

# Week 8: *Comparisons*

 EMSE 4572 / 6572: Exploratory Data Analysis

 John Paul Helveston

 October 18, 2023

*Tip of the week*

Shortcut keys

# 1) Quick shortcuts

Insert a  $\leftarrow$  operator:

- **Windows:** ALT + ←
- **Mac:** OPTION + ←

Insert a  $\%>\%$  operator:

- **Windows:** CTRL + SHIFT + M
- **Mac:** COMMAND + SHIFT + M

## 2) Edit multiple lines of code at once

1. Press and hold **ALT** (Windows) or **OPTION** (Mac)
2. Select multiple lines of code

<https://twitter.com/i/status/995394452821721088>

"At the heart of quantitative reasoning is a single question: **Compared to what?**"

-- Edward Tufte

# Today's data

```
college_all_ages <- read_csv(here('data', 'college_all_ages.csv'))  
gapminder        <- read_csv(here('data', 'gapminder.csv'))  
marathon         <- read_csv(here('data', 'marathon.csv'))  
milk_production  <- read_csv(here('data', 'milk_production.csv'))  
internet_regions <- read_csv(here('data', 'internet_users_region.csv'))
```

# New packages

```
install.packages("ggrepel")  
install.packages("ggridges")
```

# Week 8: *Comparisons*

1. Comparing to a reference

2. Comparing variables

**BREAK**

3. Comparing distributions

# Week 8: *Comparisons*

1. Comparing to a reference

2. Comparing variables

BREAK

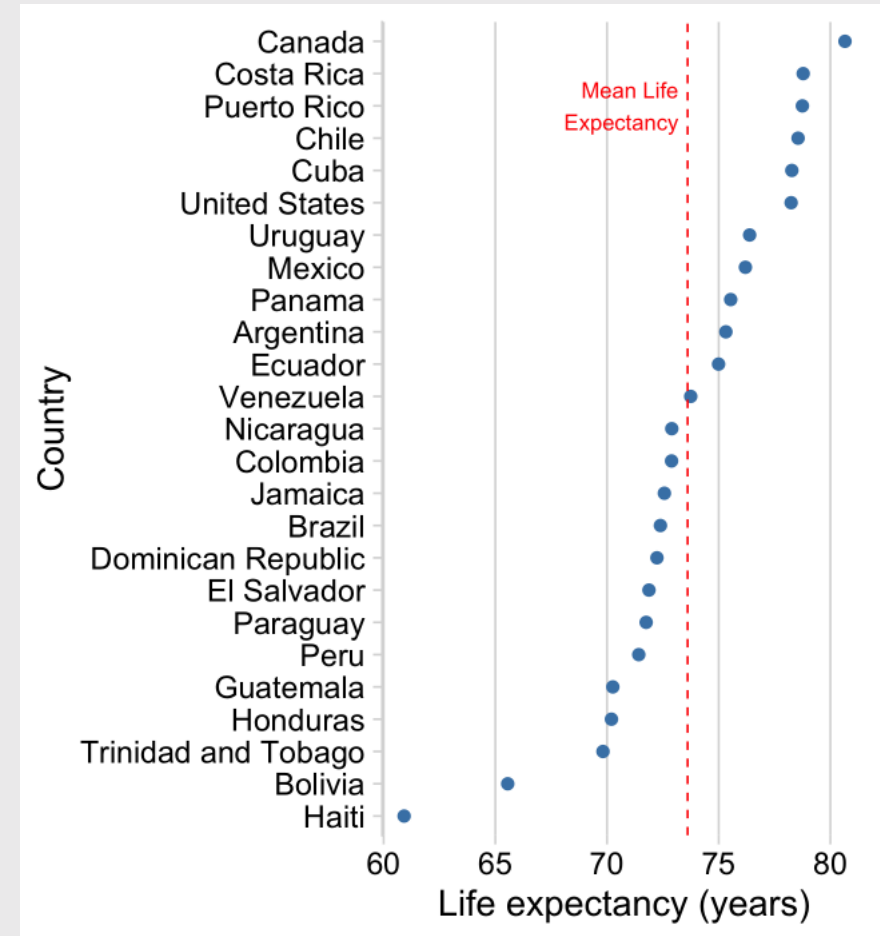
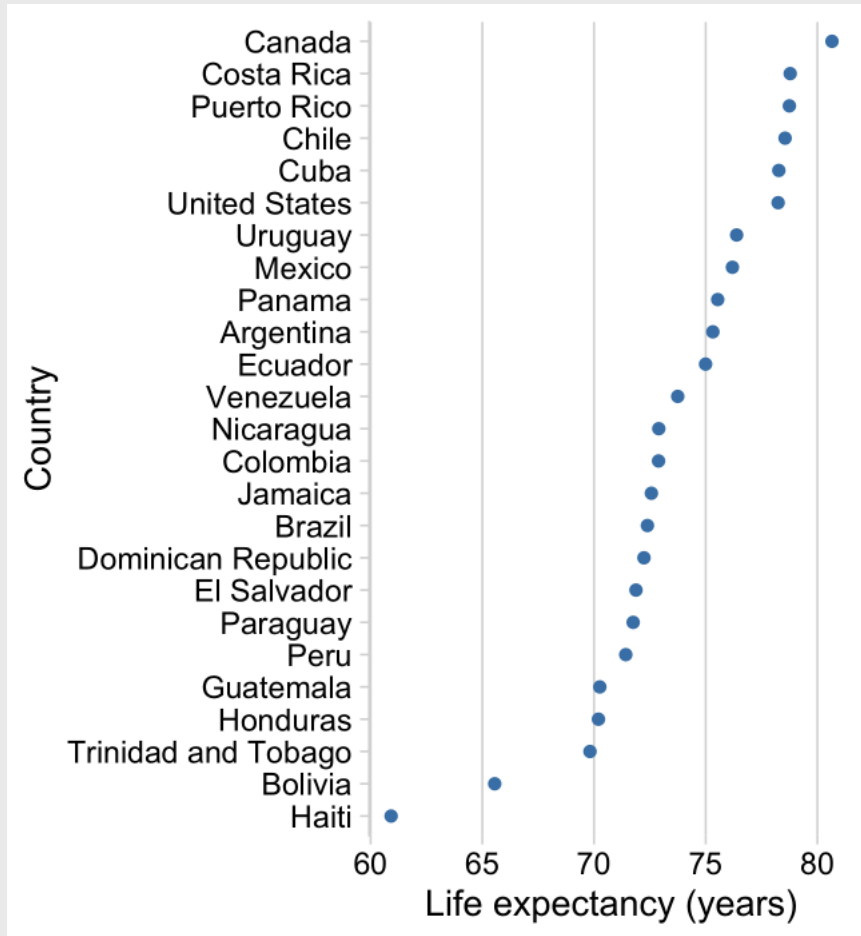
3. Comparing distributions



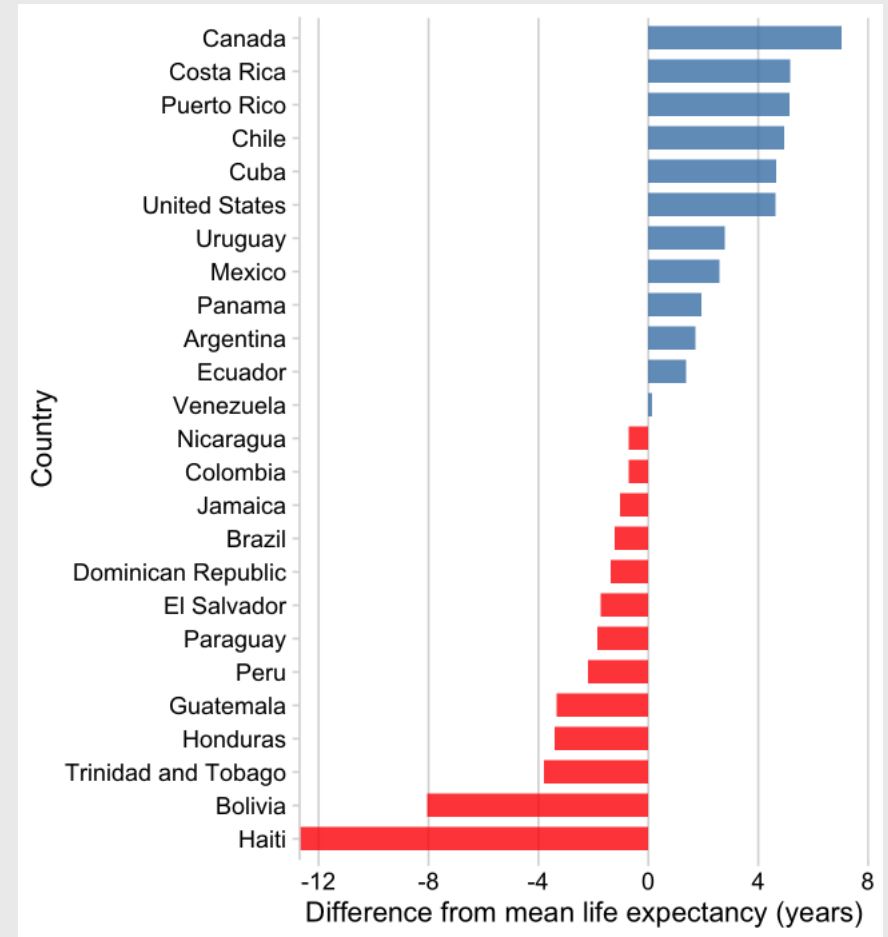
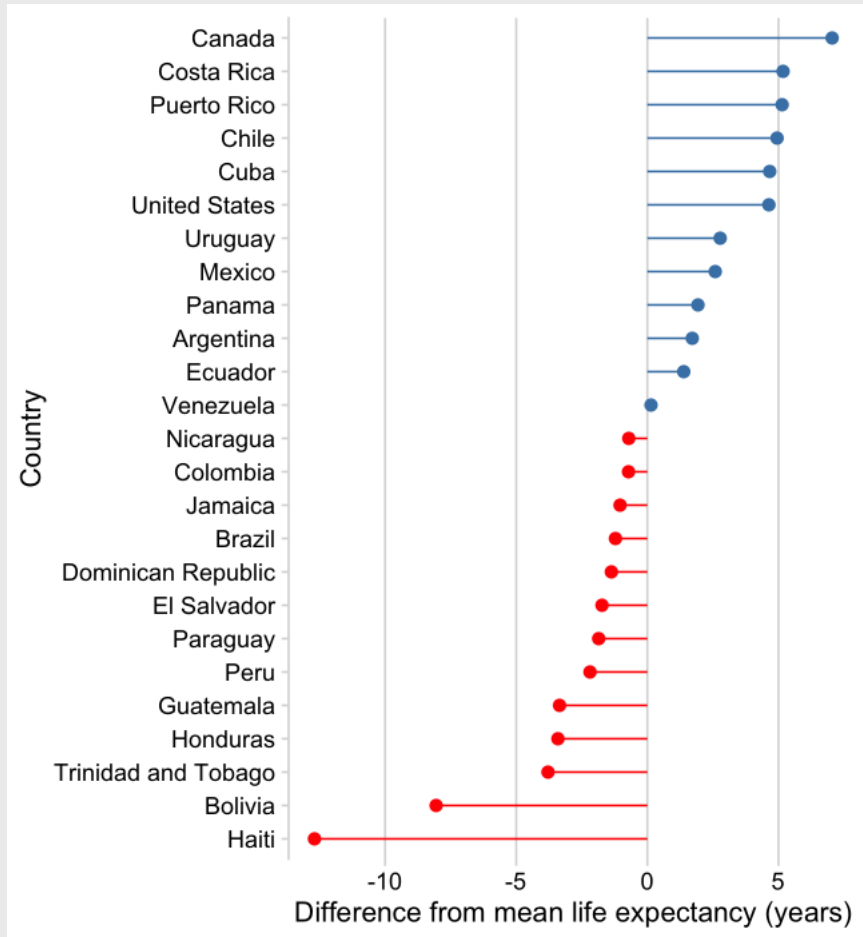
For this section, we'll be using this data frame:

```
gapminder_americas <- gapminder %>%  
  filter(continent == "Americas", year == 2007) %>%  
  mutate(country = fct_reorder(country, lifeExp))
```

# Use reference lines to add context to chart



# Or make zero the reference line

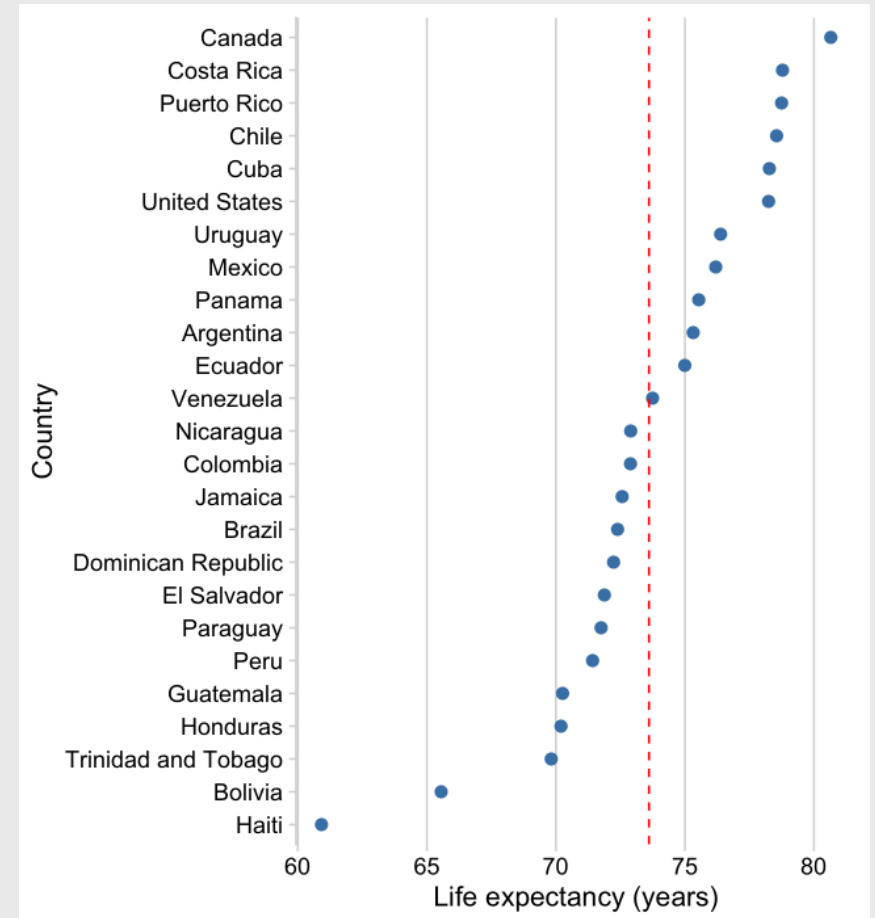


# How to add a reference line

Add horizontal line with `geom_hline()`

Add vertical line with `geom_vline()`

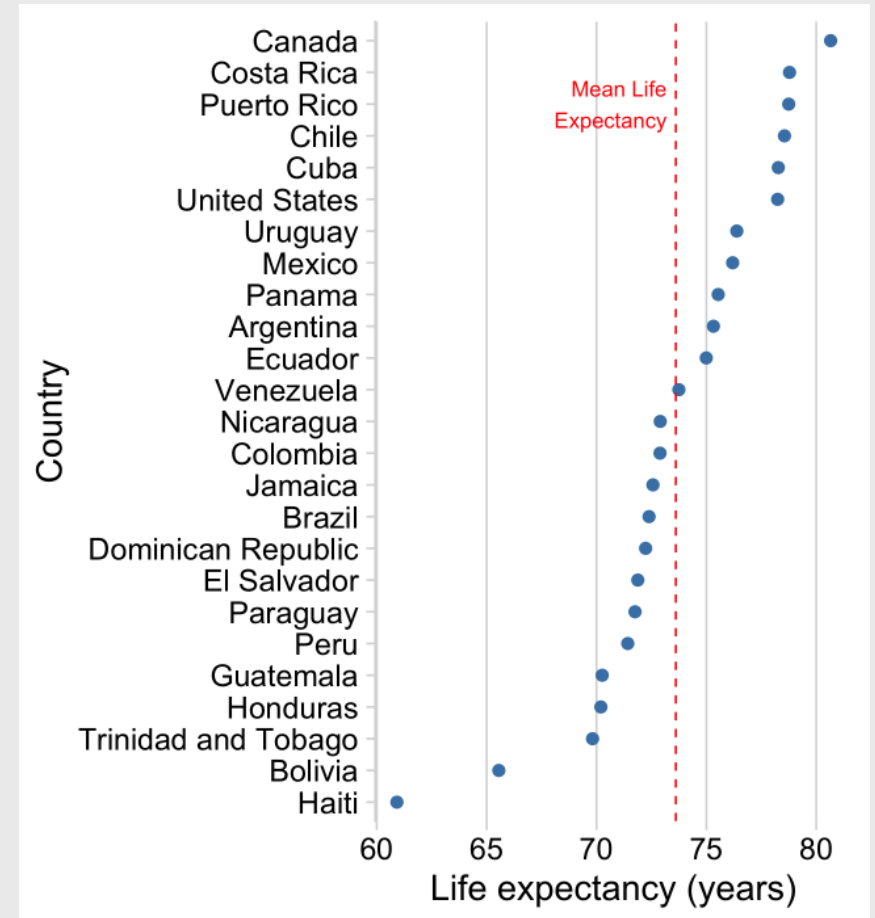
```
ggplot(gapminder_americas) +  
  geom_point(  
    aes(x = lifeExp, y = country),  
    color = 'steelblue', size = 2.5) +  
  geom_vline(  
    xintercept = mean(gapminder_americas$lifeExp),  
    color = 'red', linetype = 'dashed') +  
  theme_minimal_vgrid() +  
  labs(x = 'Life expectancy (years)',  
       y = 'Country')
```



