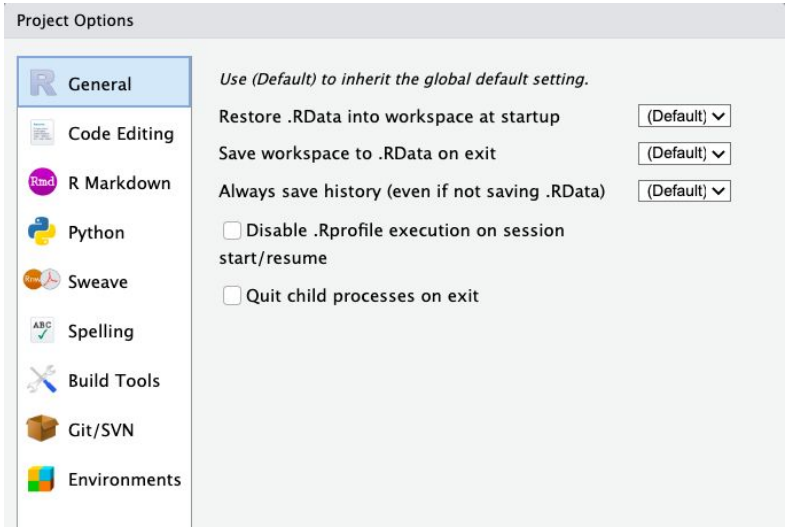
Data Visualization

Jen Cruz, Zichao Li, Isobel McEwen, Claire McLaughlin, Sanjana Srinivasan, Julien Chen

Set Up

- set up a new directory --> new project --> work in script --> save script regularly and especially before closing session
- use console for commands such as install
- use sections, section labels, and comments with command + shift + r and the #



Data Cleaning

1. Familiarize yourself with the data set
2. Check for structural errors
3. Check for data irregularities
4. Decide how to deal with missing values
5. Document data versions and changes made

Source: https://towardsdatascience.com/data-cleaning-in-r-made-simple-1b77303b0b17

