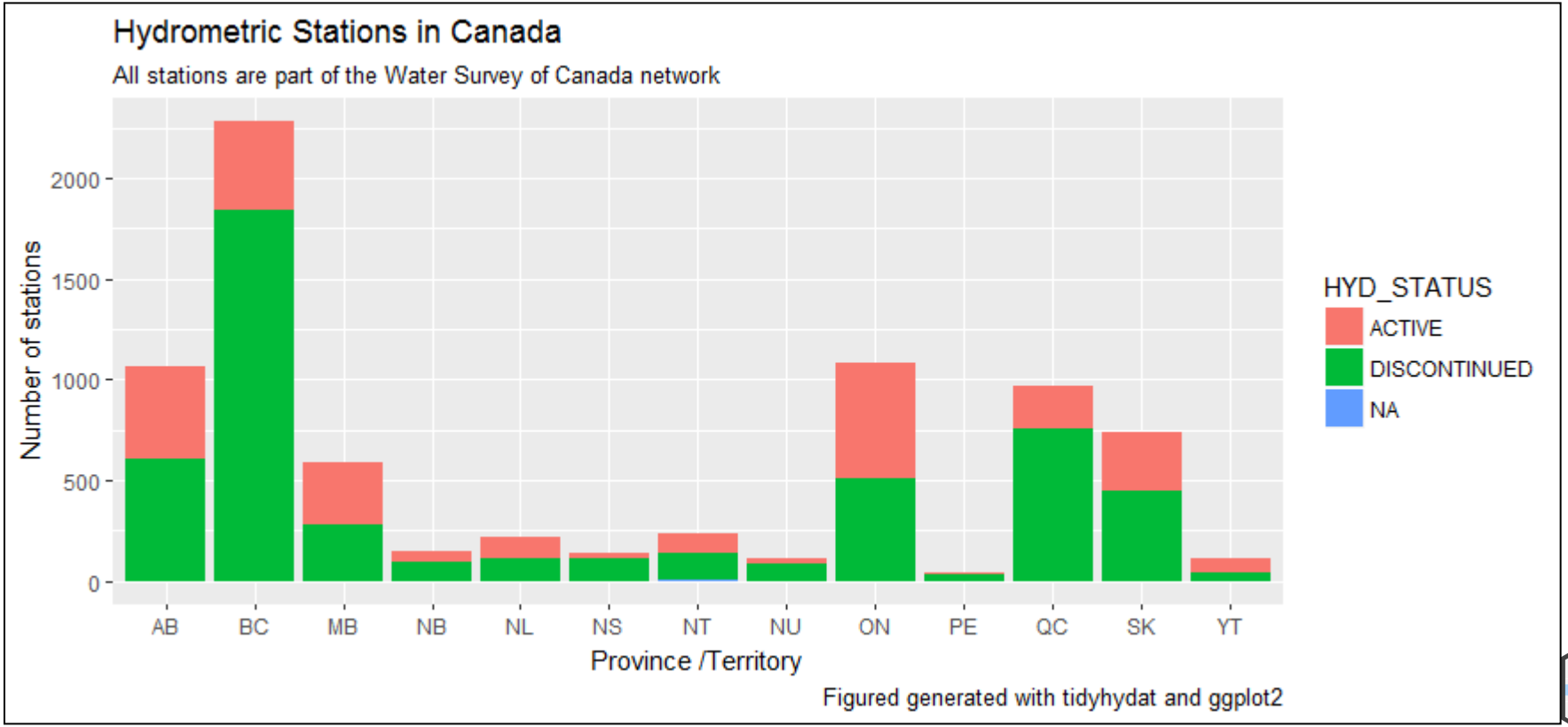
2. Choose a **geom** to display cases