# How to add a reference line

Add text with `annotate()`

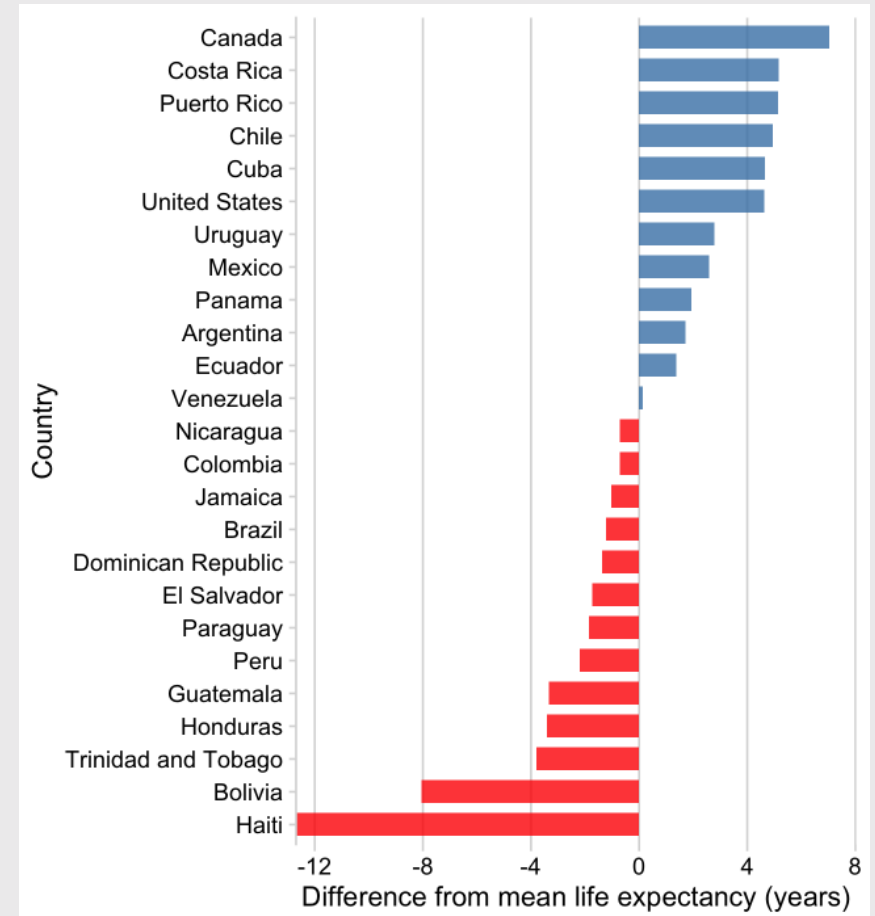
```
ggplot(gapminder_americas) +  
  geom_point(  
    aes(x = lifeExp, y = country),  
    color = 'steelblue', size = 2.5) +  
  geom_vline(  
    xintercept = mean(gapminder_americas$lifeExp),  
    color = 'red', linetype = 'dashed') +  
  annotate(  
    'text', x = 73.2, y = 'Puerto Rico',  
    color = 'red', hjust = 1,  
    label = 'Mean Life\nExpectancy') +  
  theme_minimal_vgrid() +  
  labs(x = 'Life expectancy (years)',  
       y = 'Country')
```



# How to make zero the reference point

```
gapminder_diverging <- gapminder_americas %>%  
  mutate(  
    # Subtract the mean  
    lifeExp = lifeExp - mean(lifeExp),  
    # Define the fill color  
    color = ifelse(lifeExp > 0, 'Above', 'Below'))
```

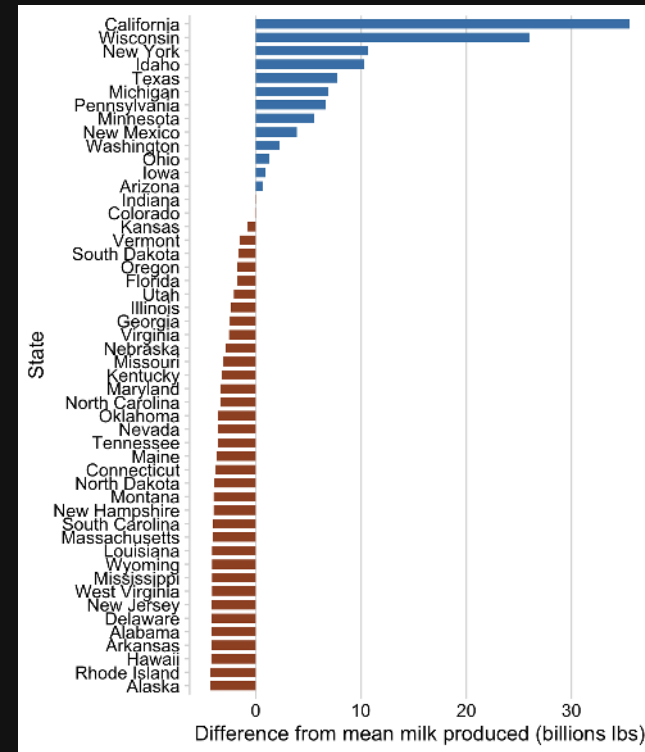
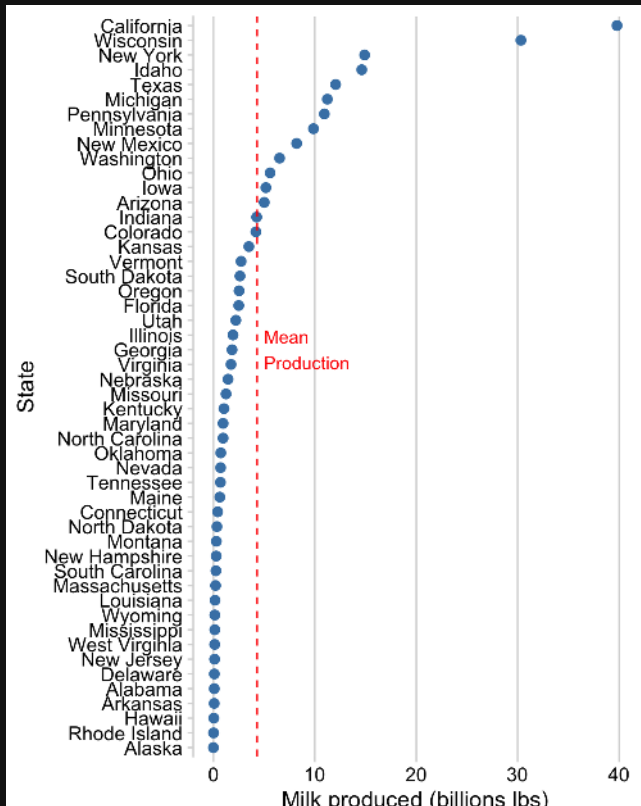
```
ggplot(gapminder_diverging) +  
  geom_col(  
    aes(x = lifeExp, y = country, fill = color),  
    width = 0.7, alpha = 0.8) +  
  scale_fill_manual(  
    values = c('steelblue', 'red')) +  
  theme_minimal_vgrid() +  
  theme(legend.position = 'none') +  
  labs(  
    x = 'Country',  
    y = 'Difference from mean life expectancy (years)')
```



# Your turn - comparing to a reference

15:00

Make a ranking chart and either a) add a reference line or b) show the ranking as the *difference* from a reference (e.g. the mean). Use any dataset you want in the "data" folder (examples below using `milk_production.csv`)



# Week 8: *Comparisons*

1. Comparing to a reference

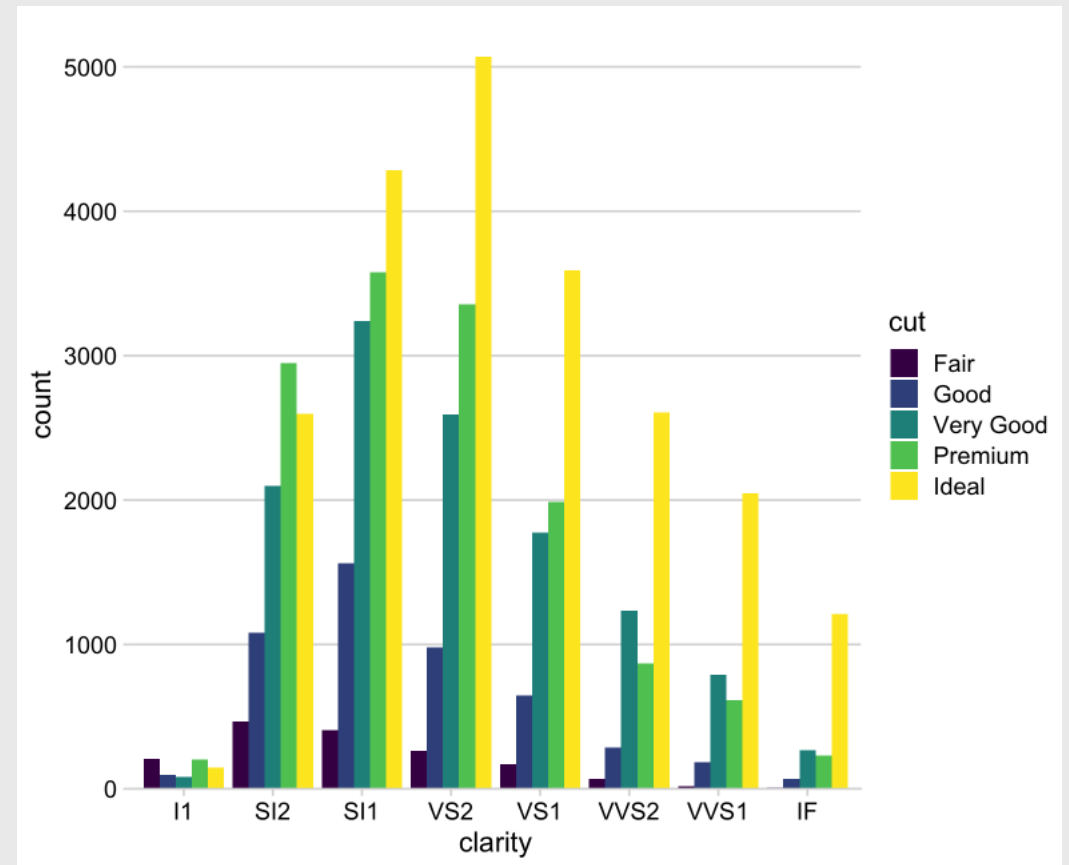
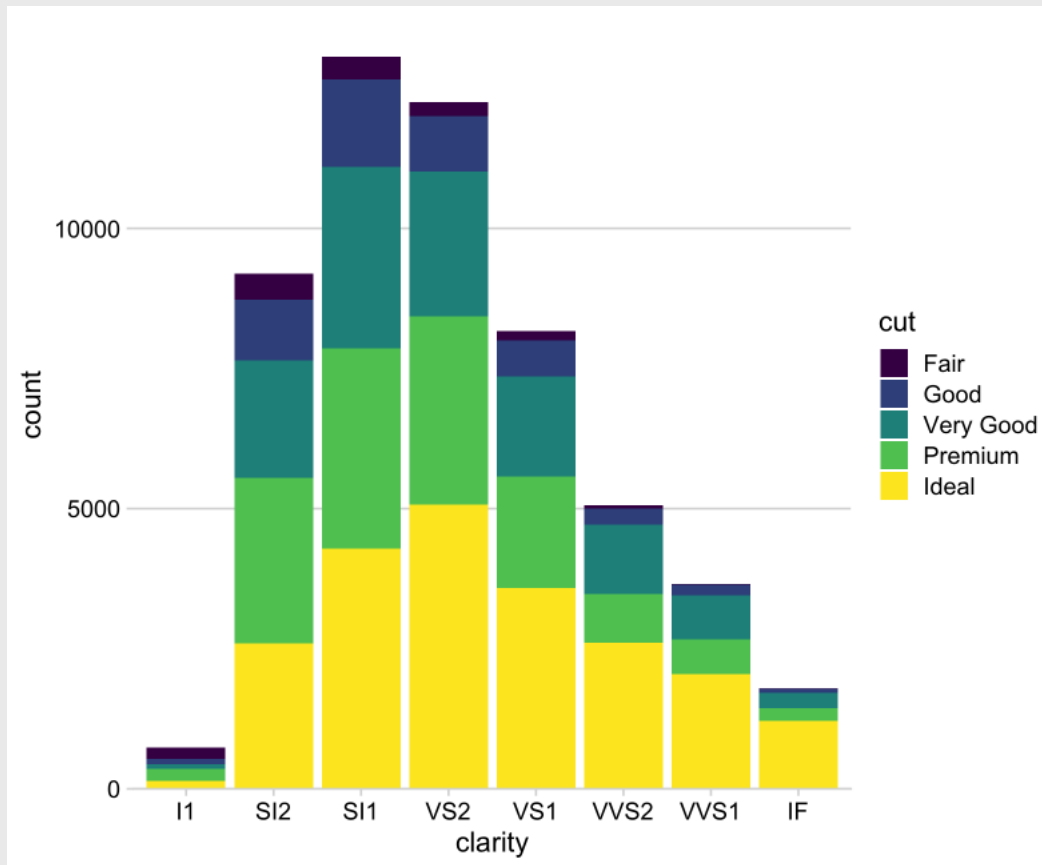
2. Comparing variables

BREAK

3. Comparing distributions

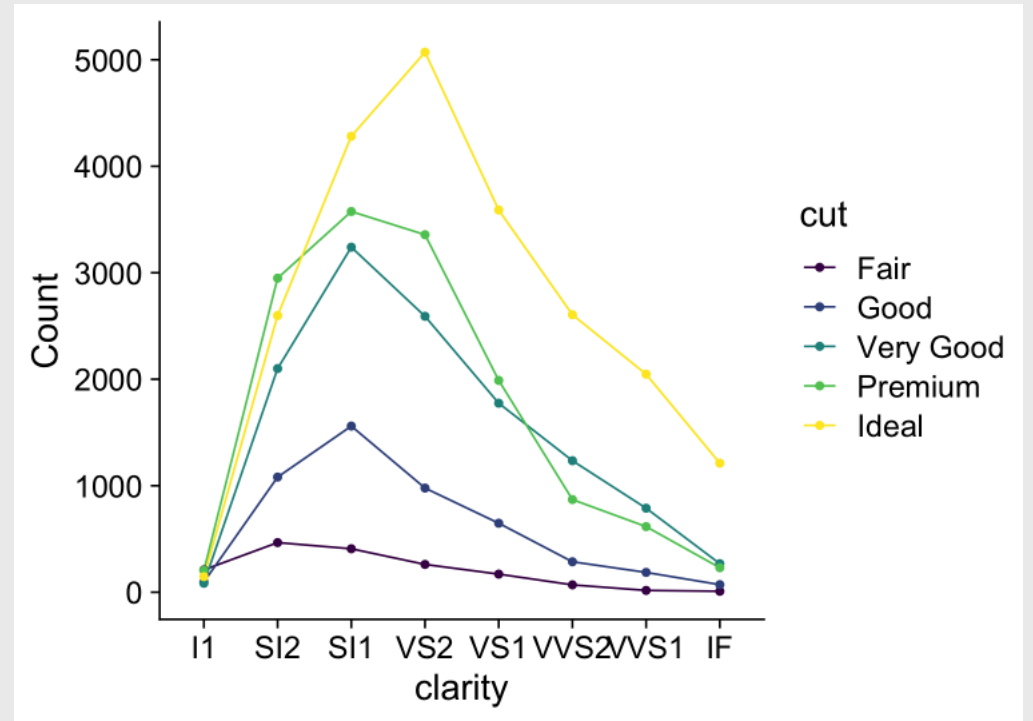


# Neither of these charts are great



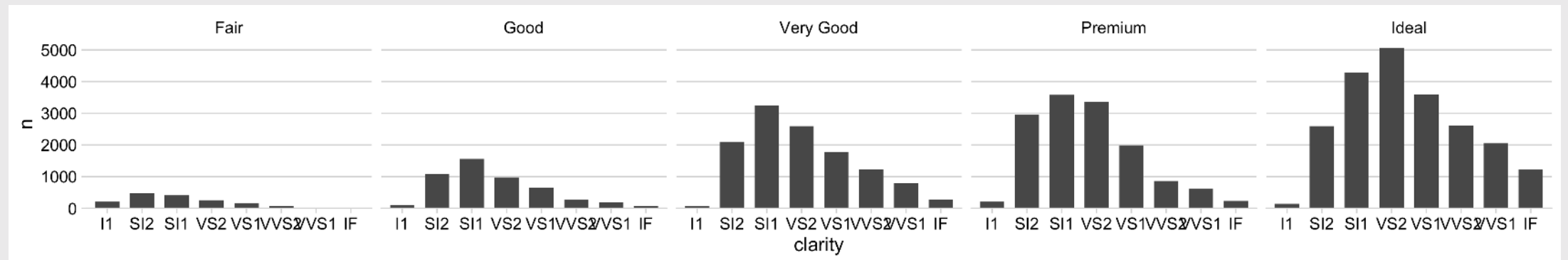
# "Parallel Coordinates" plots work well

```
diamonds %>%  
  count(clarity, cut) %>%  
  ggplot(  
    aes(x = clarity, y = n,  
        color = cut, group = cut)) +  
  geom_line() +  
  geom_point() +  
  scale_y_continuous(limits = c(0, 5100)) +  
  theme_half_open(font_size = 18) +  
  labs(y = "Count")
```