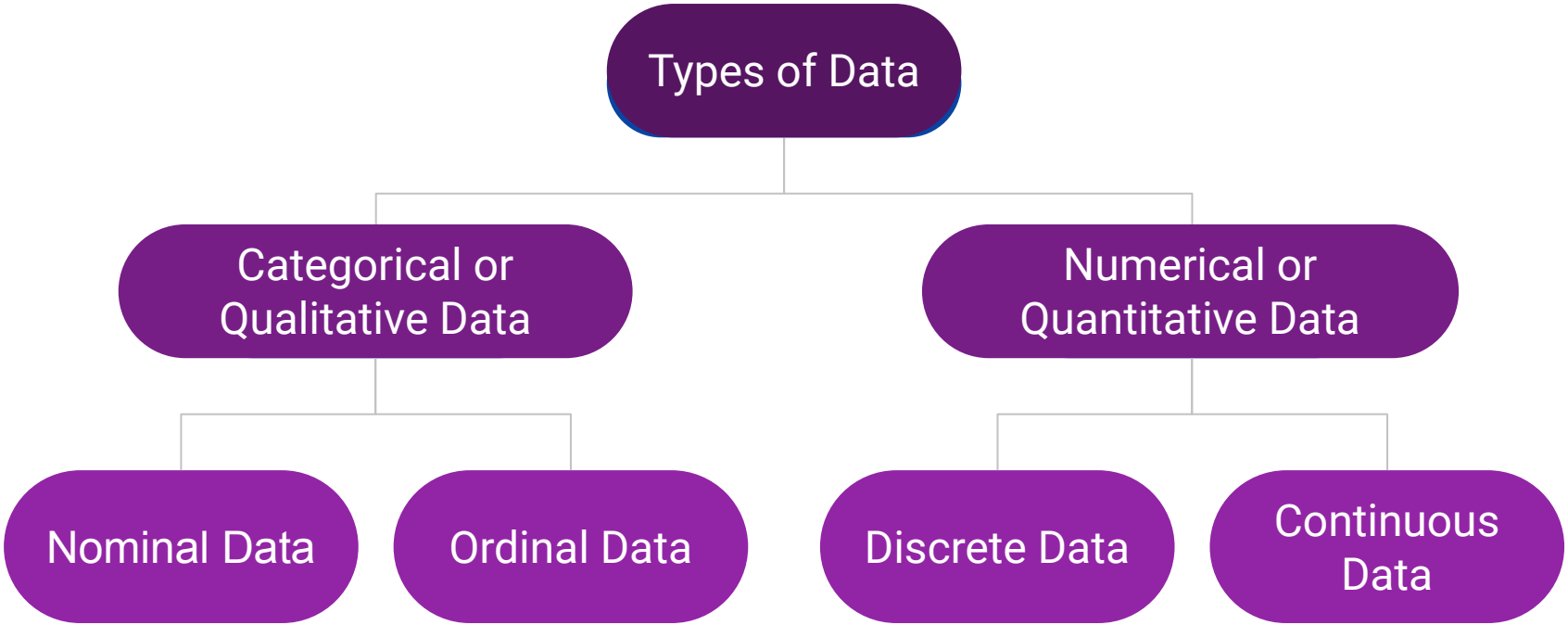
Dplyr

“Dplyr changed my life!” -Jarvis

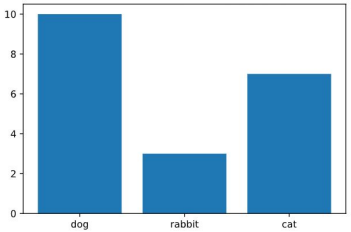
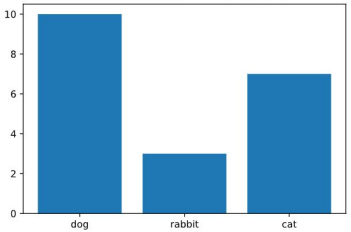
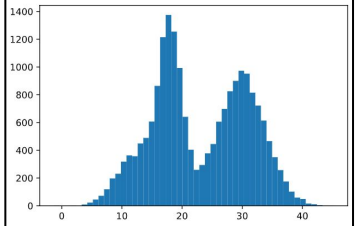
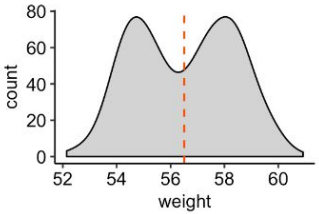
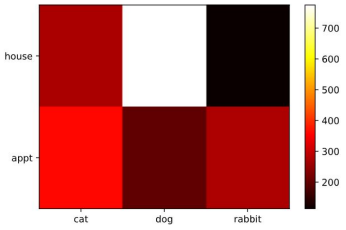
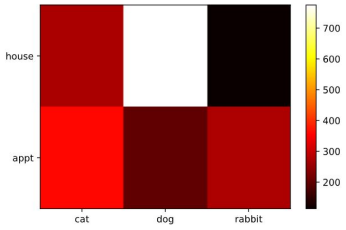
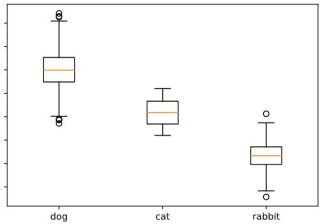
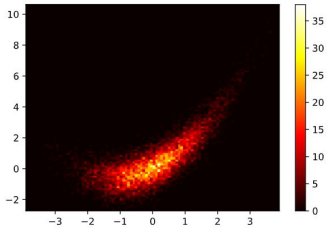
```
popsize_and_poverty <- popsize_and_poverty |>
  mutate(
    proportion_in_poverty = in_poverty / total_for_povert
  )

popsize_and_poverty <- popsize_and_poverty |>
  select(GEOID, popsize, proportion_in_poverty)
```

What type of data do you have?



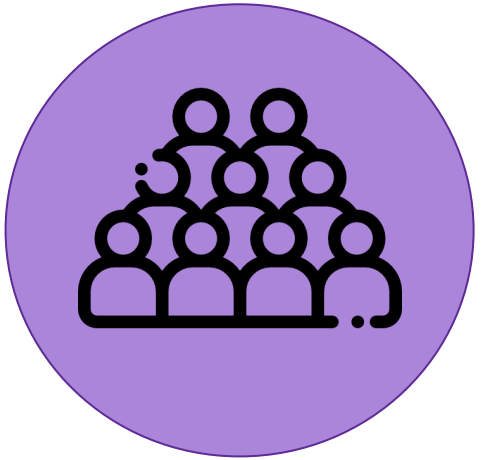
Picking a way to visualize your data

	Categorical	Ordinal	Discrete	Continuous
1-D	Bar Chart 	Bar Chart 	Histogram 	Density 
2-D	Heat Maps 	Heat Maps 	Box and Whisker 	Scatter Plot 