3. **Map** aesthetic properties to variables

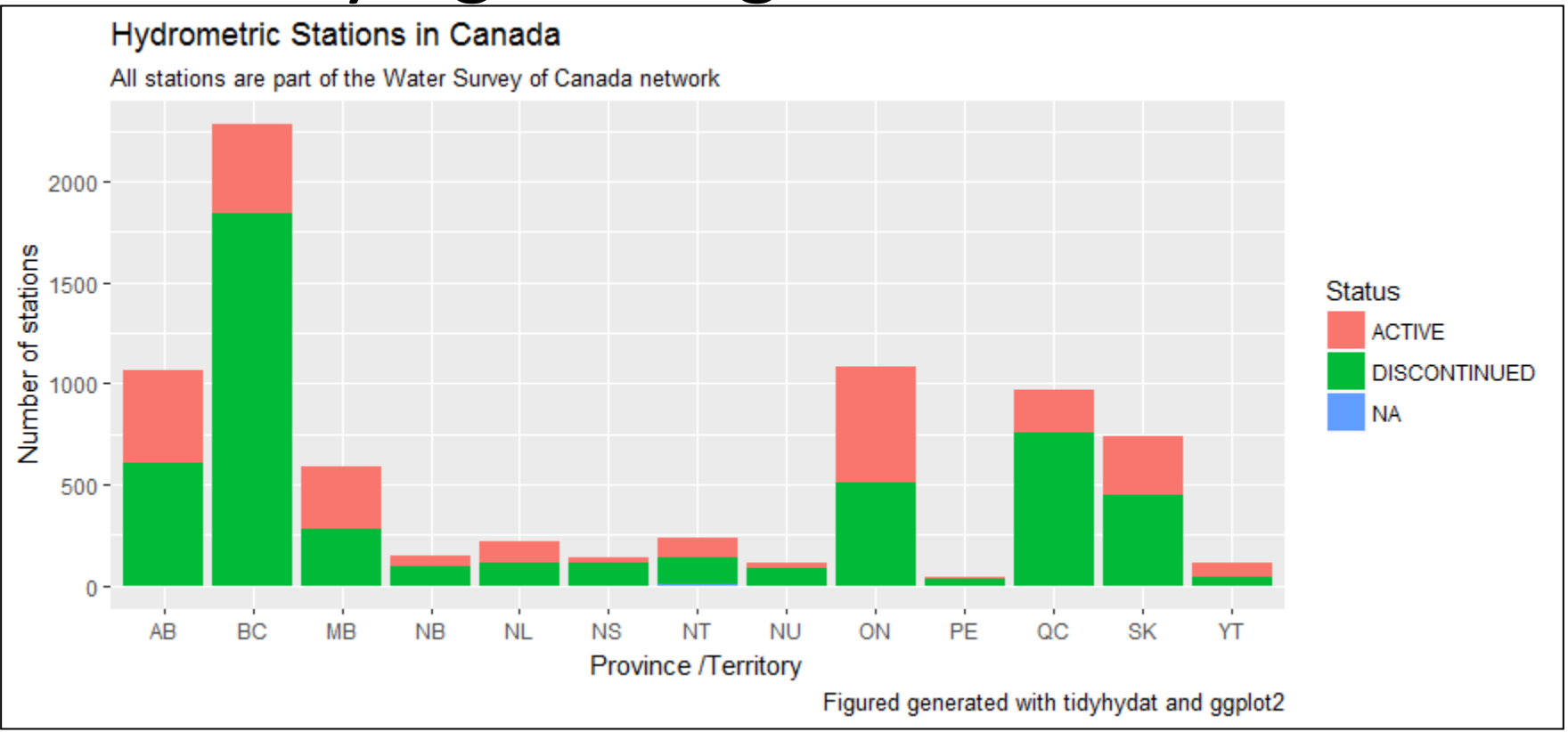
What else?