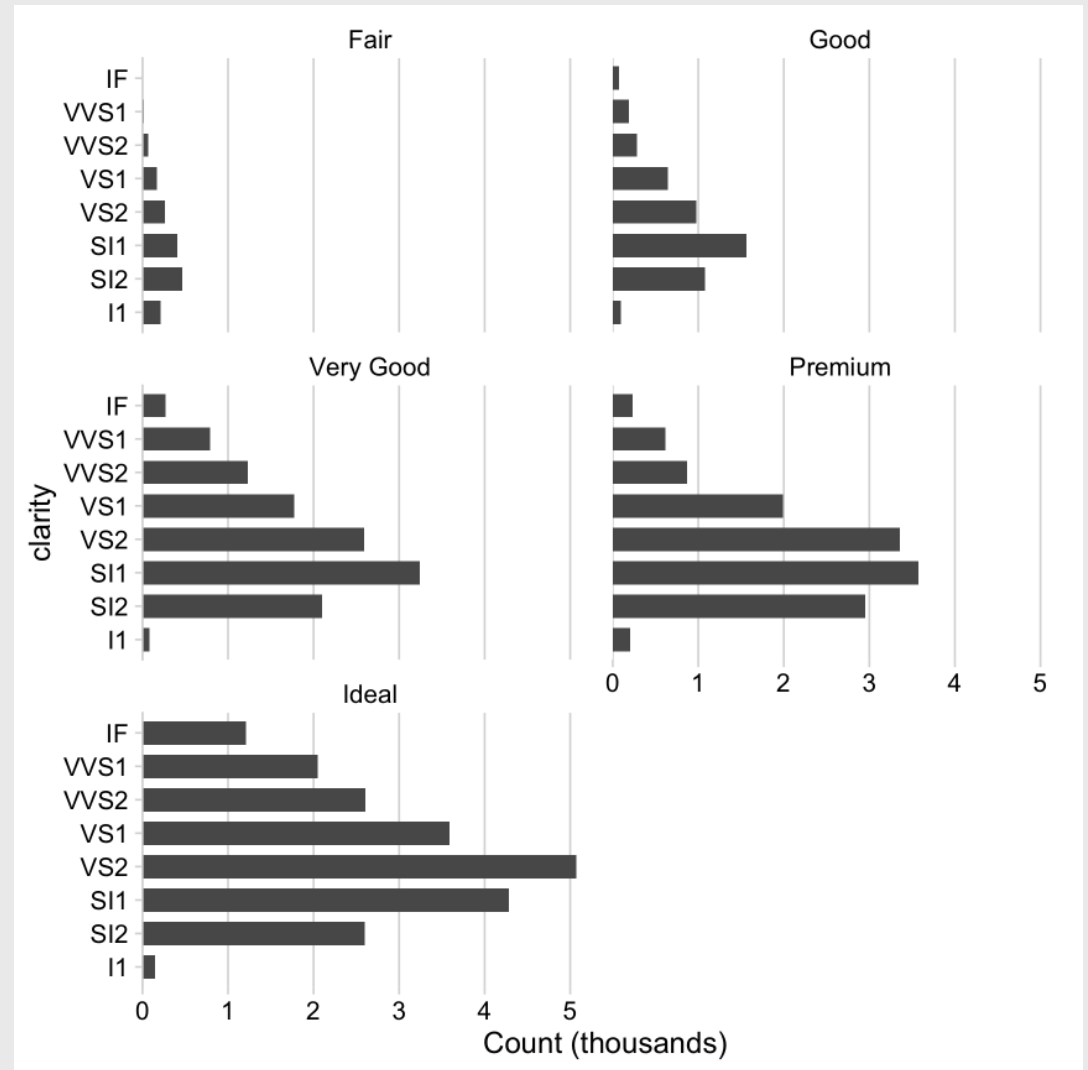
# Consider facets for **comparing across categories**

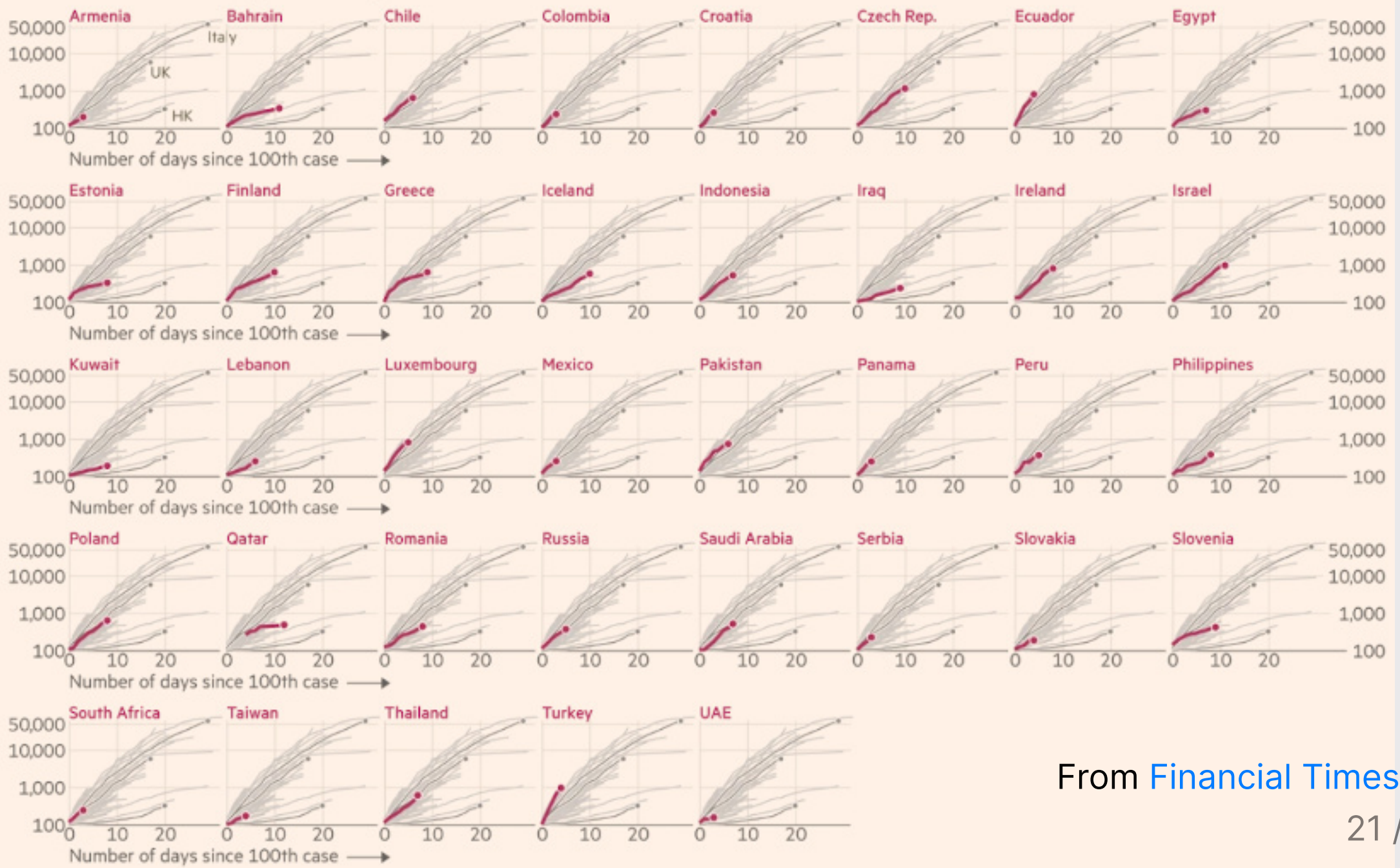
```
diamonds %>%  
  count(clarity, cut) %>%  
  ggplot() +  
  geom_col(aes(x = clarity, y = n),  
            width = 0.7) +  
  facet_wrap(vars(cut), nrow = 1) +  
  scale_y_continuous(  
    expand = expansion(mult = c(0, 0.05))) +  
  theme_minimal_hgrid(font_size = 16)
```



# Consider facets for **comparing across categories**

```
diamonds %>%  
  count(clarity, cut) %>%  
  mutate(n = n / 1000) %>%  
  ggplot() +  
  geom_col(aes(x = clarity, y = n),  
           width = 0.7) +  
  facet_wrap(vars(cut), ncol = 2) +  
  coord_flip() +  
  scale_y_continuous(  
    expand = expansion(mult = c(0, 0.05))) +  
  theme_minimal_vgrid(font_size = 16) +  
  labs(y = "Count (thousands)")
```

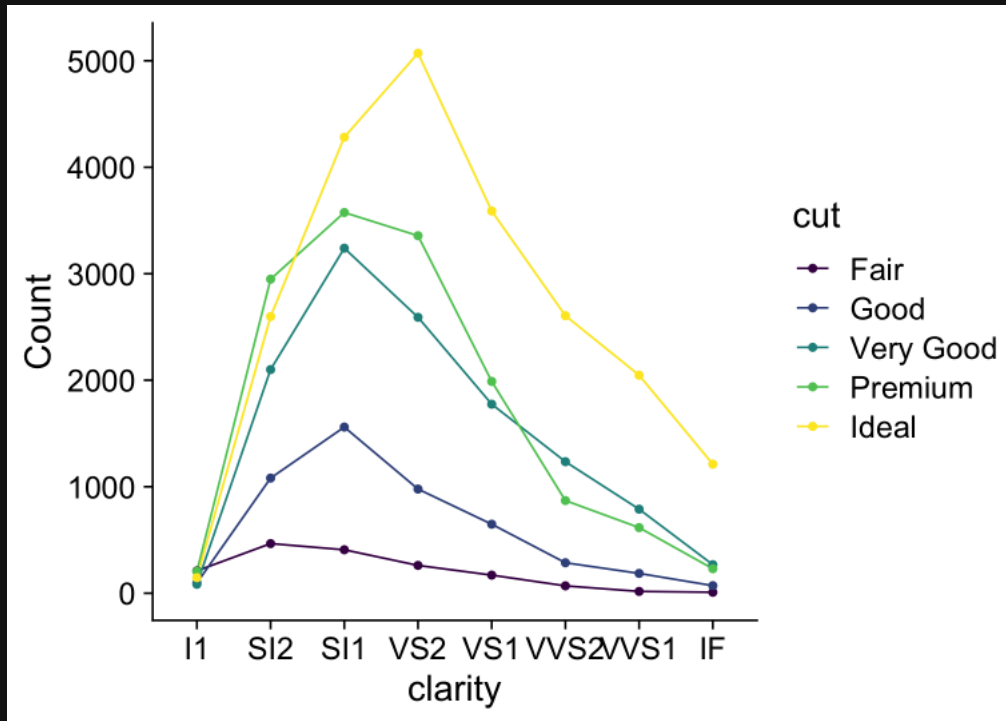




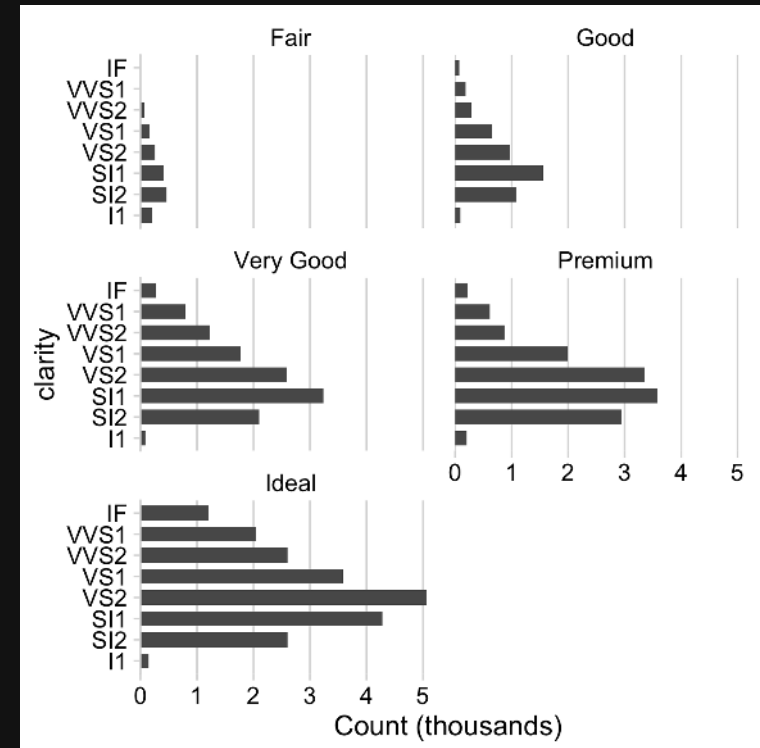
From [Financial Times](#)