Goals of data visualization

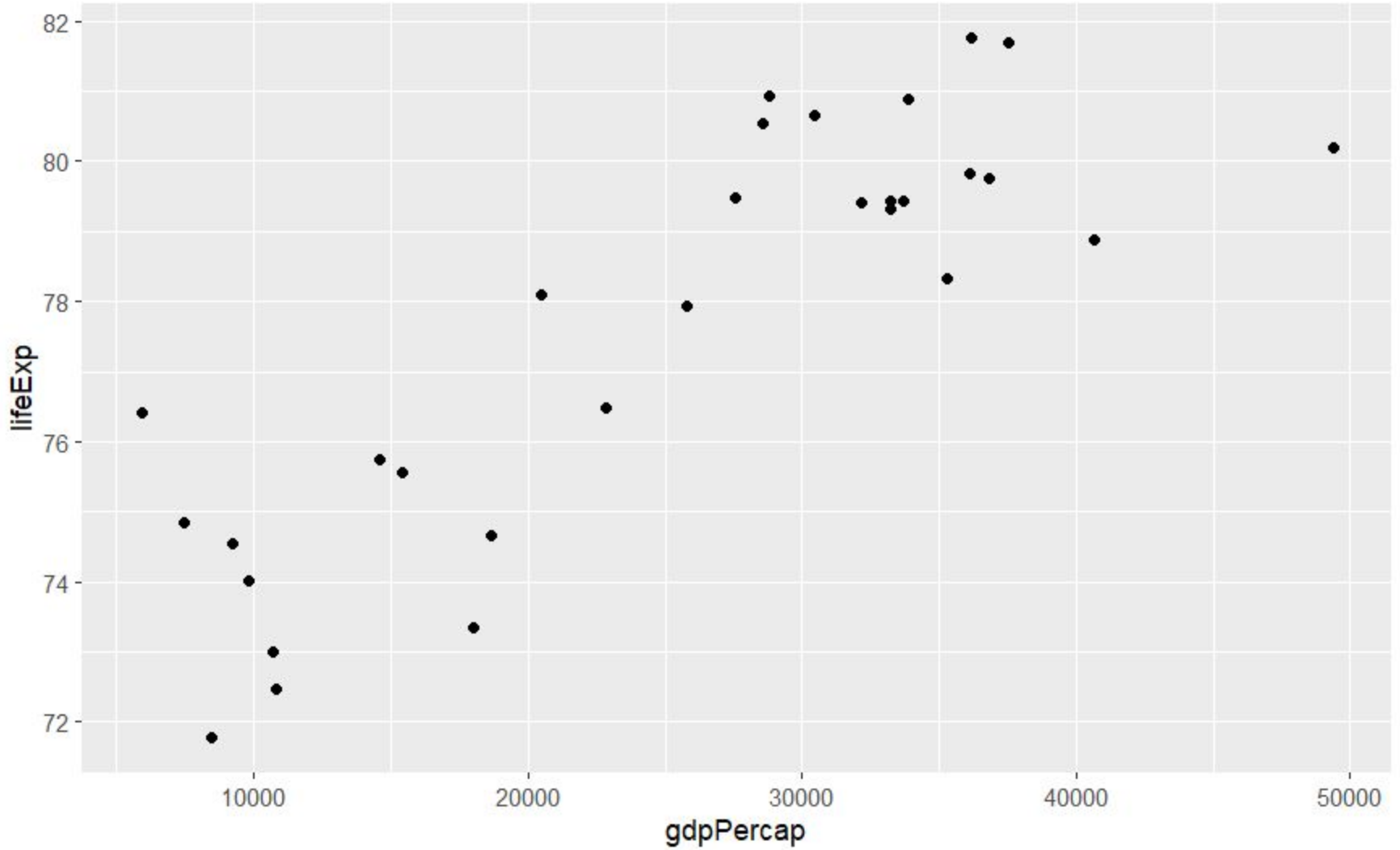
Understanding

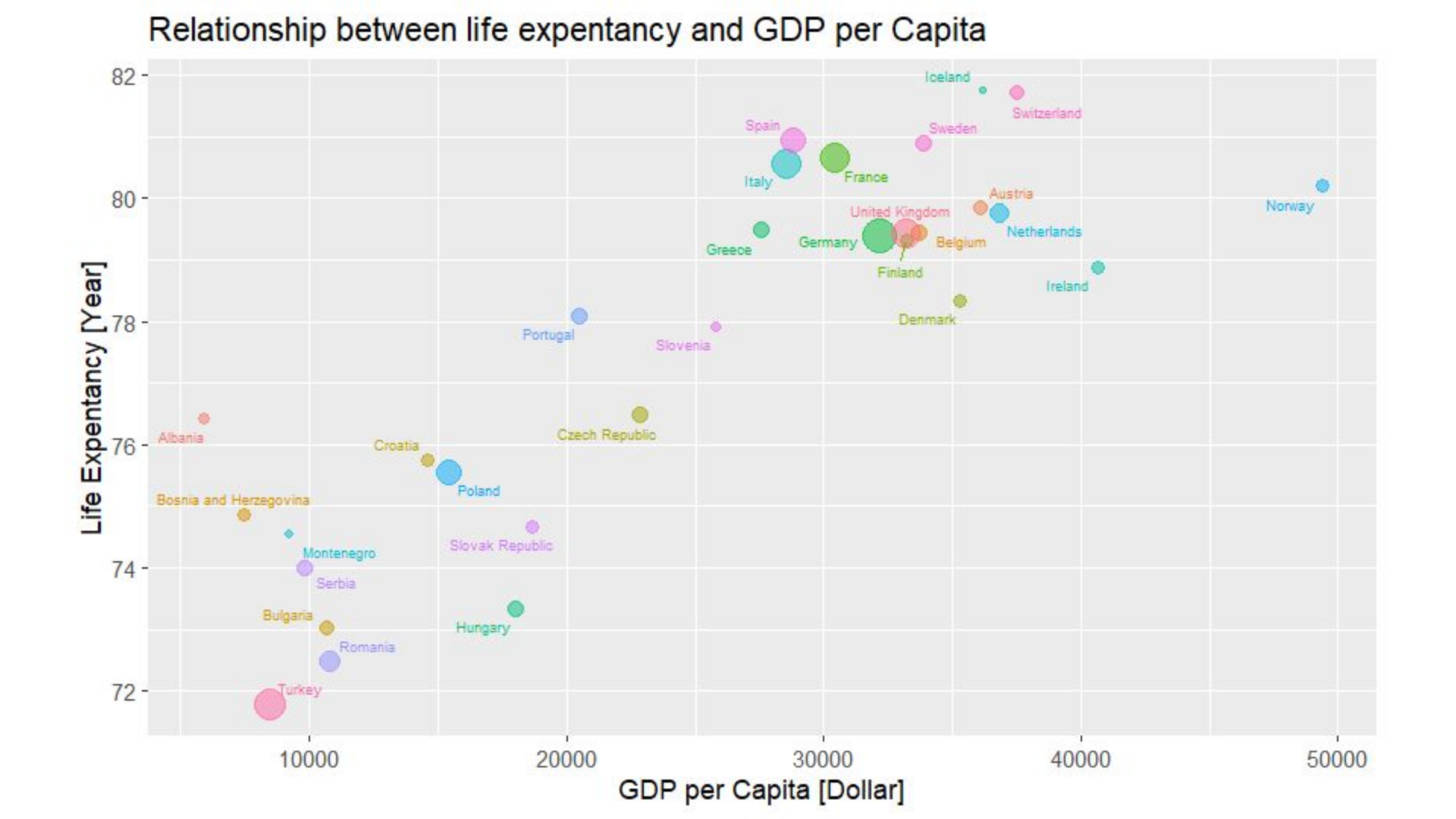
Storytelling



Accessibility

Adapted from: https://lisacharlottemuth.com/2017/03/10/why-do-we-visualize-data/


82
80
78
76
74
72
lifeExp
10000
20000
30000
40000
50000
gdpPercap

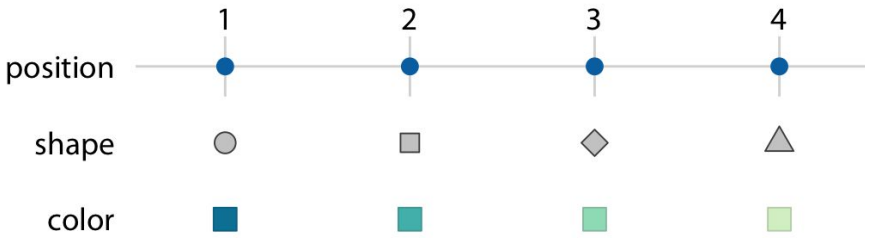

Relationship between life expentancy and GDP per Capita
Life Expentancy [Year]
GDP per Capita [Dollar]
82
80
78
76
74
72
10000
20000
30000
40000
50000
Iceland
Switzerland
Spain
Sweden
Italy
France
Norway
Austria
United Kingdom
Netherlands
Germany
Belgium
Greece
Finland
Ireland
Denmark
Portugal
Slovenia
Albania
Czech Republic
Croatia
Poland
Bosnia and Herzegovina
Slovak Republic
Montenegro
Serbia
Bulgaria
Hungary
Romania
Turkey

Scale

“A scale defines a unique mapping between data and aesthetics.”