Titles and captions+ labs()

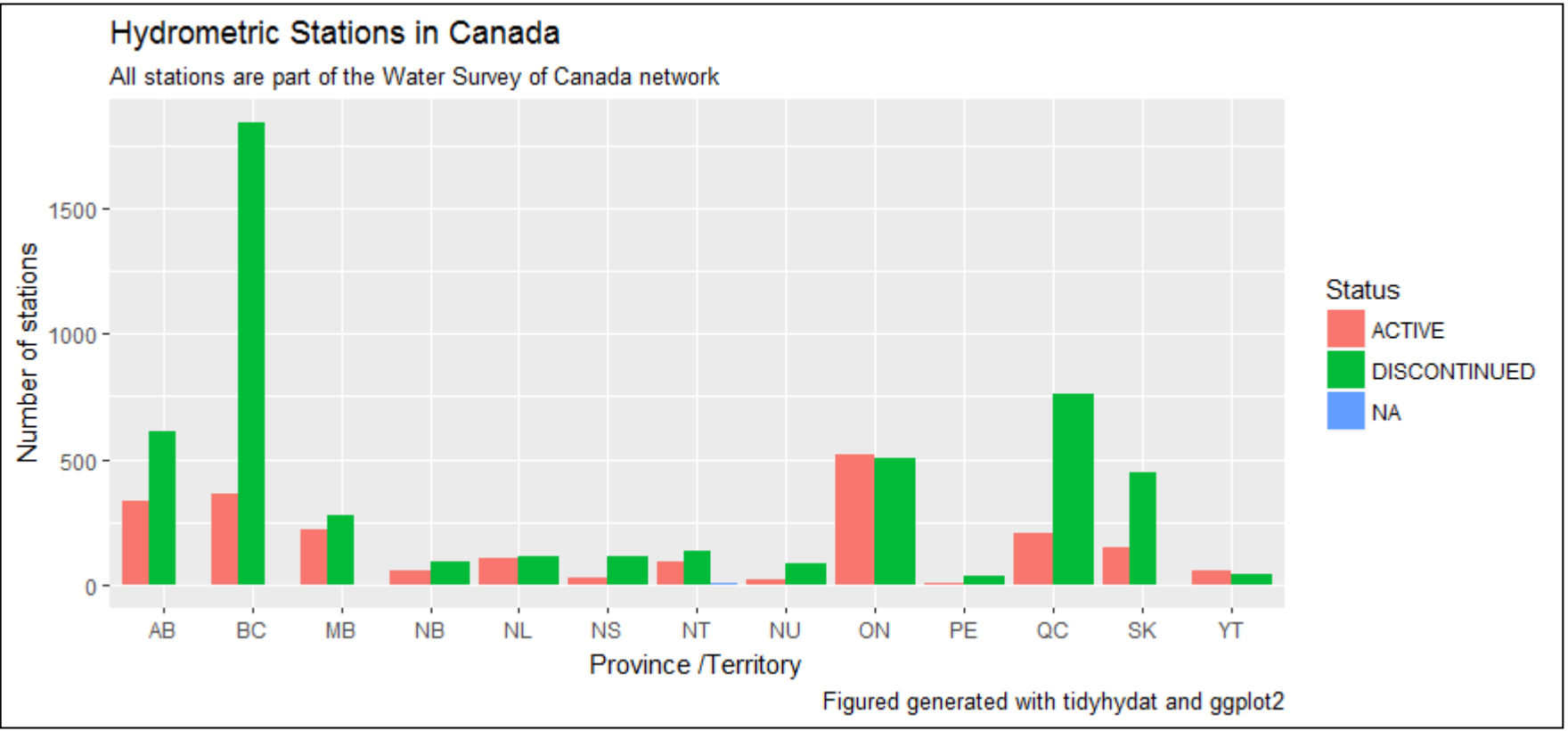