# When comparing across multiple categories, consider:

## Parallel coordinates charts



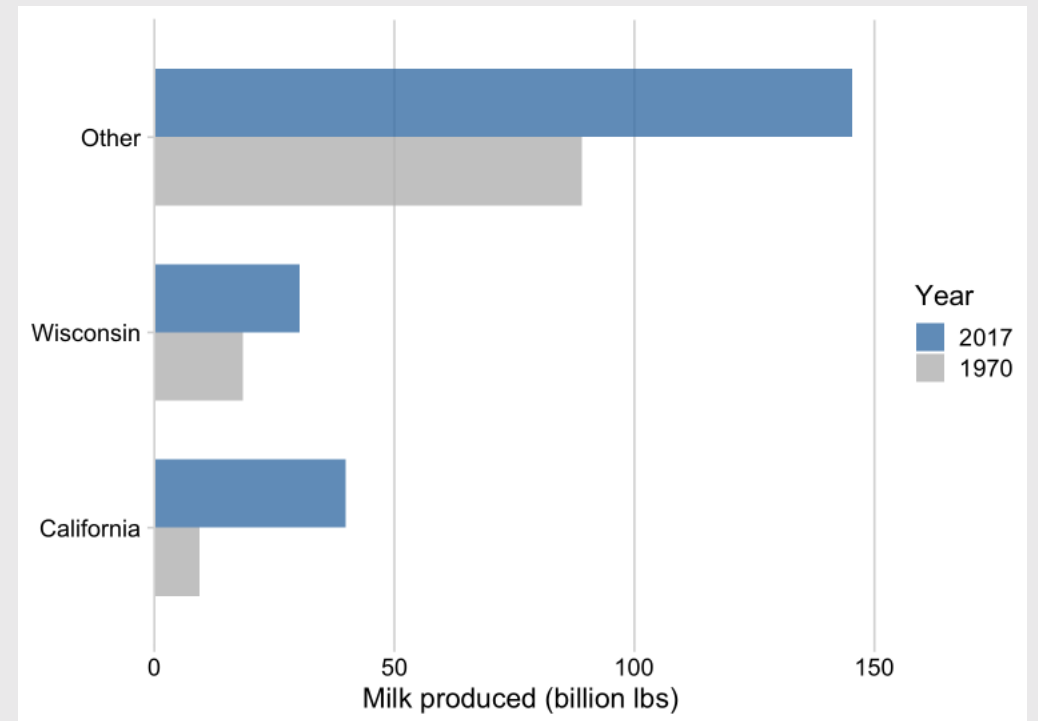
## Faceting



# When comparing **only 2** things, dodged bars are a good starting point

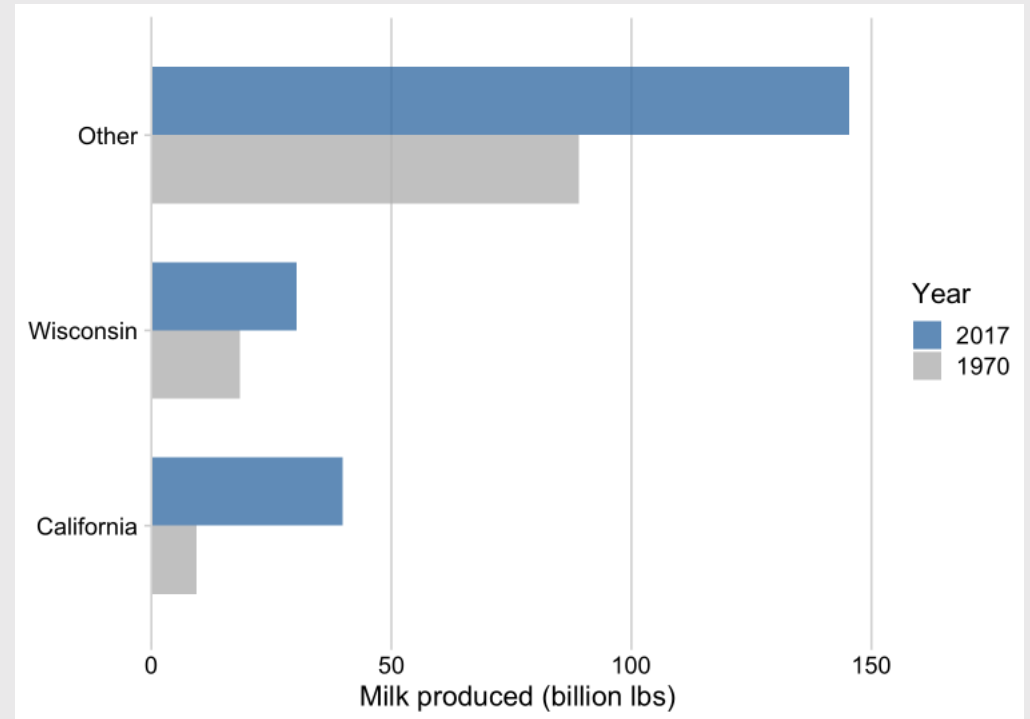
```
milk_compare <- milk_production %>%  
  filter(year %in% c(1970, 2017)) %>%  
  mutate(state = fct_other(state,  
    keep = c('California', 'Wisconsin')))) %>%  
  group_by(year, state) %>%  
  summarise(  
    milk_produced = sum(milk_produced) / 10^9)
```

```
#> # A tibble: 6 × 3  
#> # Groups:   year [2]  
#>   year state      milk_produced  
#>   <dbl> <fct>      <dbl>  
#> 1  1970 California    9.46  
#> 2  1970 Wisconsin   18.4  
#> 3  1970 Other      89.1  
#> 4  2017 California   39.8  
#> 5  2017 Wisconsin   30.3  
#> 6  2017 Other     145.
```



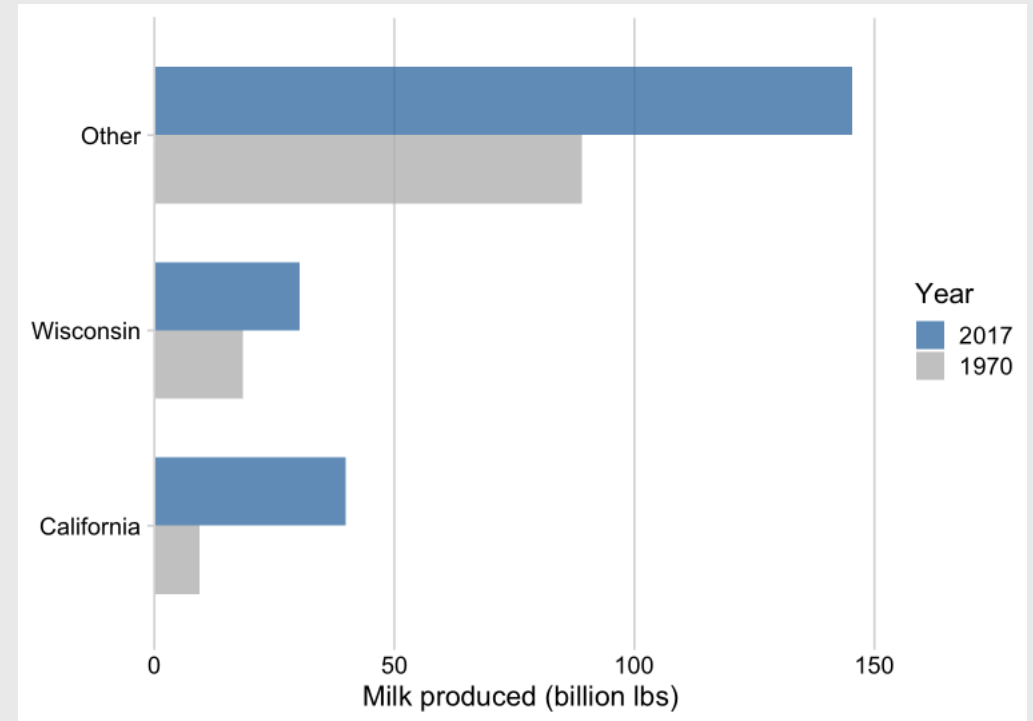
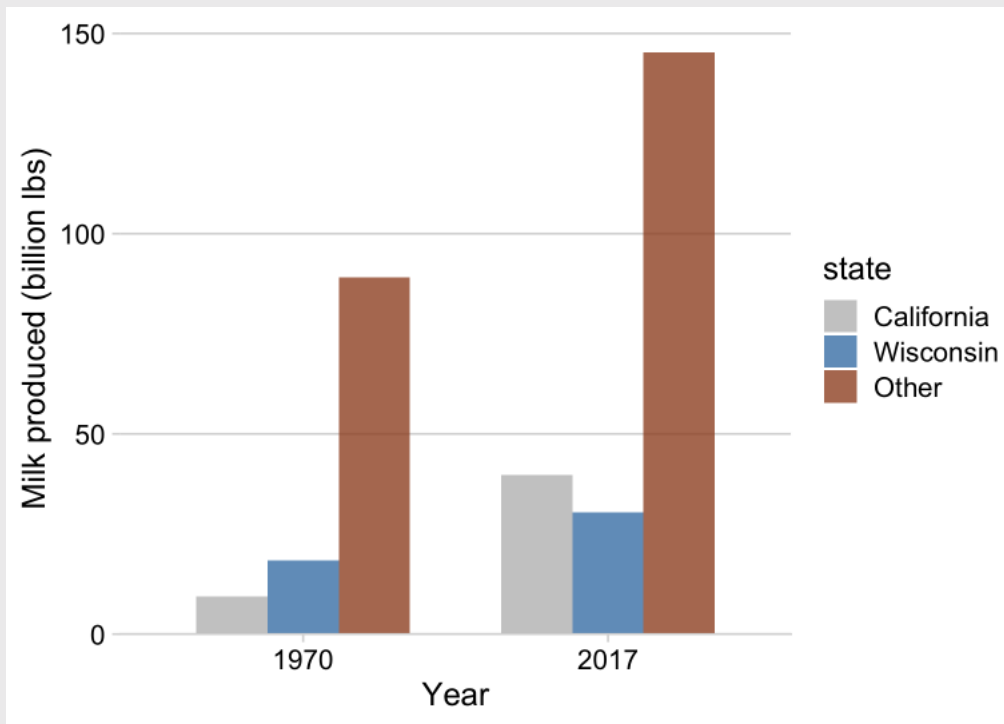
# When comparing **only 2** things, dodged bars are a good starting point

```
ggplot(milk_compare) +  
  geom_col(  
    aes(x = milk_produced, y = state,  
        fill = as.factor(year)),  
    width = 0.7, alpha = 0.8,  
    position = 'dodge') +  
  scale_fill_manual(  
    values = c('grey', 'steelblue'),  
    guide = guide_legend(reverse = TRUE)) +  
  scale_x_continuous(  
    expand = expansion(mult = c(0, 0.05))) +  
  theme_minimal_vgrid() +  
  labs(  
    x = 'Milk produced (billion lbs)',  
    y = NULL,  
    fill = 'Year')
```

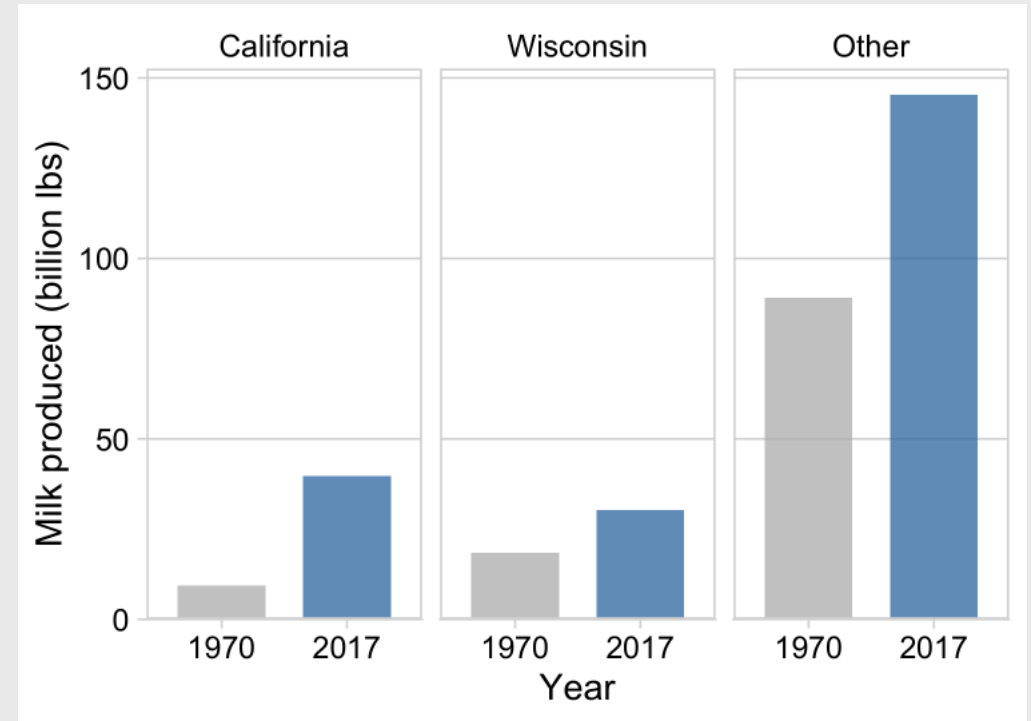
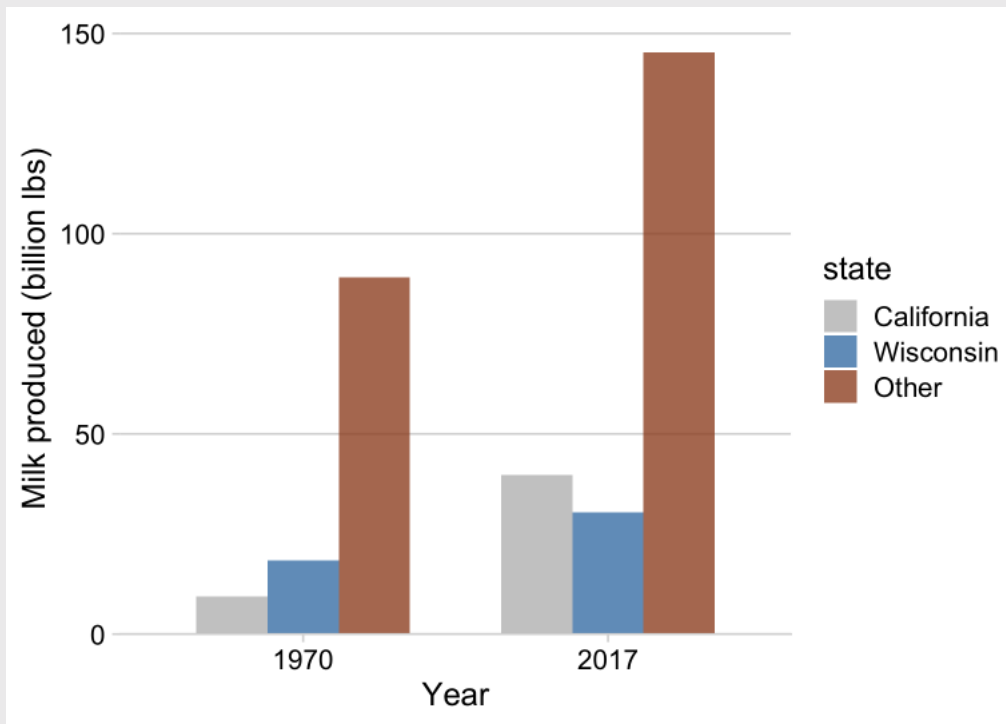




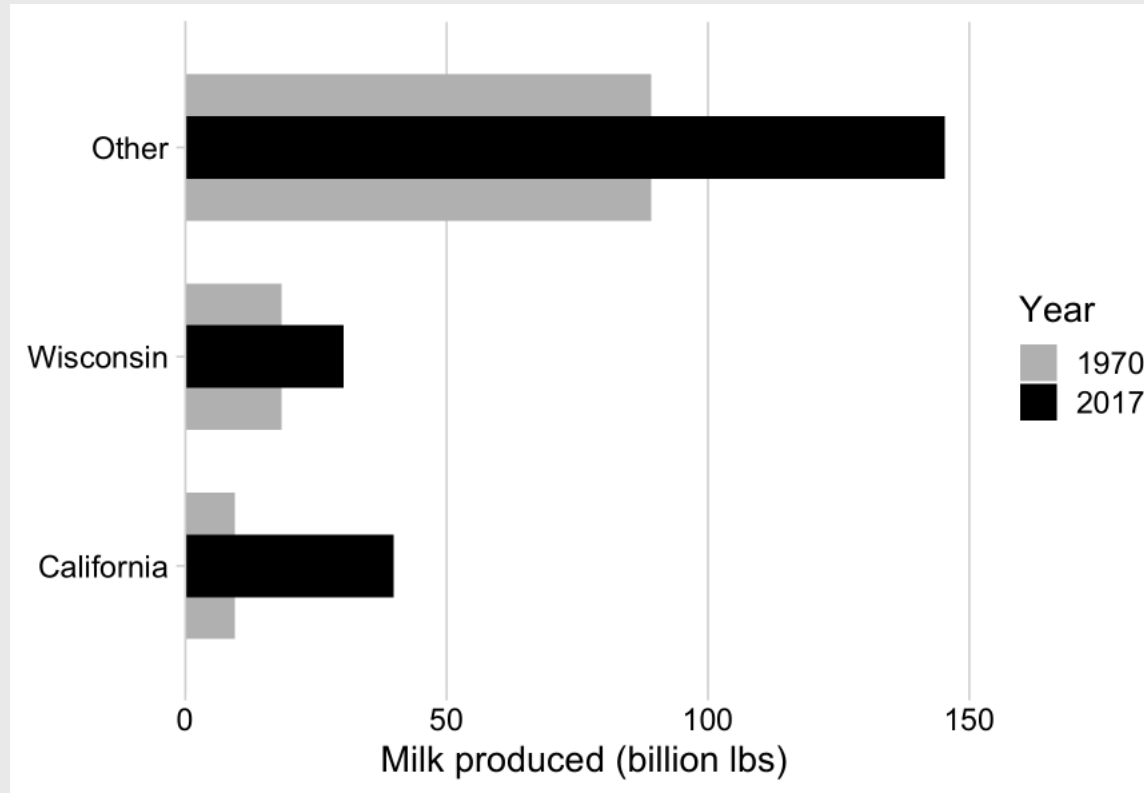
# Avoid putting >2 categories in legend (if possible)