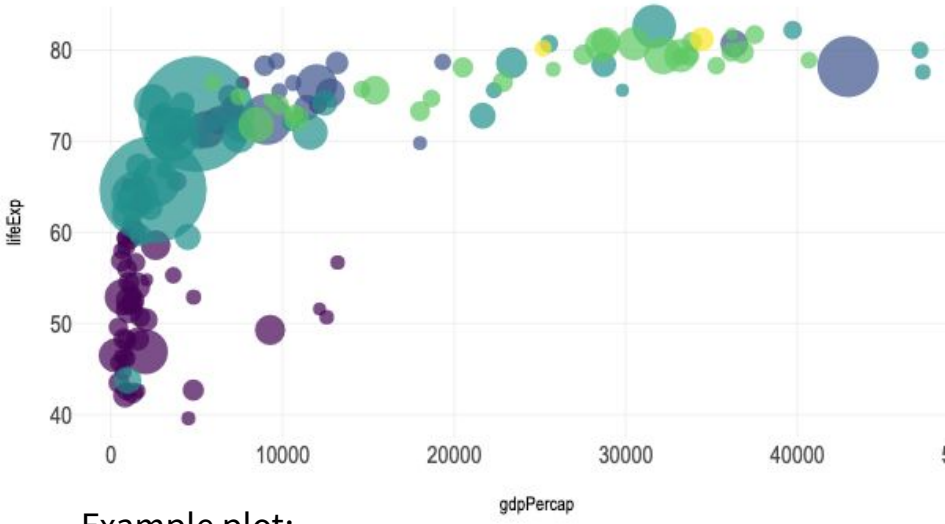
Rule of thumb: Each data value must have a unique scale.

3 different scales:



Source: https://clauswilke.com/dataviz/aesthetic-mapping.html

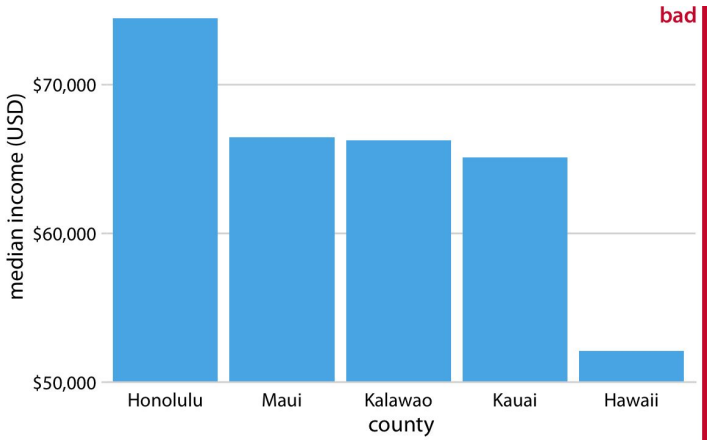
Example plot:

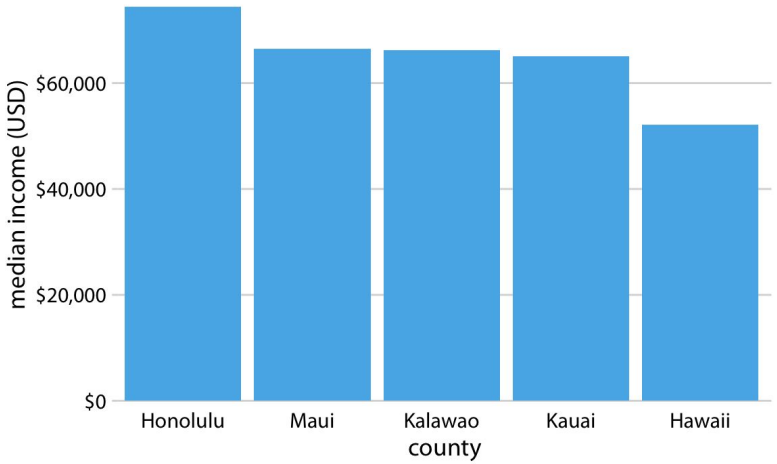


Interactive bubble chart – the R Graph Gallery (r-graph-gallery.com)

Axis

Quantitative axes:





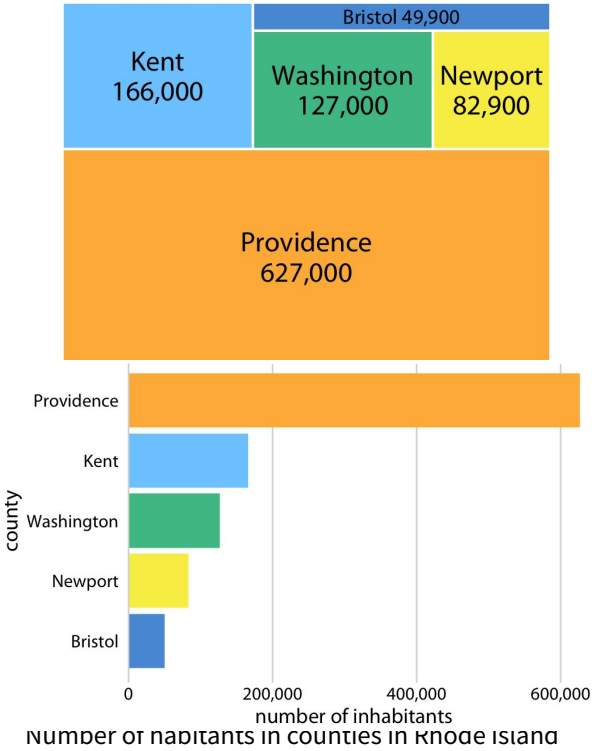
Example plot: 2 bar graphs of median income of 5 counties of Hawaii the state

Why do the income gaps look so different in the two graphs?