Modifying the legend title

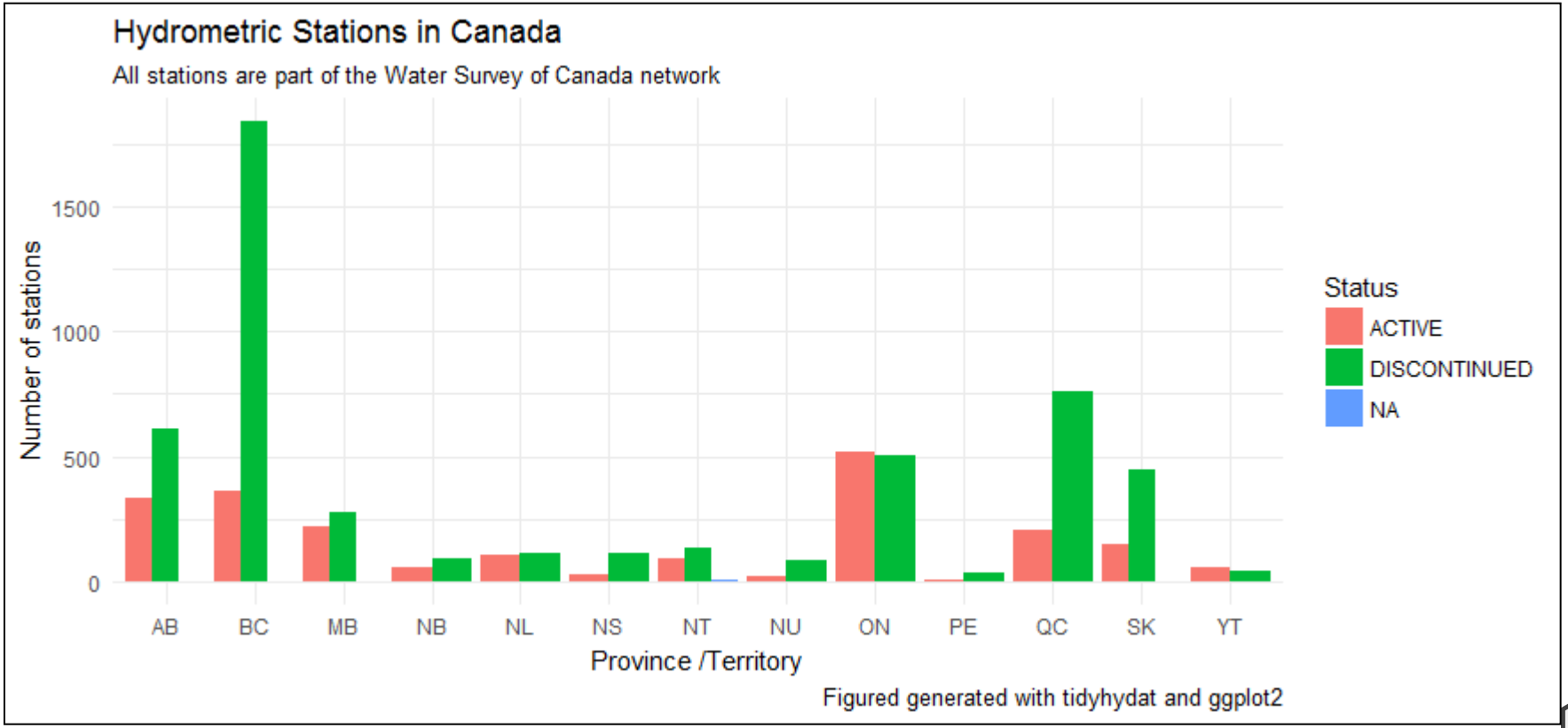


Position Adjustments