# Or use facets to get rid of the legend!

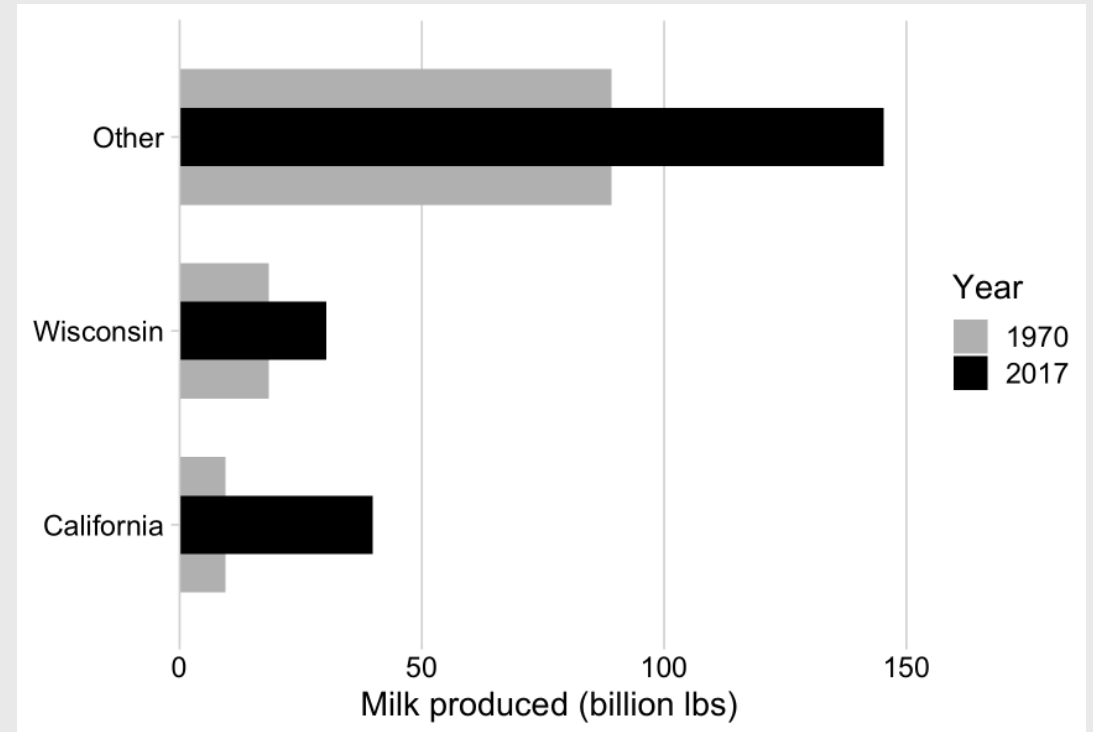


"Bullet" charts are also effective for comparing **2** things



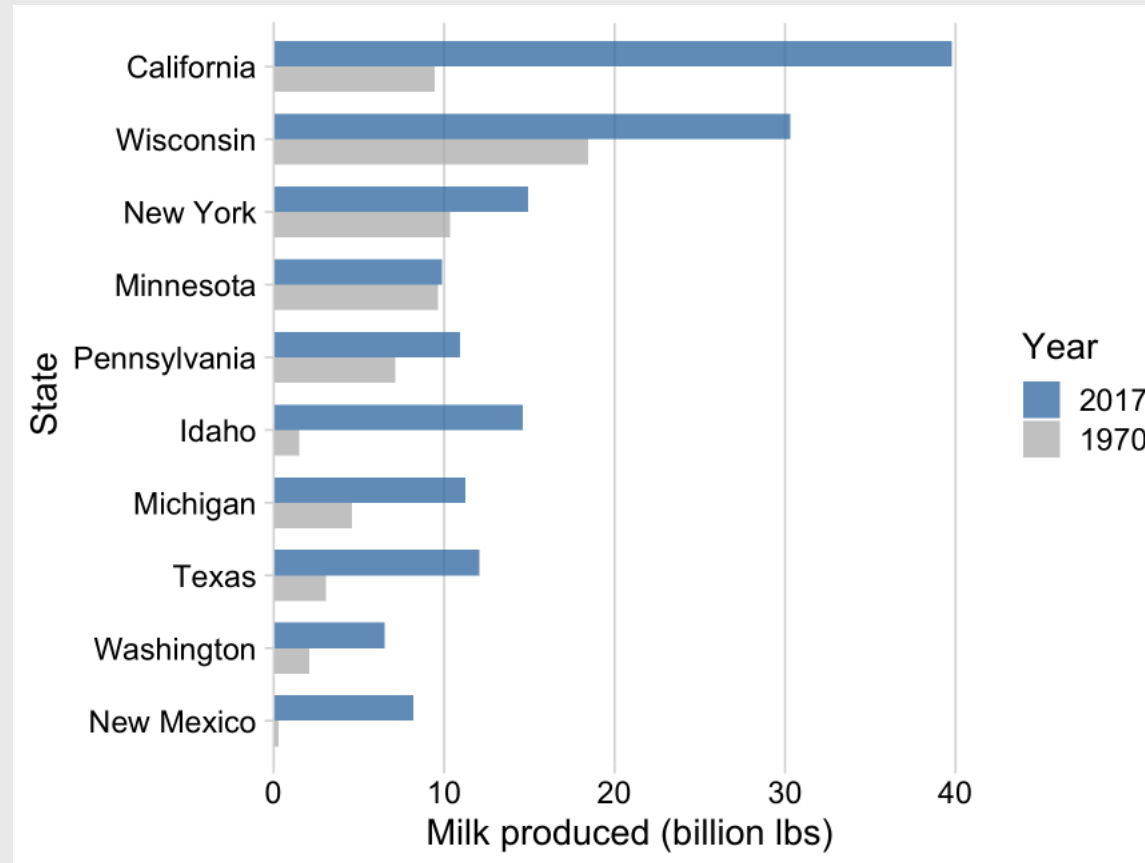
# How to make a "bullet" chart

```
milk_compare %>%  
  pivot_wider(  
    names_from = year,  
    values_from = milk_produced)  
ggplot() +  
  geom_col(  
    aes(x = `1970`, y = state, fill = '1970'),  
    width = 0.7) +  
  geom_col(  
    aes(x = `2017`, y = state, fill = '2017'),  
    width = 0.3) +  
  scale_fill_manual(  
    values = c('grey', 'black')) +  
  scale_x_continuous(  
    expand = expansion(mult = c(0, 0.05))) +  
  theme_minimal_vgrid(font_size = 18) +  
  labs(  
    x = 'Milk produced (billion lbs)',  
    y = NULL,  
    fill = "Year")
```



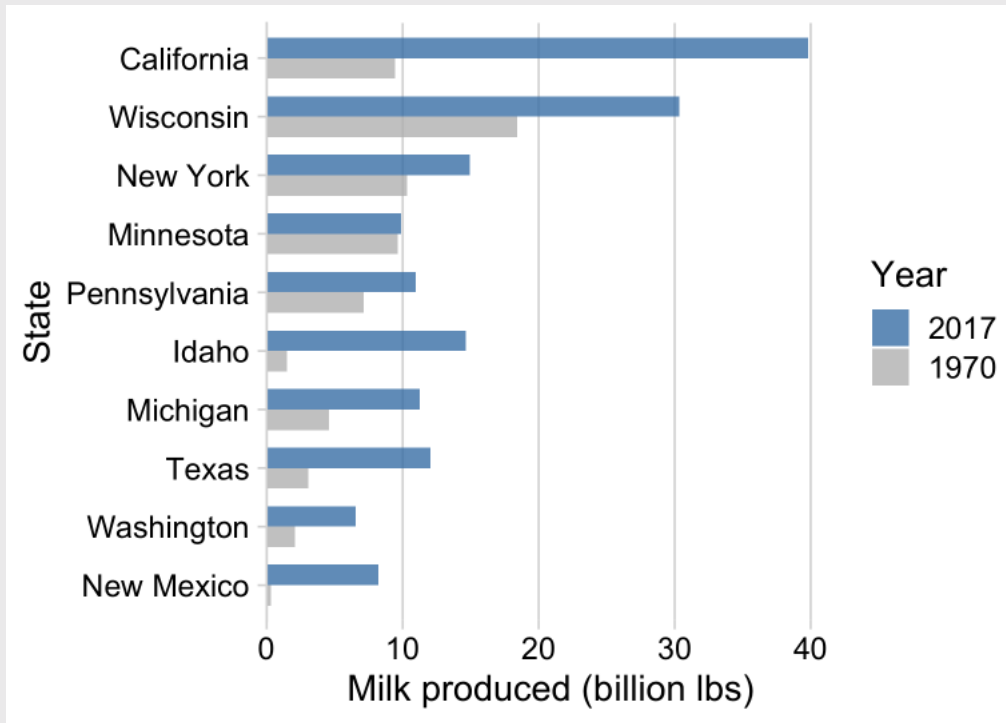
# With **more than 2** things, dodged bars can get confusing

Still comparing 2 time periods, but across **10** categories

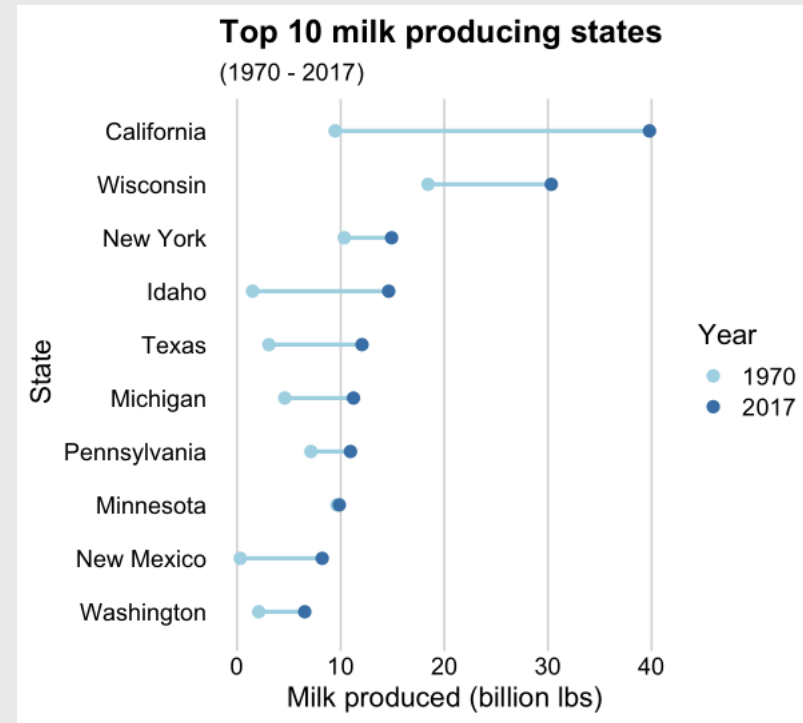


# Strategies for comparing 2 things across **more than 2 categories**

## Dodged bars 🙄

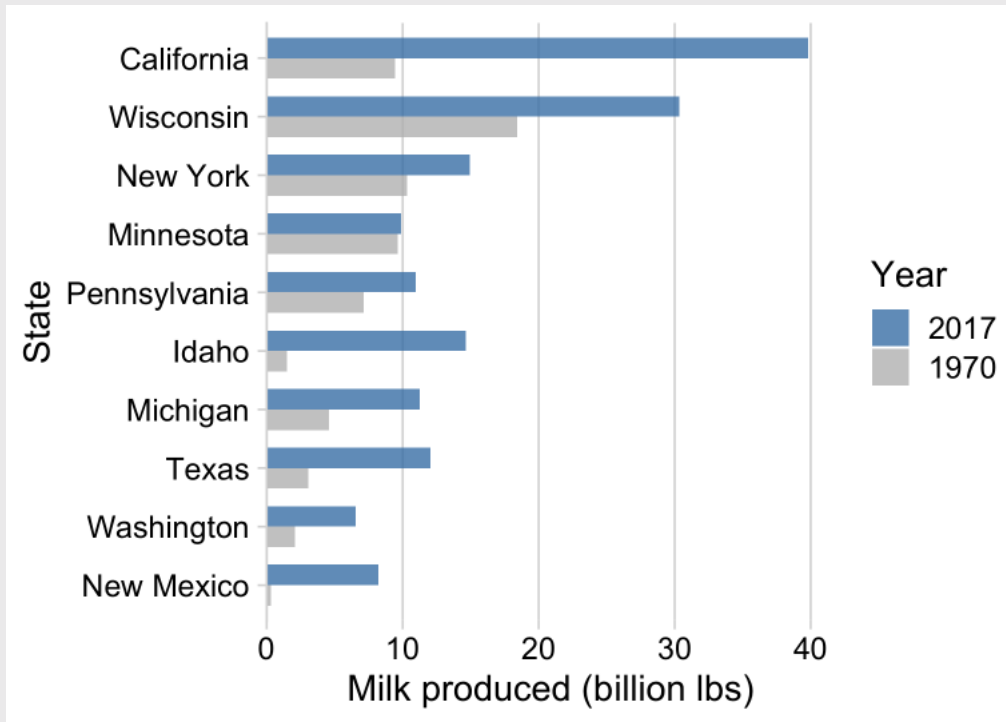


## Dumbbell bars 😄

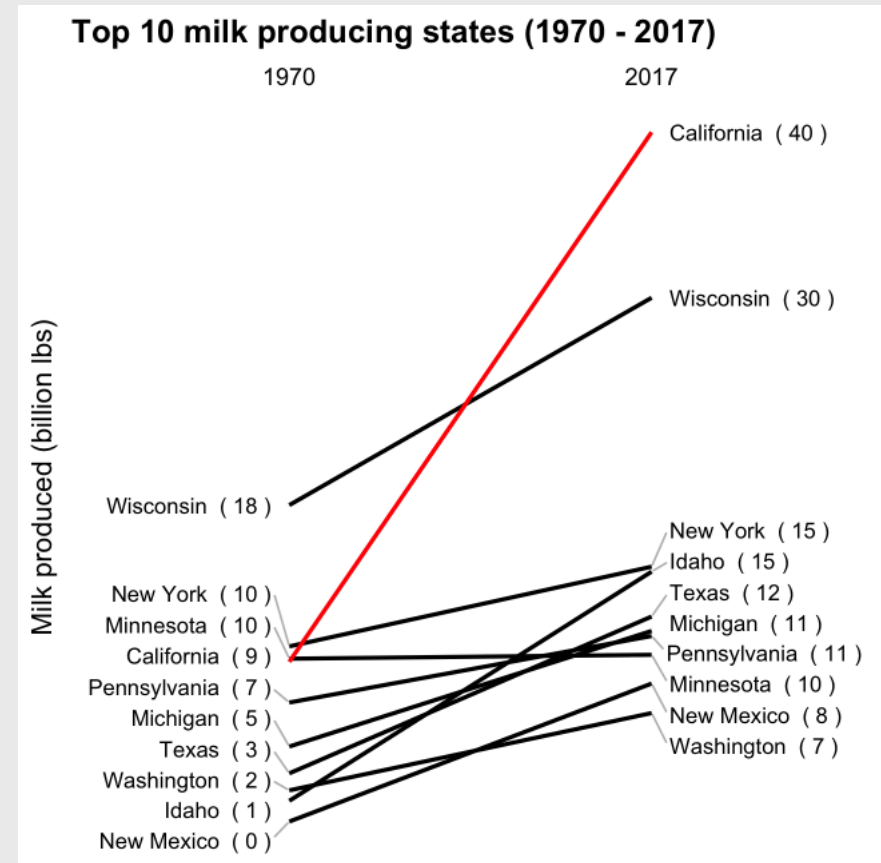


# Strategies for comparing 2 things across **more than 2 categories**

## Dodged bars 🙄

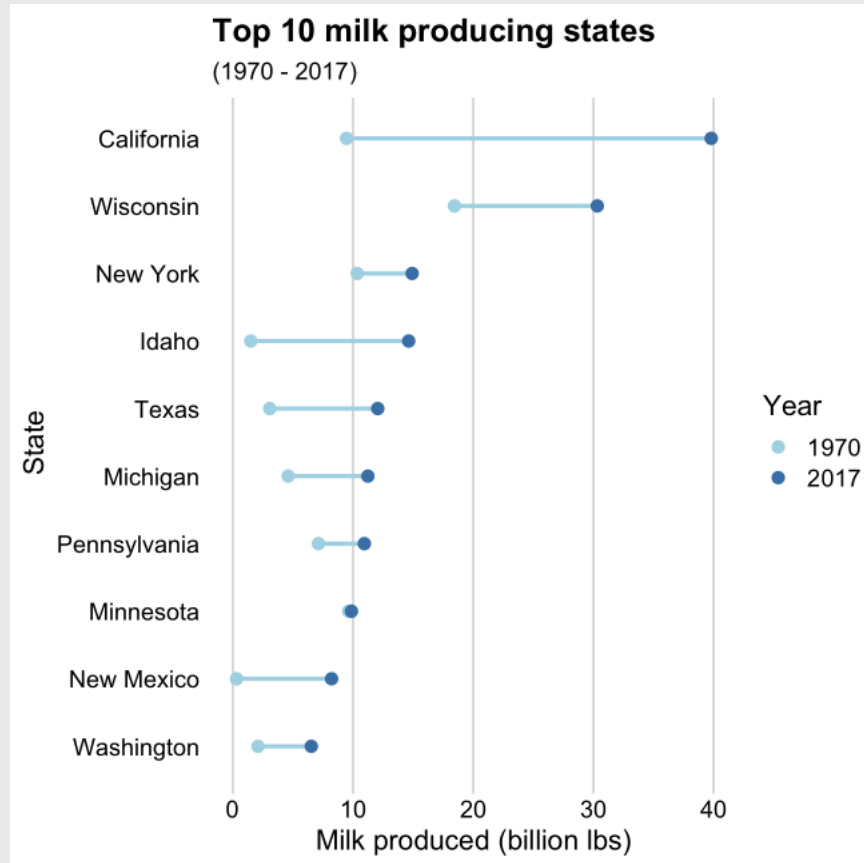


## Slope chart 😊



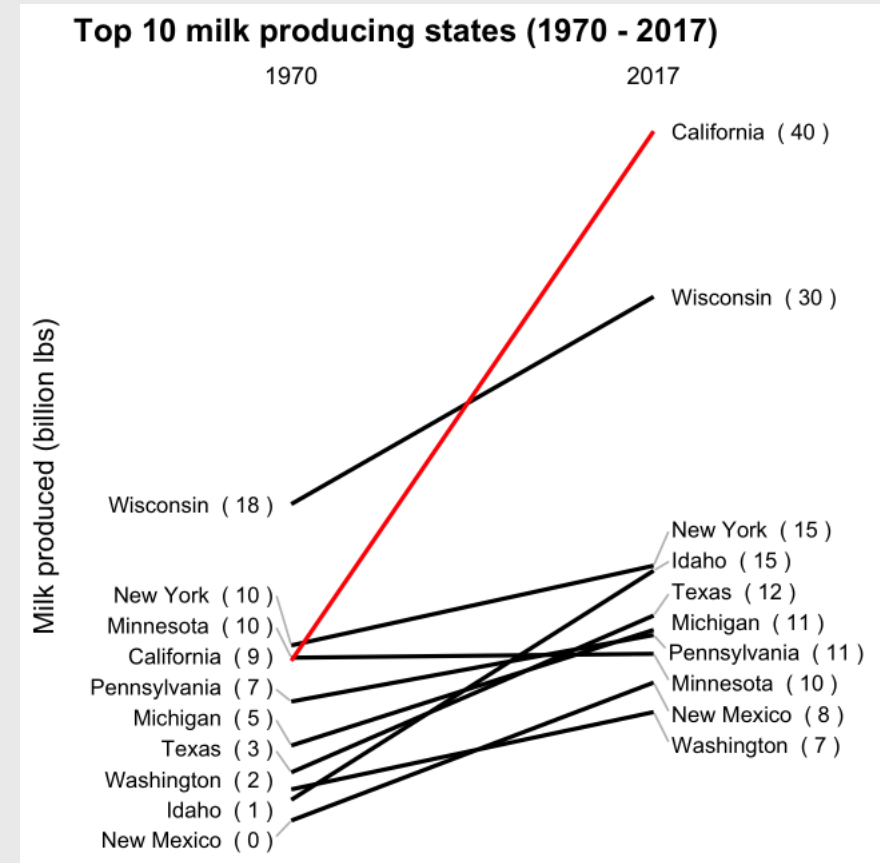
## Dumbbell charts highlight:

- Compare **magnitudes** across two periods / groups



## Slope charts highlight:

- *Change* in **rankings**
- Highlight individual categories

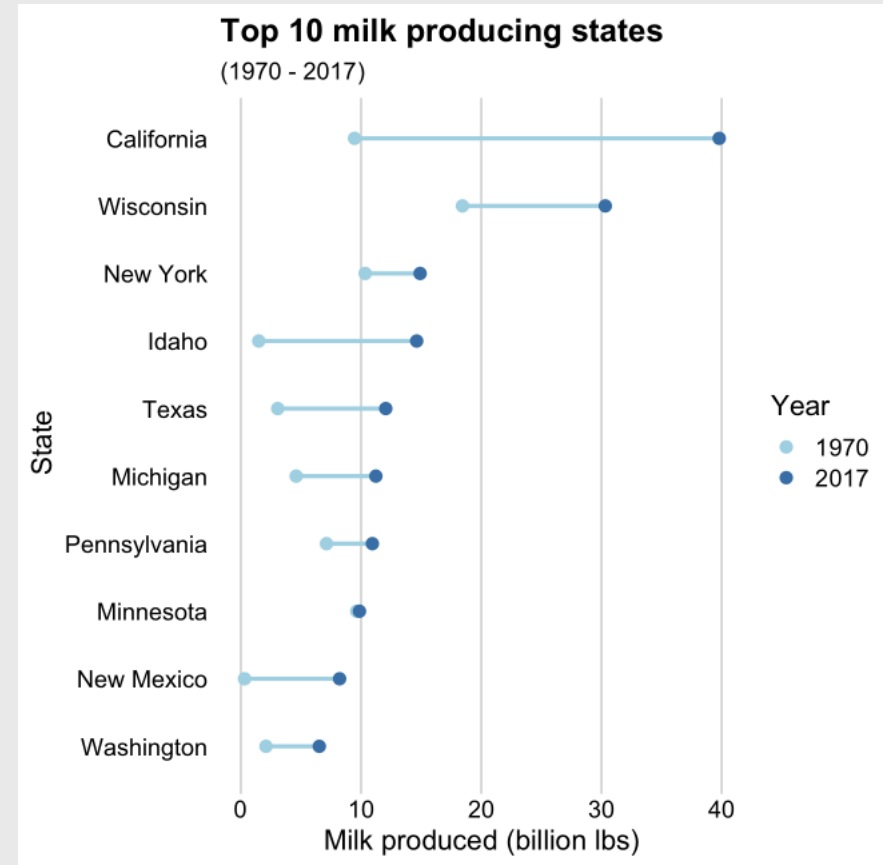




# How to make a **Dumbbell chart**

Create data frame for plotting

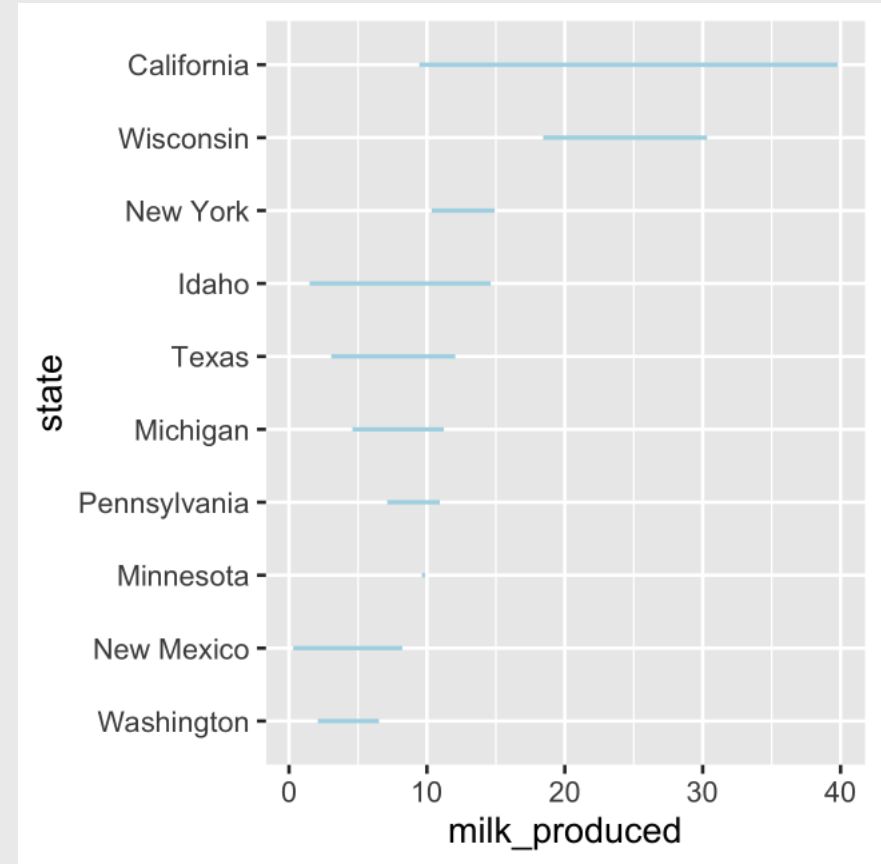
```
top10states <- milk_production %>%  
  filter(year == 2017) %>%  
  arrange(desc(milk_produced)) %>%  
  slice(1:10)  
  
milk_summary_dumbbell <- milk_production %>%  
  filter(  
    year %in% c(1970, 2017),  
    state %in% top10states$state) %>%  
  mutate(  
    # Reorder state variables  
    state = fct_reorder2(state,  
      year, desc(milk_produced)),  
    # Convert year to discrete variable  
    year = as.factor(year),  
    # Modify units  
    milk_produced = milk_produced / 10^9)
```



# How to make a **Dumbbell chart**

Make lines (note the `group` variable)

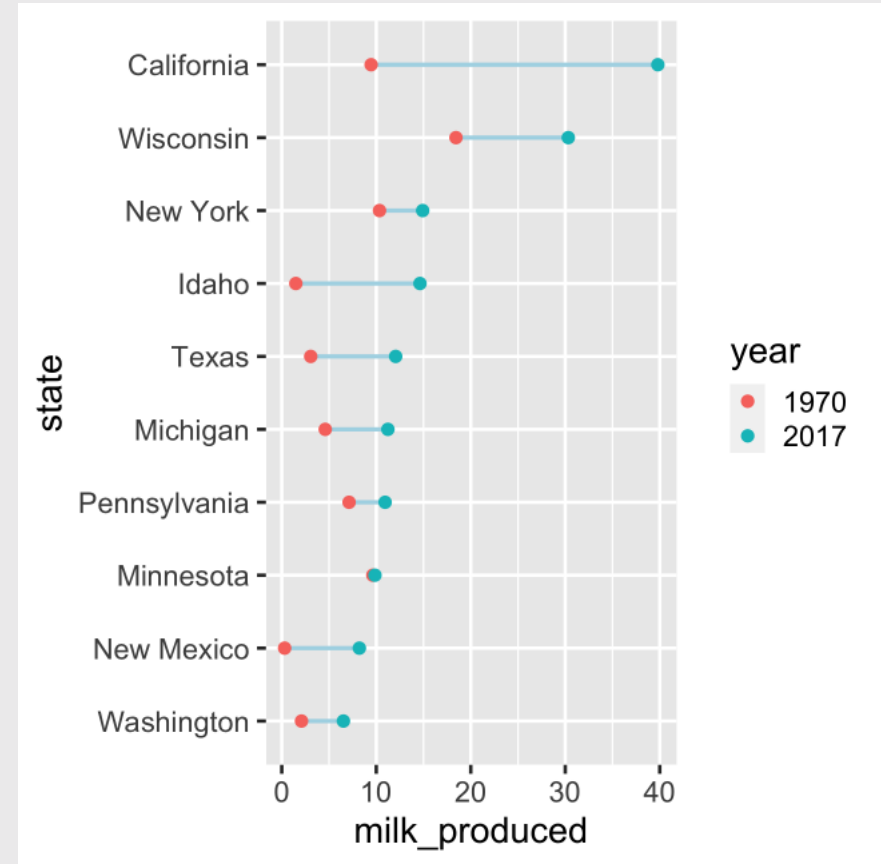
```
ggplot(milk_summary_dumbbell,  
       aes(x = milk_produced, y = state)) +  
  geom_line(aes(group = state),  
           color = 'lightblue', size = 1)
```



# How to make a **Dumbbell chart**

Add points (note the `color` variable)

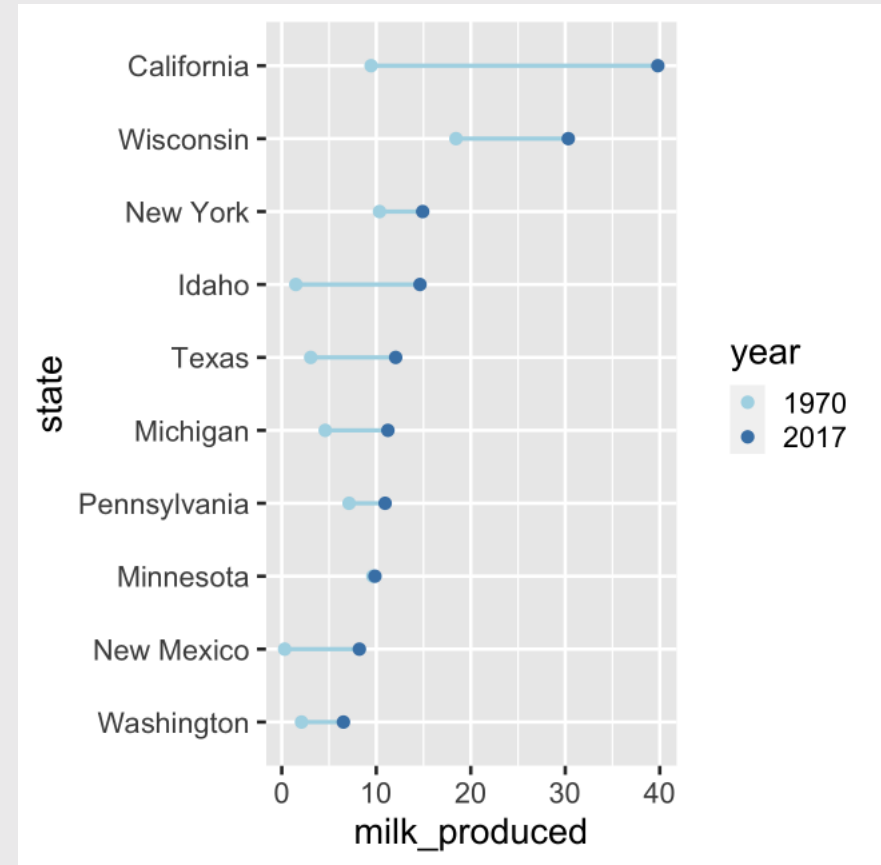
```
ggplot(milk_summary_dumbbell,  
       aes(x = milk_produced, y = state)) +  
  geom_line(aes(group = state),  
           color = 'lightblue', size = 1) +  
  geom_point(aes(color = year), size = 2.5)
```



# How to make a **Dumbbell chart**

Change the colors:

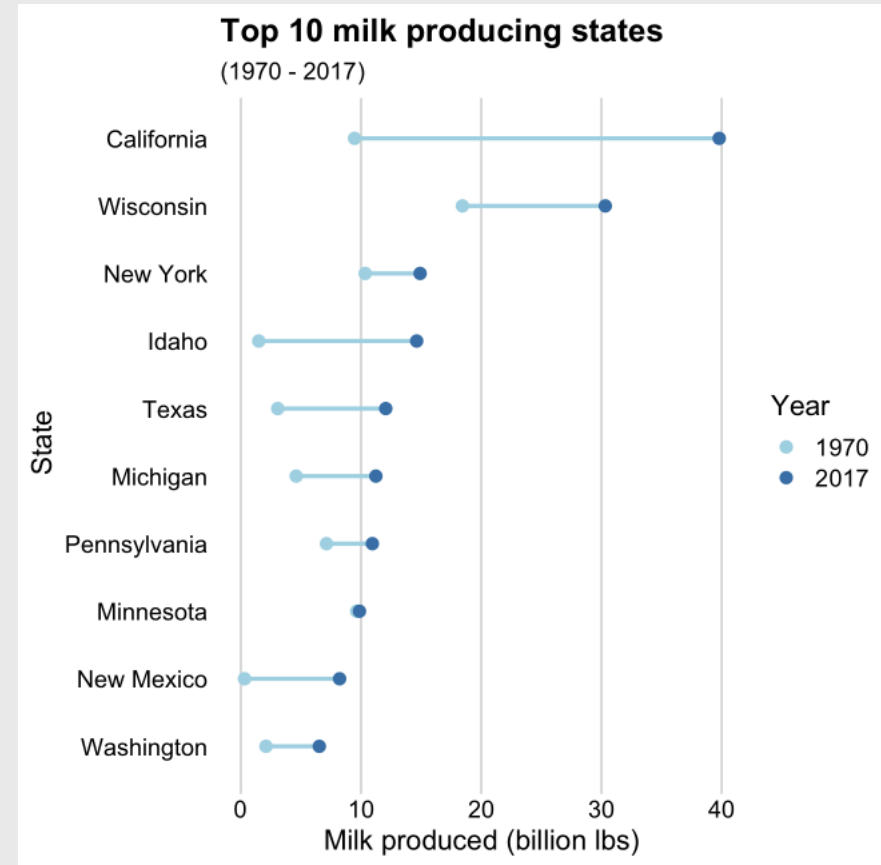
```
ggplot(milk_summary_dumbbell,  
       aes(x = milk_produced, y = state)) +  
  geom_line(aes(group = state),  
            color = 'lightblue', size = 1) +  
  geom_point(aes(color = year), size = 2.5) +  
  scale_color_manual(  
    values = c('lightblue', 'steelblue'))
```