Source: https://clauswilke.com/dataviz/proportional-ink.html

Axis

Categorical axes:



Number of habitants in counties in Rhode Island

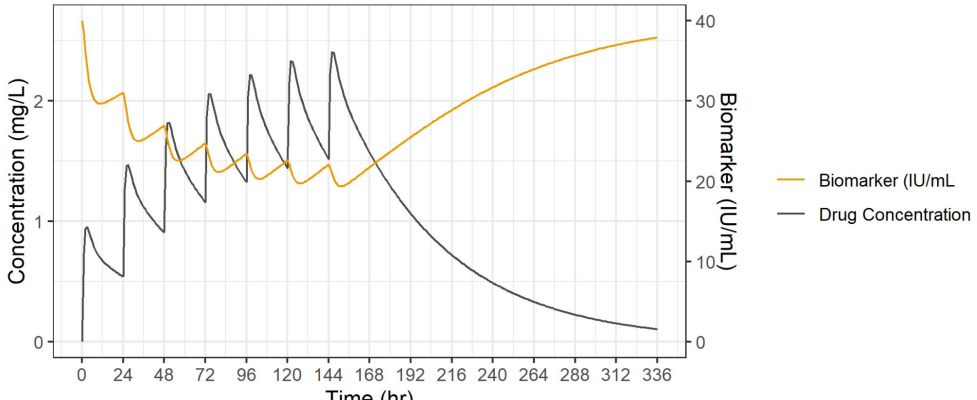
Source: https://clauswilke.com/dataviz/proportional-ink.html
https://finchstudio.io/blog/ggplot-dual-y-axes/

Dual axes: two different y axes

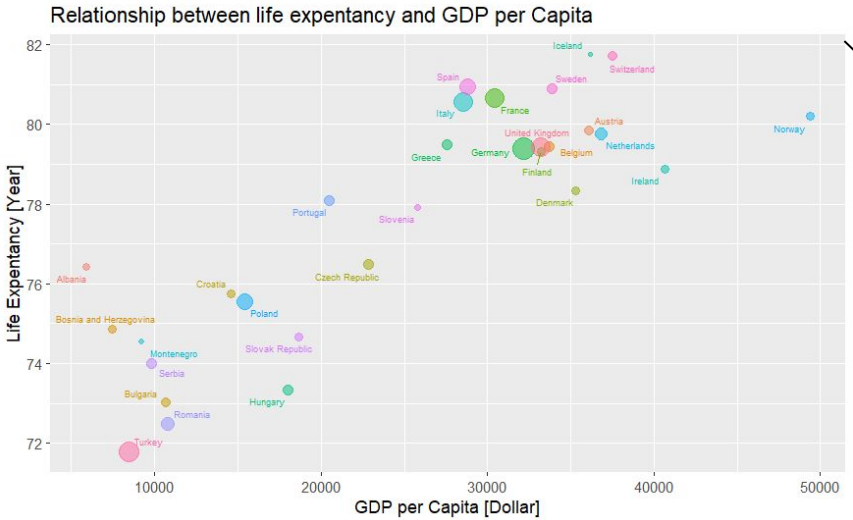
```r
scale = 15

pkpd <- ggplot(res, aes(x = time, y = CP)) +
  geom_line(aes(color = "Drug Concentration")) +
  geom_line(aes(y = RESP/scale, color = "Biomarker (IU/mL")) +
  scale_x_continuous(breaks = seq(0, 336, 24)) +
  scale_y_continuous(sec.axis = sec_axis(~.*scale, name="Biomarker (IU/mL)")) +
  labs(x = "Time (hr)", y = "Concentration (mg/L)", color = "") +
  scale_color_manual(values = c("orange2", "gray30"))

print(pkpd)
```



Style for Accessibility




```
devtools::install_github("AndreaCirilloAC/paletter")
```