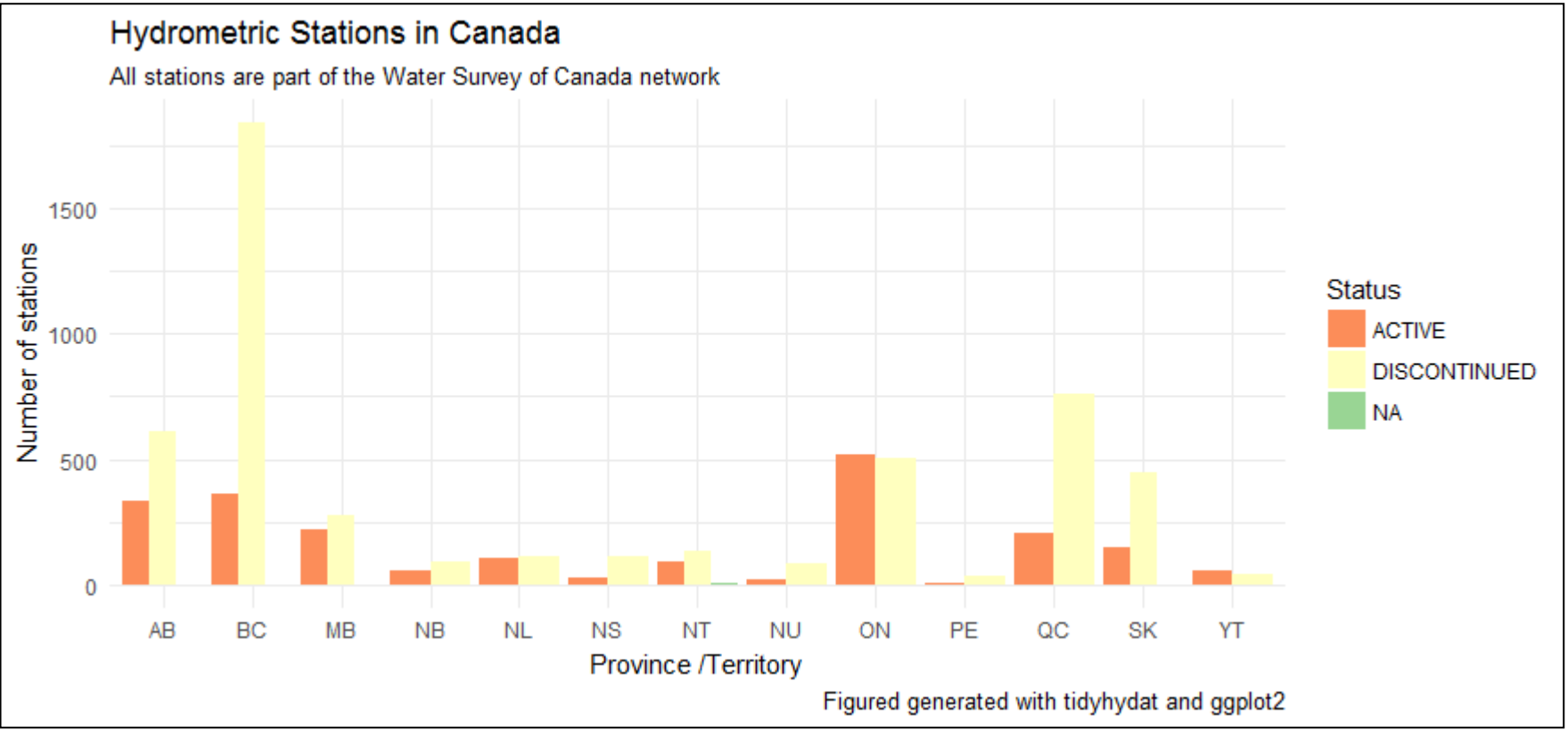
How overlapping objects are arranged