# How to make a **Dumbbell chart**

Adjust the theme and annotate

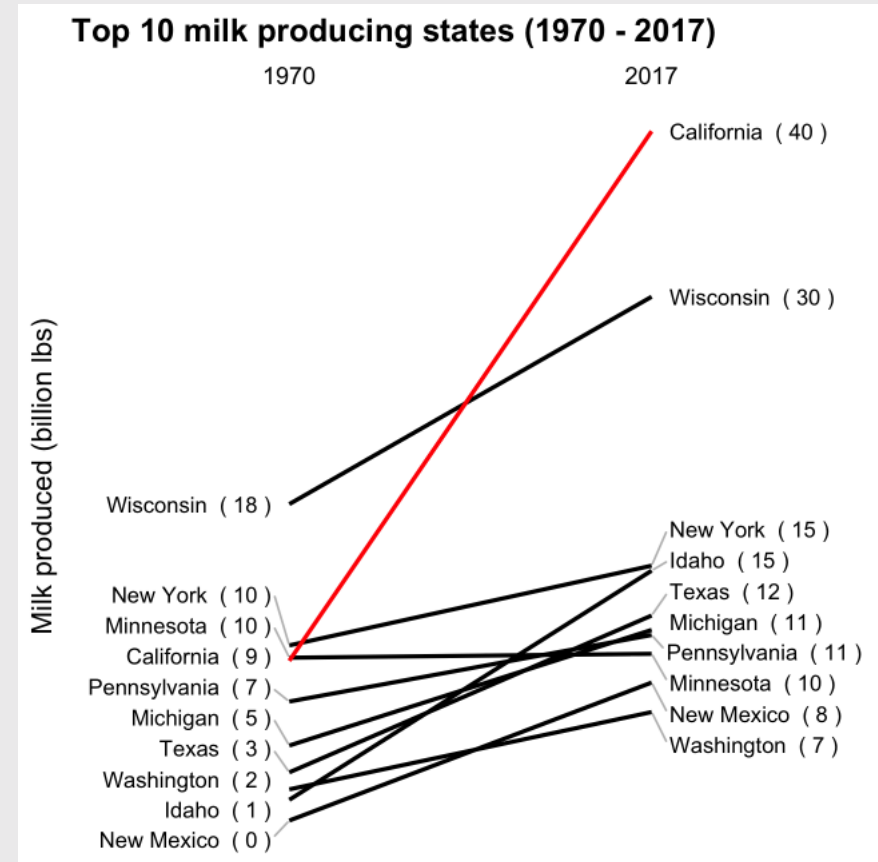
```
ggplot(milk_summary_dumbbell,  
       aes(x = milk_produced, y = state)) +  
  geom_line(aes(group = state),  
            color = 'lightblue', size = 1) +  
  geom_point(aes(color = year), size = 2.5) +  
  scale_color_manual(  
    values = c('lightblue', 'steelblue')) +  
  theme_minimal_vgrid() +  
  # Remove y axis line and tick marks  
  theme(  
    axis.line.y = element_blank(),  
    axis.ticks.y = element_blank()) +  
  labs(x = 'Milk produced (billion lbs)',  
       y = 'State',  
       color = 'Year',  
       title = 'Top 10 milk producing states',  
       subtitle = '(1970 - 2017)')
```



## Create data frame for plotting

```
top10states <- milk_production %>%  
  filter(year == 2017) %>%  
  arrange(desc(milk_produced)) %>%  
  slice(1:10)  
  
milk_summary_slope <- milk_production %>%  
  filter(  
    year %in% c(1970, 2017),  
    state %in% top10states$state) %>%  
  mutate(  
    # Reorder state variables  
    state = fct_reorder2(state,  
      year, desc(milk_produced)),  
    # Convert year to discrete variable  
    year = as.factor(year),  
    # Modify units  
    milk_produced = milk_produced / 10^9,  
    # Define line color  
    lineColor = if_else(  
      state == 'California', 'CA', 'other'),  
    # Make labels  
    label = paste(state, ' (',  
      round(milk_produced), ')'),  
    label_left = ifelse(year == 1970, label, NA),  
    label_right = ifelse(year == 2017, label, NA))
```

## How to make a **Slope chart**



## Create data frame for plotting

```
top10states <- milk_production %>%
  filter(year == 2017) %>%
  arrange(desc(milk_produced)) %>%
  slice(1:10)

milk_summary_slope <- milk_production %>%
  filter(
    year %in% c(1970, 2017),
    state %in% top10states$state) %>%
  mutate(
    # Reorder state variables
    state = fct_reorder2(state,
      year, desc(milk_produced)),
    # Convert year to discrete variable
    year = as.factor(year),
    # Modify units
    milk_produced = milk_produced / 10^9,
    # Define line color
    lineColor = if_else(
      state == 'California', 'CA', 'other'),
    # Make labels
    label = paste(state, ' (',
      round(milk_produced), ')'),
    label_left = ifelse(year == 1970, label, NA),
    label_right = ifelse(year == 2017, label, NA))
```

## How to make a Slope chart

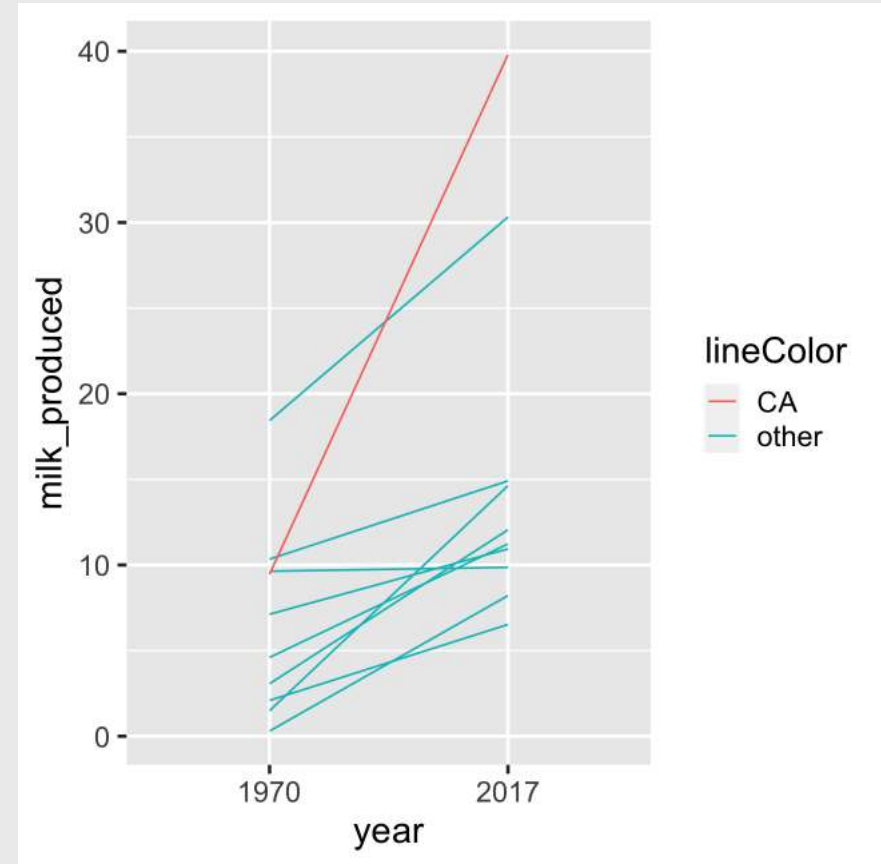
```
milk_summary_slope %>%
  select(state, year, milk_produced, label, lineColor)
```

```
#> # A tibble: 20 × 5
#>   state      year milk_produced label                lineCo
#>   <fct>      <fct>      <dbl> <chr>                <chr>
#> 1 New York   1970         10.3 New York ( 10 )      other
#> 2 Pennsylvania 1970         7.12 Pennsylvania ( 7 ) other
#> 3 Michigan   1970         4.60 Michigan ( 5 )     other
#> 4 Wisconsin 1970        18.4 Wisconsin ( 18 )   other
#> 5 Minnesota  1970         9.64 Minnesota ( 10 )   other
#> 6 Texas      1970         3.06 Texas ( 3 )        other
#> 7 Idaho      1970         1.49 Idaho ( 1 )        other
#> 8 New Mexico 1970         0.304 New Mexico ( 0 )   other
#> 9 Washington 1970         2.09 Washington ( 2 )   other
#> 10 California 1970         9.46 California ( 9 )   CA
#> 11 New York   2017        14.9 New York ( 15 )    other
#> 12 Pennsylvania 2017        10.9 Pennsylvania ( 11 ) other
#> 13 Michigan   2017        11.2 Michigan ( 11 )    other
#> 14 Wisconsin  2017        30.3 Wisconsin ( 30 )   other
#> 15 Minnesota  2017         9.86 Minnesota ( 10 )   other
#> 16 Texas      2017        12.1 Texas ( 12 )       other
#> 17 Idaho      2017        14.6 Idaho ( 15 )       other
#> 18 New Mexico 2017         8.21 New Mexico ( 8 )   other
#> 19 Washington 2017         6.53 Washington ( 7 )   other
#> 20 California 2017        39.8 California ( 40 )  CA
```

# How to make a **Slope chart**

Start with a line plot - note the **group** variable:

```
ggplot(milk_summary_slope,  
       aes(x = year, y = milk_produced,  
           group = state)) +  
  geom_line(aes(color = lineColor))
```





# How to make a **Slope chart**

Add labels:

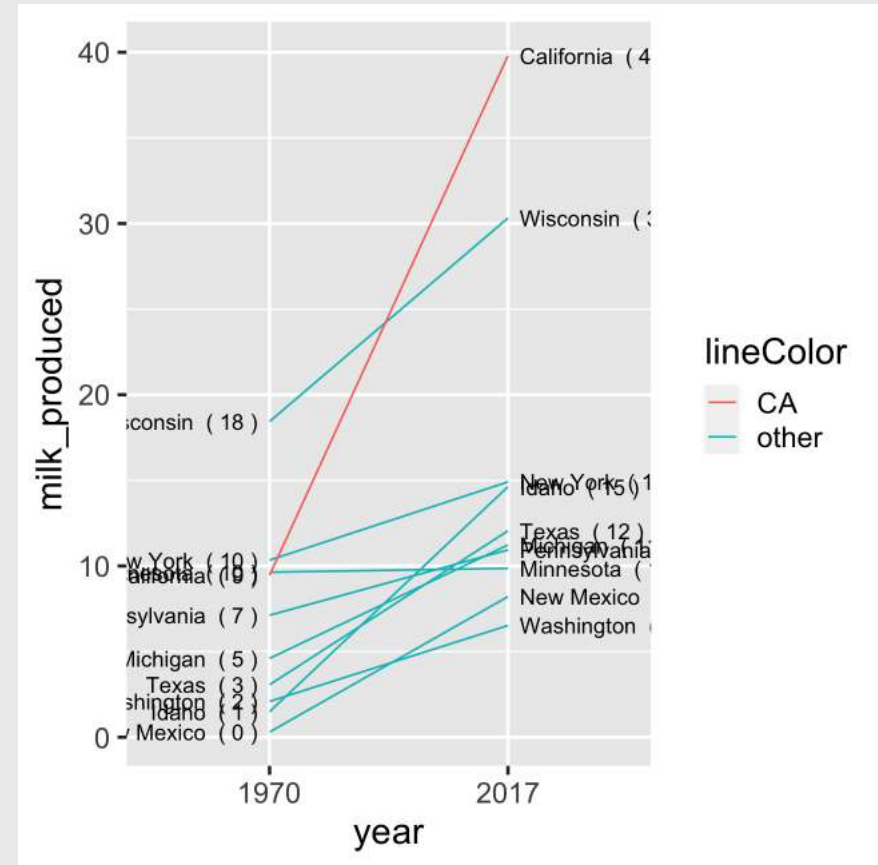
```
ggplot(milk_summary_slope,  
  aes(x = year, y = milk_produced,  
    group = state)) +  
  geom_line(aes(color = lineColor)) +  
  # Add 1970 labels (left side)  
  geom_text(aes(label = label_left),  
    hjust = 1, nudge_x = -0.05) +  
  # Add 2017 labels (right side)  
  geom_text(aes(label = label_right),  
    hjust = 0, nudge_x = 0.05)
```

## Justification **hjust**

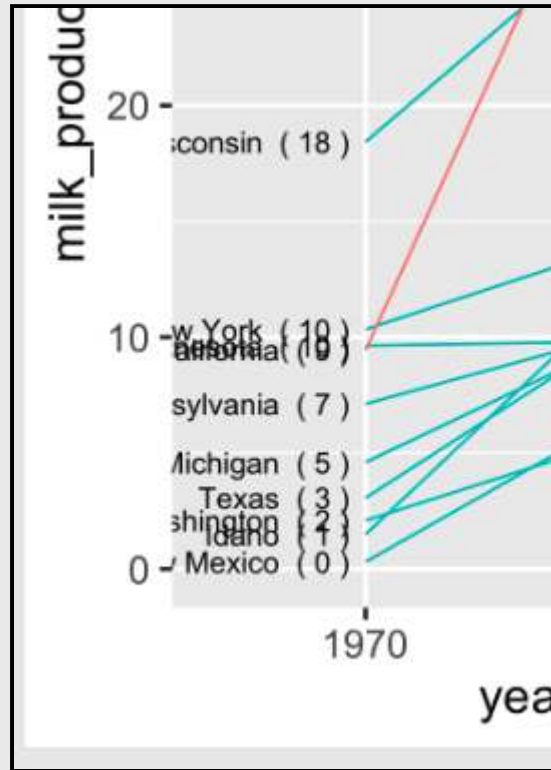
Left 0

Center 0.5

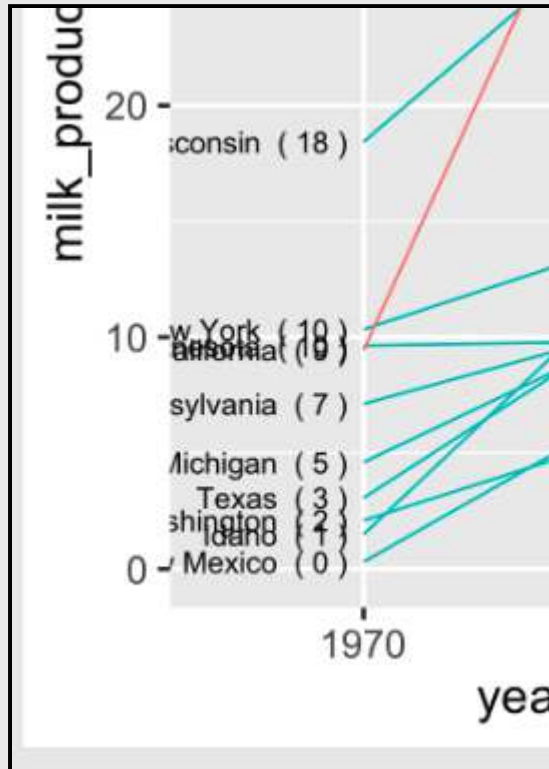
Right 1



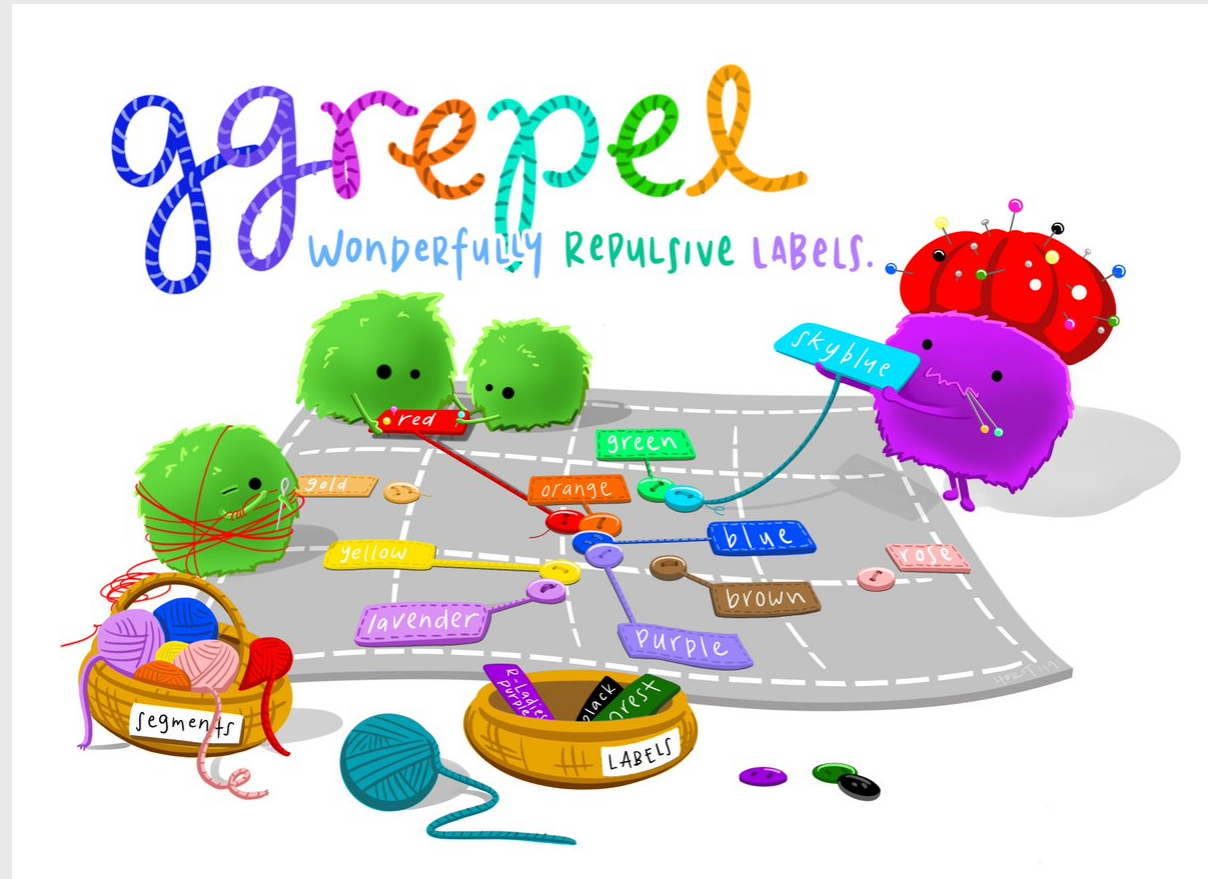
# Overlapping labels?