#E8588E	#E5AFC6	#FEC3D7	#BD5EF7
#FCB4CF	#B9CB8C	#E78472	#F884B2
#9BE0B7	#C1D3FD		

Viz Palette by Elijah Meeks & Susie Lu

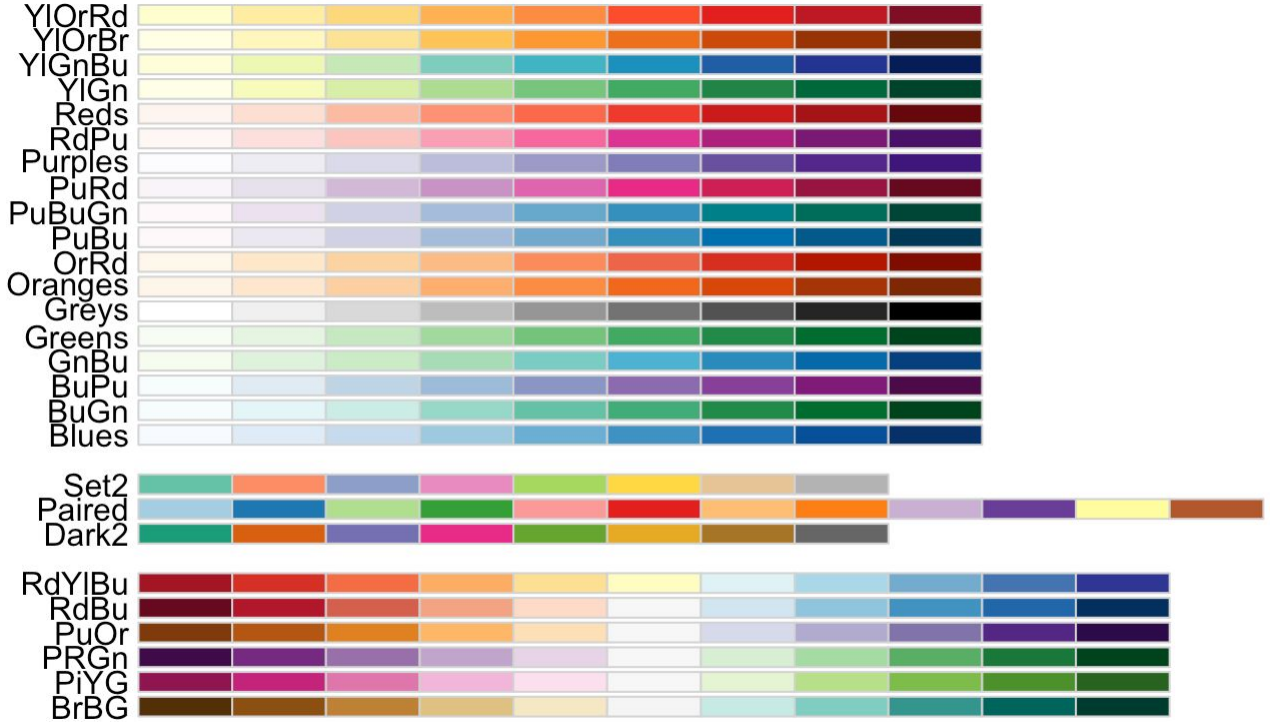
COLOR REPORT

Arcs link colors difficult to tell apart as:

- Lines or small points
- Medium areas
- Large areas

- Minimize name conflicts for categorical palettes

```
install.packages("RColorBrewer")
library(RColorBrewer)
display.brewer.all(colorblindFriendly = TRUE)
```

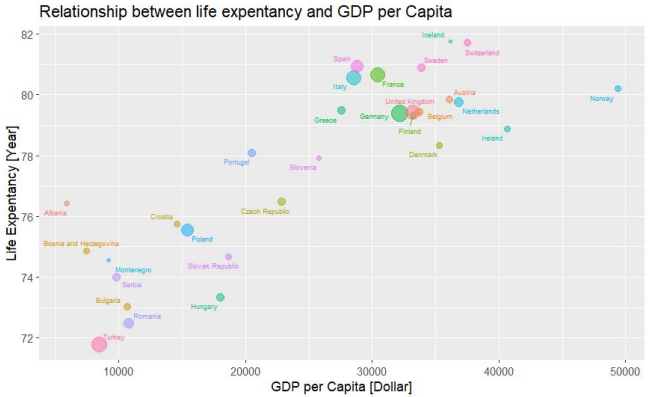


```
install.packages("wesanderson")
library(wesanderson)
```

```
> names(wes_palettes)
 [1] "BottleRocket1"  "BottleRocket2"  "Rushmore1"      "Rushmore"
 [5] "Royal1"         "Royal2"         "Zissou1"        "Darjeeling1"
 [9] "Darjeeling2"    "Chevalier1"     "FantasticFox1"  "Moonrise1"
[13] "Moonrise2"      "Moonrise3"      "Cavalcanti1"    "GrandBudapest1"
[17] "GrandBudapest2" "IsleofDogs1"    "IsleofDogs2"
```

```
wes_palette("FantasticFox1")
```

Style for Accessibility (cont.)

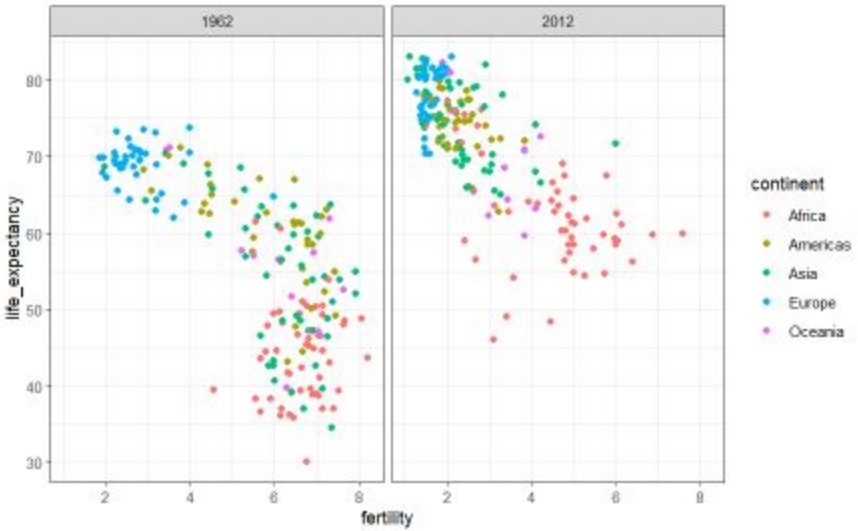



```
library(BrailleR)
```

```
## This chart has title 'Relationship between life expentancy and GDP per Capita'.
## It has x-axis 'GDP per Capita [Dollar]' with labels 10000, 20000, 30000, 40000 and 50000.
## It has y-axis 'Life Expentancy [Year]' with labels 72, 74, 76, 78, 80 and 82.
## In this chart colour is used to show country. The legend that would normally indicate this has been hidden.
## In this chart size is used to show pop. The legend that would normally indicate this has been hidden.
## It has 2 layers.
## Layer 1 is a set of 30 points.
## Layer 1 has alpha set to 0.5.
## Layer 2 is a textrepel graph that VI can not process.
## Layer 2 has size set to 2.
```