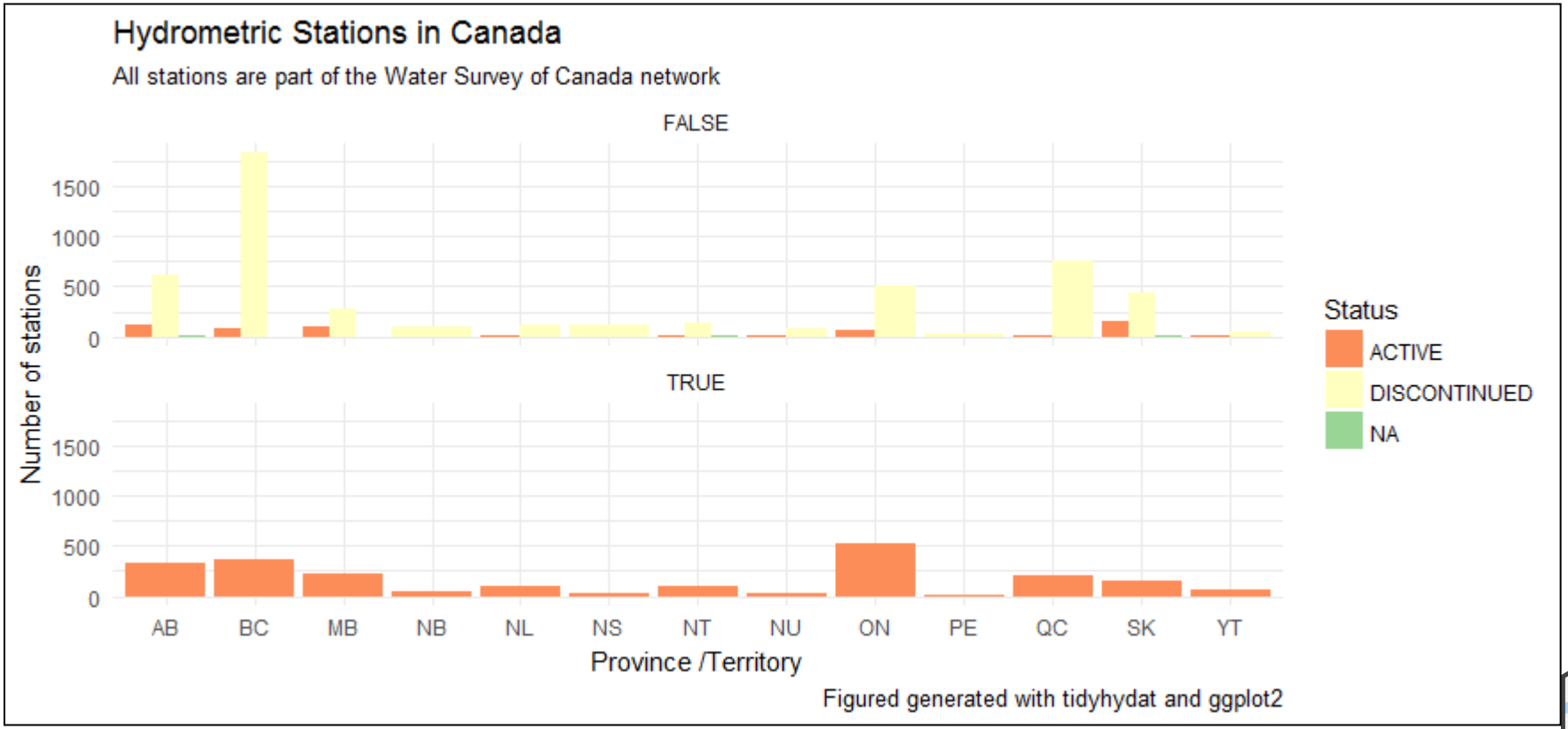
Theme - Visual appearance of non-data elements



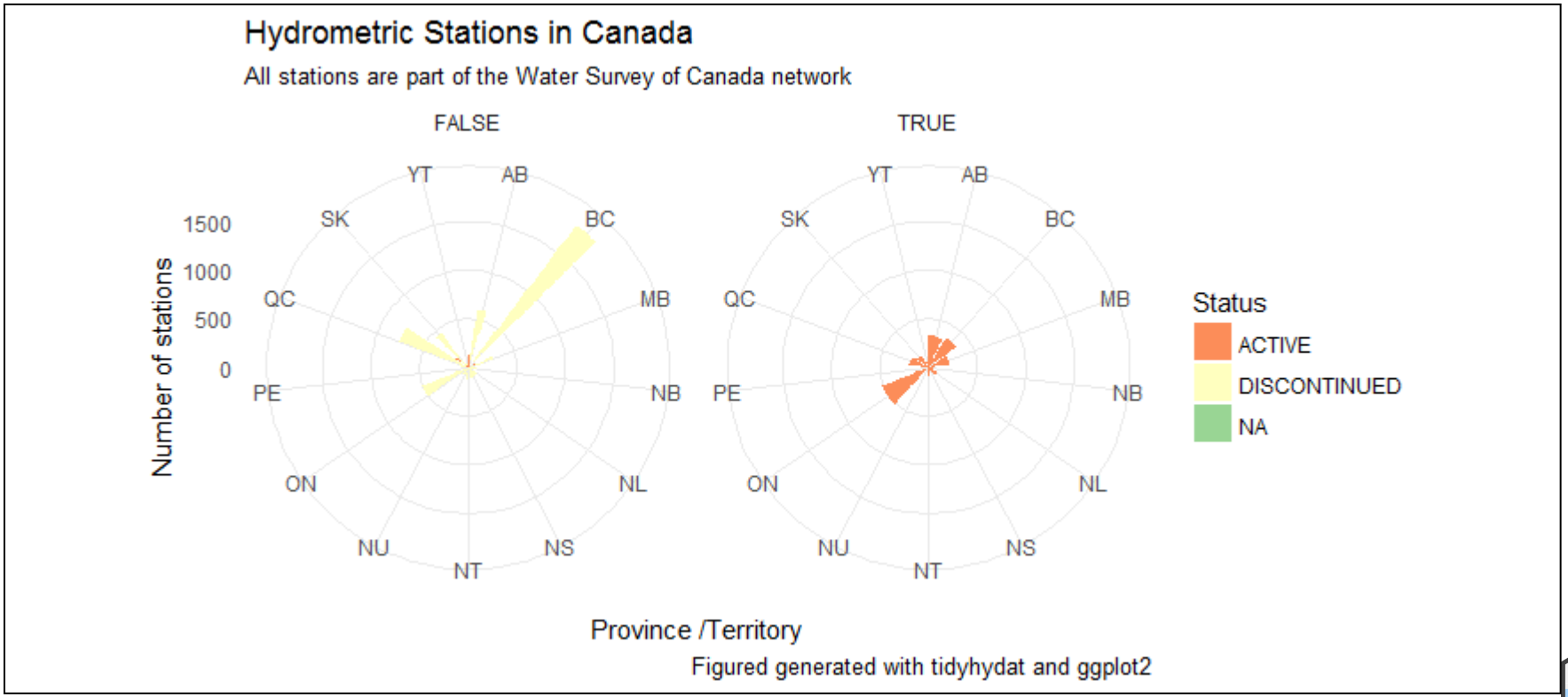
Scales - Customize color scales, other mappings



Facets - Subplots that display subsets of the data.



Coordinate systems



A ggplot2 template

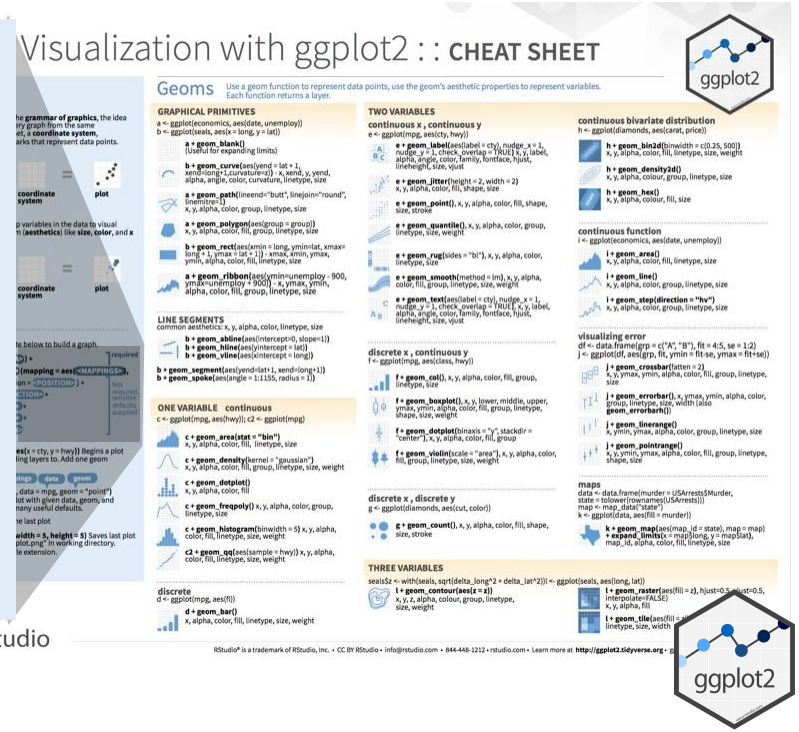
Make any plot by filling in the parameters of this template

Complete the template below to build a graph.