facet_wrap

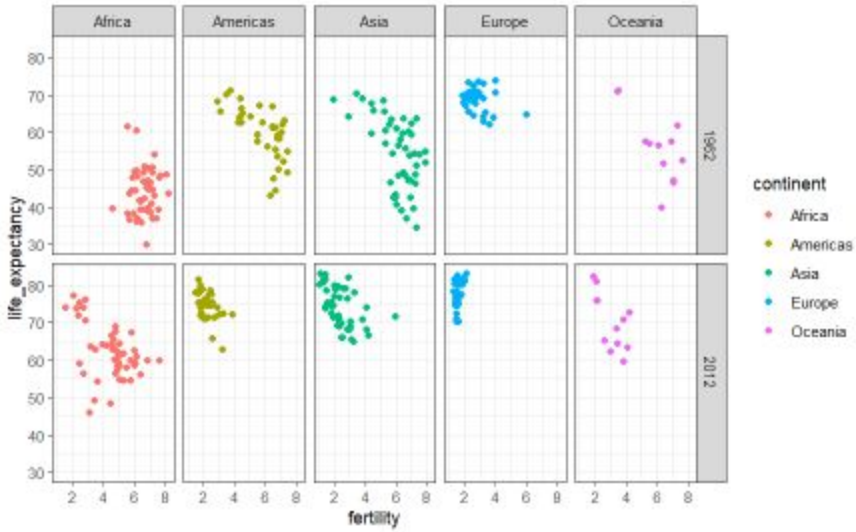
Faceting can group and visualize each subset. Facet_wrap can only visualize non-empty univariate.




```r
filter(gapminder, year%in%c(1962, 2012)) |> ggplot(aes(fertility, life_expectancy, col = continent)) +
geom_point() +
facet_wrap(.~year)
```

facet_grid

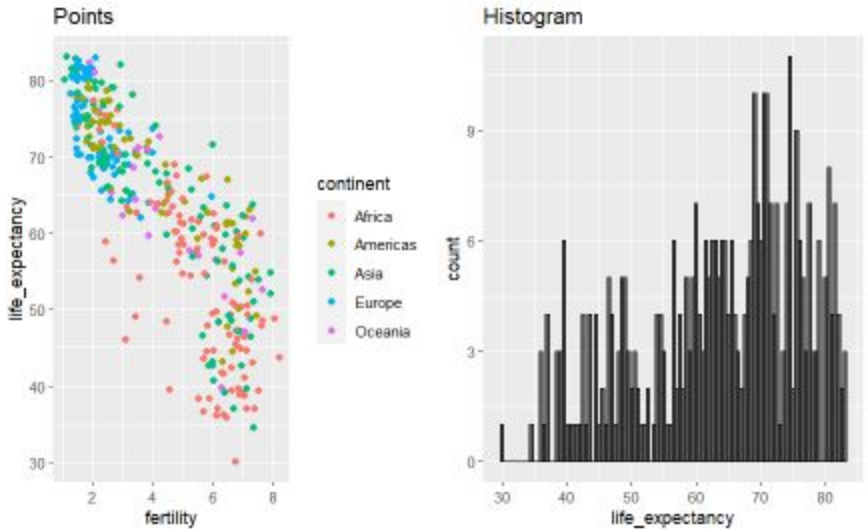
Facet_grid can visualize bivariate even some are empty.




```{r}
filter(gapminder, year%in%c(1962, 2012)) |> ggplot(aes(fertility, life_expectancy, col = continent)) +
geom_point() +
facet_grid(year~continent)
```

grid.arrange

grid.arrange can layout multiple graphs on the same page

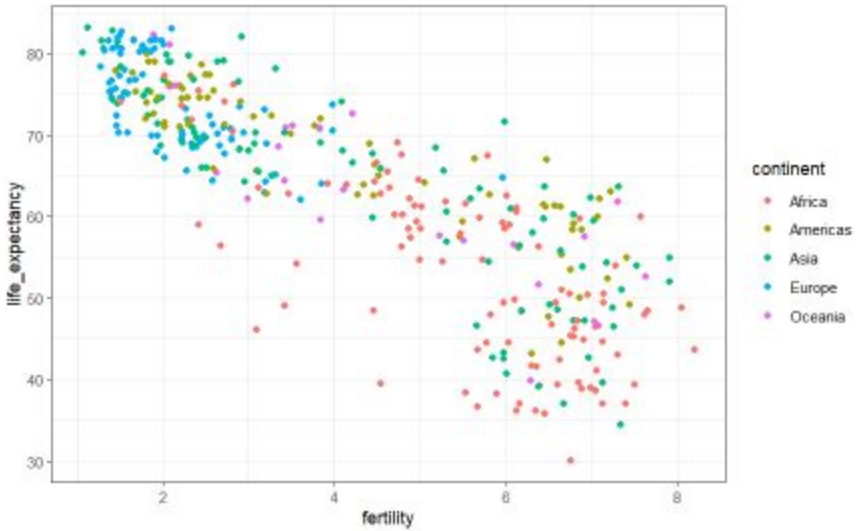



```{r}
p1=filter(gapminder, year%in%c(1962, 2012)) |> ggplot(aes(fertility, life_expectancy, col = continent)) + geom_point() + ggtitle("Points")

p2=filter(gapminder, year%in%c(1962, 2012)) |> ggplot(aes(life_expectancy)) + geom_histogram(binwidth = 0.5, color = "black") + ggtitle("Histogram")

grid.arrange(p1, p2, ncol = 2)```
```

ggplot2 can build a complex plot a layer at a time. Each layer can come from a different dataset and have a different aesthetic mapping. Layers are created using geom_* or stat_*




```{r}
filter(gapminder, year%in%c(1962, 2012)) |> ggplot(aes(fertility, life_expectancy, col = continent)) + geom_point()
```

```
p + layer(
  mapping = NULL,
  data = NULL,
  geom = "point",
  stat = "identity",
  position = "identity"
)
```

Geom*

One variable:

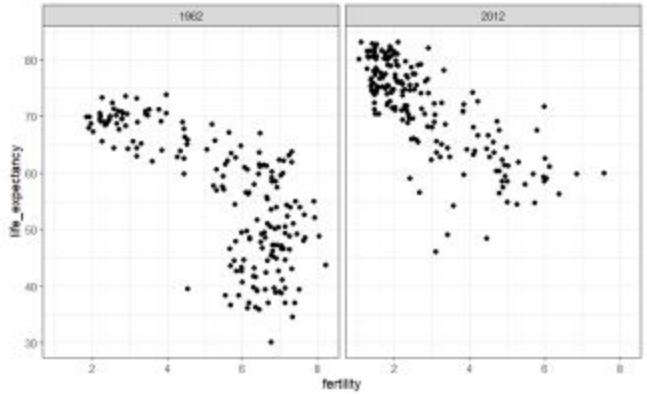
- geom_bar(): display distribution of discrete variable.
- geom_histogram(): bin and count continuous variable, display with bars.
- geom_density(): smoothed density estimate.
- geom_dotplot(): stack individual points into a dot plot.
- geom_freqpoly(): bin and count continuous variable, display with lines.

Two variables:

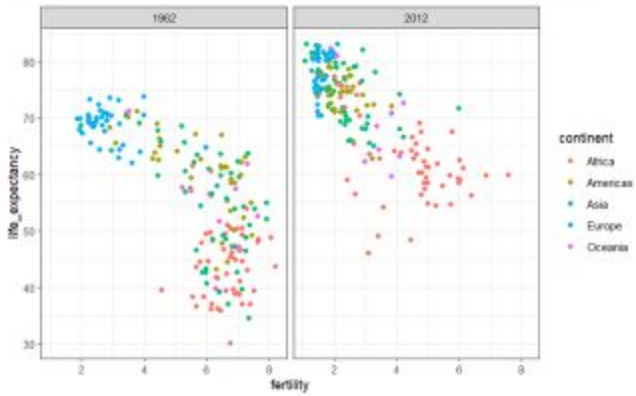
- geom_point(): scatterplot.
- geom_quantile(): smoothed quantile regression.
- geom_rug(): marginal rug plots.
- geom_smooth(): smoothed line of best fit.
- geom_text(): text labels.

Aesthetic Mapping*

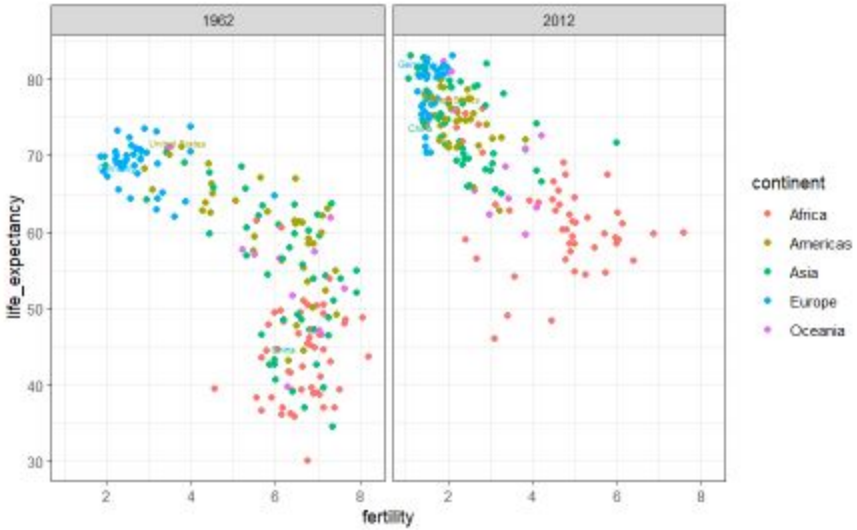
```{r}
filter(gapminder, year==2012) |>
ggplot(aes(fertility, life_expectancy)) +
geom_point() +
Facet_wrap(. ~ year)
```




```{r}
filter(gapminder, year==2012) |>
ggplot(aes(fertility, life_expectancy,
col=continent)) + geom_point() +
Facet_wrap(. ~ year)
```



Data




```{r}
highlight <- c("Germany","China", "United States")
filter(gapminder, year %in% c(1962, 2012) ) %>%  ggplot(aes(fertility, life_expectancy, color = continent)) + geom_point() + geom_text_repel(size = 2, show.legend = FALSE, aes(fertility, life_expectancy, label=country),  data = filter(gapminder, year %in% c(1962, 2012) & country %in% highlight)) + facet_grid(. ~ year)
```

Stats*

stat_bin(): geom_bar(), geom_freqpoly(), geom_histogram()
stat_bin2d(): geom_bin2d()
stat_bindot(): geom_dotplot()
stat_binhex(): geom_hex()
stat_boxplot(): geom_boxplot()
stat_contour(): geom_contour()
stat_quantile(): geom_quantile()
stat_smooth(): geom_smooth()
stat_sum(): geom_count()

Position*

Three position adjustments for points:

- position_nudge(): move points by a fixed offset.
- position_jitter(): add a little random noise to every position.
- position_jitterdodge(): dodge points within groups, then add a little random noise.

Three positions adjustments for bars:

- position_stack(): stack overlapping bars (or areas) on top of each other.
- position_fill(): stack overlapping bars, scaling so the top is always at 1.
- position_dodge(): place overlapping bars (or boxplots) side-by-side.

Saving and Exporting Visualizations

If using ggplot:

```
ggsave(
  filename,                                  ← The name of the file locally        ex. “mygraphic.png”
  plot = last_plot(),                        ← Plot to export                      ex. plot1
  device = NULL,                             ← Type of export                      ex. “Png”, “jpeg”
  path = NULL,                               ← File location to be saved           ex. “mydocuments/folder1”
  scale = 1,                                 ← Multiplicative factor
  width = NA,                                ← Dimensions of graphic               ex. 3,5,10 (inches, pixels…)
  height = NA,
  units = c("in", "cm", "mm", "px"),         ← unit of size
  dpi = 300,                                 ← Set resolution                      ex. “retina”,300, “screen”
  limitsize = TRUE,                          ← max image size of 50 x 50 inches
  bg = NULL,                                 ← Background color                    ex. “gray”
  ...
)
```

Interactability

If looking for individuals to be able to change elements in your visualization, try creating an **RShiny App**

Shiny App Example

- Easy way to share ggplot figures, and others, in a way that allows users to change elements of the visualization easily
- See more information here: R Shiny Tutorial

Key Takeaways

- Don’t take shortcuts in your code
- Be intentional with your visualizations
- Making good visualizations of data takes time
- Making accessible visualizations increases impact by broadening audience