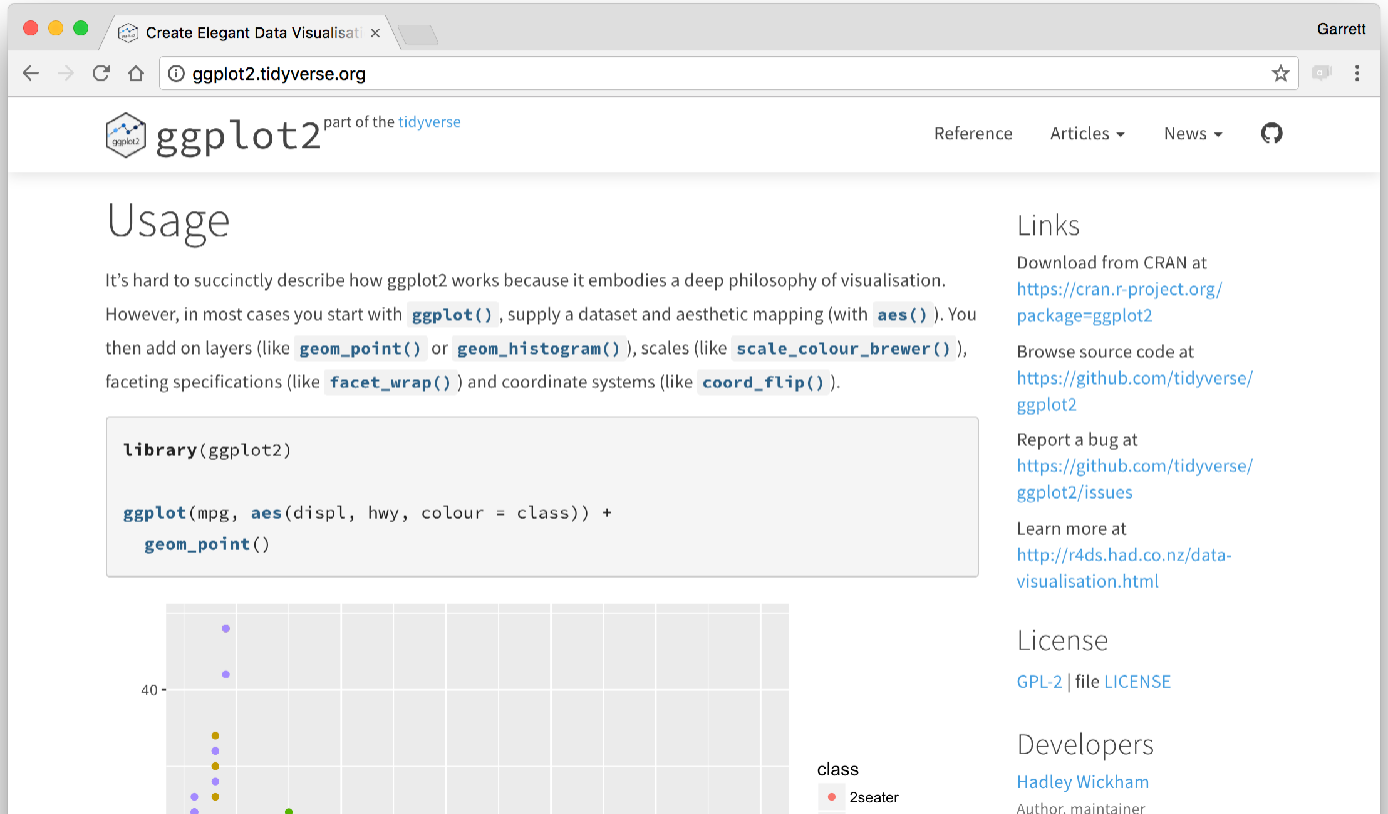
```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
    stat = <STAT>, position = <POSITION>) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION> +
  <SCALE_FUNCTION> +
  <THEME_FUNCTION>
```

required

Not required, sensible defaults supplied



ggplot2.tidyverse.org



Visualize Data with