# Overlapping labels? **ggrepel** library to the rescue!



Artwork by [@allison\\_horst](#)

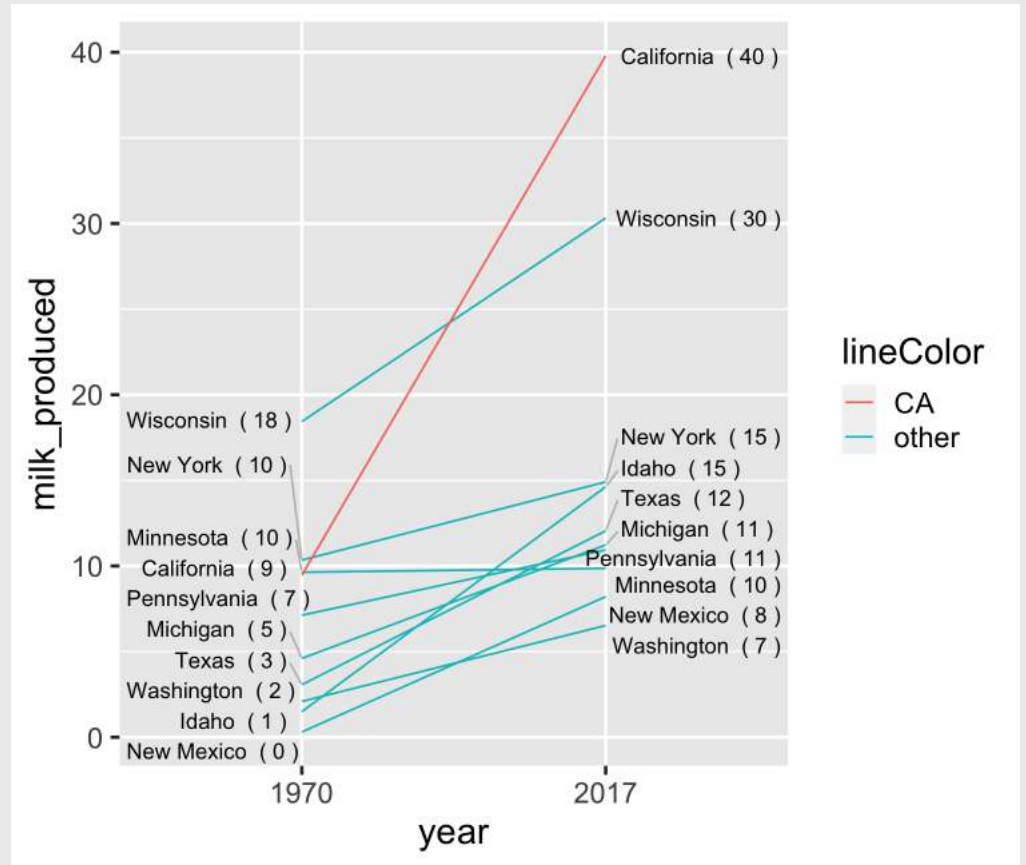


# How to make a **Slope chart**

Align labels so they don't overlap:

```
library(ggrepel)

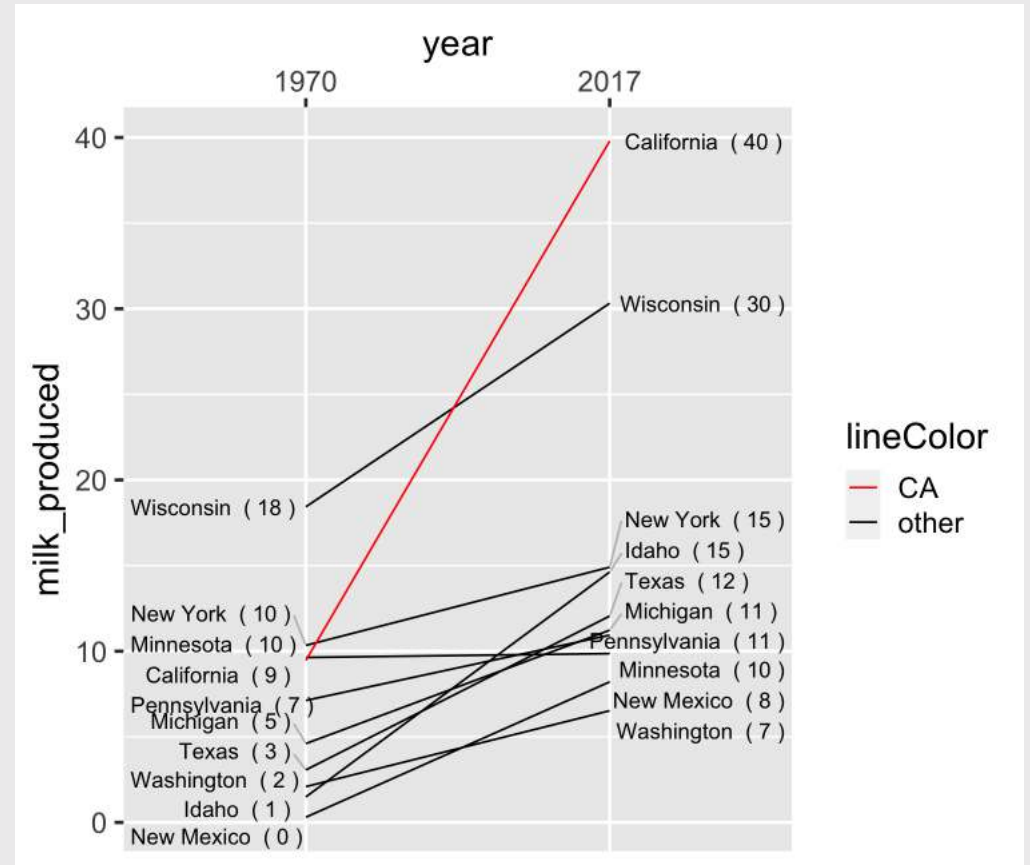
ggplot(milk_summary_slope,
       aes(x = year, y = milk_produced,
           group = state)) +
  geom_line(aes(color = lineColor)) +
  # Add 1970 labels (left side)
  geom_text_repel(
    aes(label = label_left),
    hjust = 1, nudge_x = -0.05,
    direction = 'y', segment.color = 'grey')
  # Add 2017 labels (right side)
  geom_text_repel(
    aes(label = label_right),
    hjust = 0, nudge_x = 0.05,
    direction = 'y', segment.color = 'grey')
```



# How to make a **Slope chart**

Adjust colors:

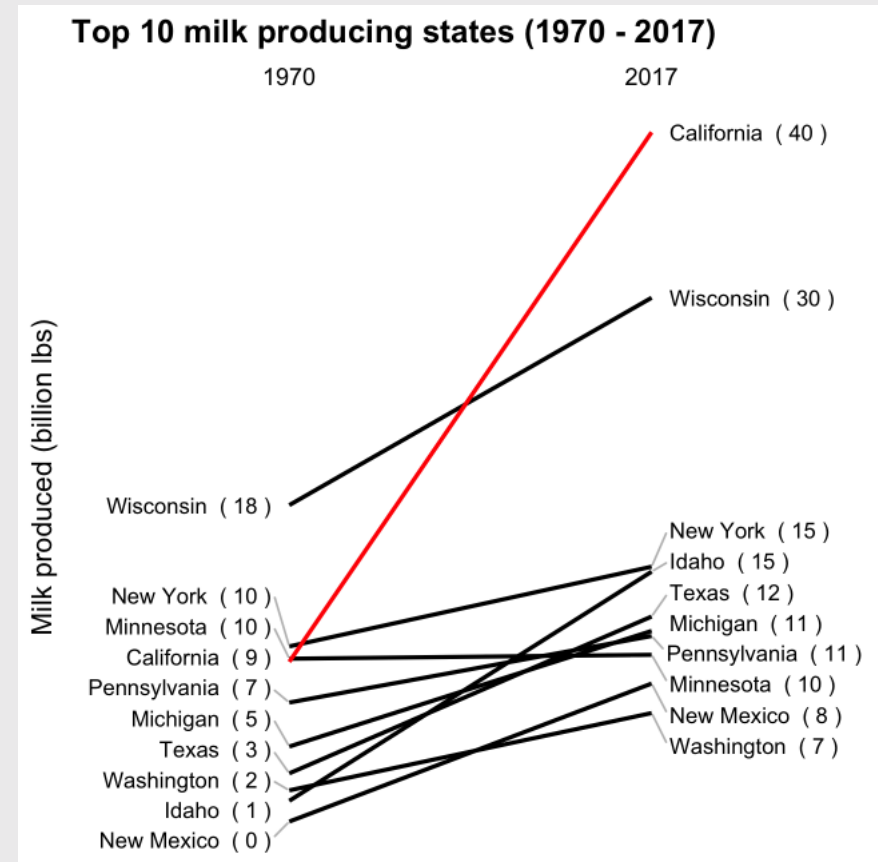
```
ggplot(milk_summary_slope,  
  aes(x = year, y = milk_produced,  
      group = state)) +  
  geom_line(aes(color = lineColor)) +  
  geom_text_repel(  
    aes(label = label_left),  
    hjust = 1, nudge_x = -0.05,  
    direction = 'y', segment.color = 'grey')  
  geom_text_repel(  
    aes(label = label_right),  
    hjust = 0, nudge_x = 0.05,  
    direction = 'y', segment.color = 'grey')  
  # Move year labels to top, modify line color  
  scale_x_discrete(position = 'top') +  
  scale_color_manual(values = c('red', 'black'))
```



## Adjust the theme and annotate

```
ggplot(milk_summary_slope,
  aes(x = year, y = milk_produced,
    group = state)) +
  geom_line(aes(color = lineColor)) +
  # Add 1970 labels (left side)
  geom_text_repel(
    aes(label = label_left),
    hjust = 1, nudge_x = -0.05,
    direction = 'y', segment.color = 'grey') +
  # Add 2017 labels (right side)
  geom_text_repel(aes(label = label_right),
    hjust = 0, nudge_x = 0.05,
    direction = 'y', segment.color = 'grey') +
  # Move year labels to top, modify line colors
  scale_x_discrete(position = 'top') +
  scale_color_manual(values = c('red', 'black')) +
  # Annotate & adjust theme
  labs(x = NULL,
    y = 'Milk produced (billion lbs)',
    title = 'Top 10 milk producing states (1970 -
  theme_minimal_grid() +
  theme(panel.grid = element_blank(),
    axis.text.y = element_blank(),
    axis.ticks = element_blank(),
    legend.position = 'none')
```

## How to make a **Slope chart**



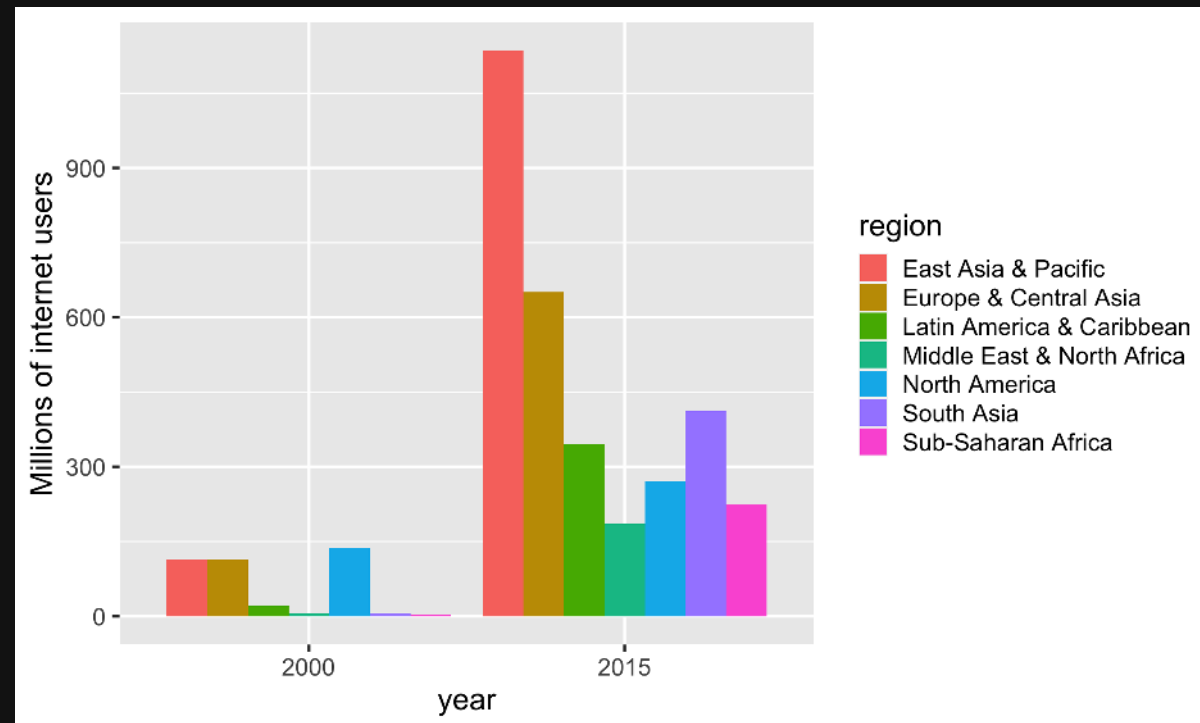
# Your turn - comparing multiple categories

20:00

Using the `internet_regions` data frame, pick a strategy and create an improved version of this chart.

Strategies:

- Dodged bars
- Facets
- Bullet chart
- Dumbbell chart
- Slope chart



# Break!

Stand up, Move around, Stretch!

05:00



# Week 8: *Comparisons*

1. Comparing to a reference

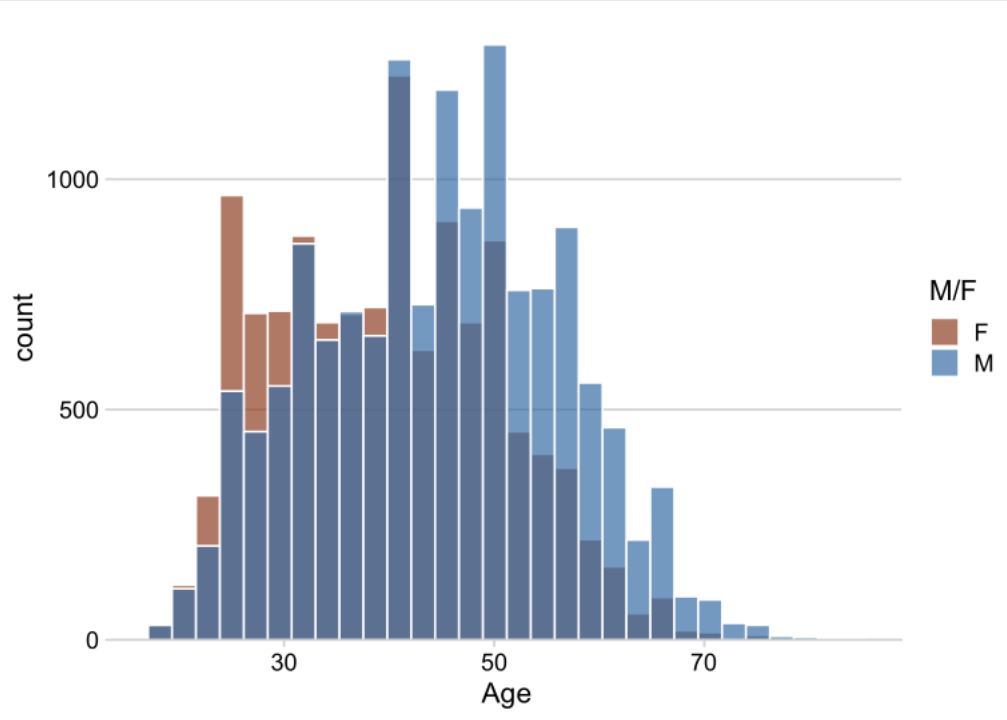
2. Comparing variables

BREAK

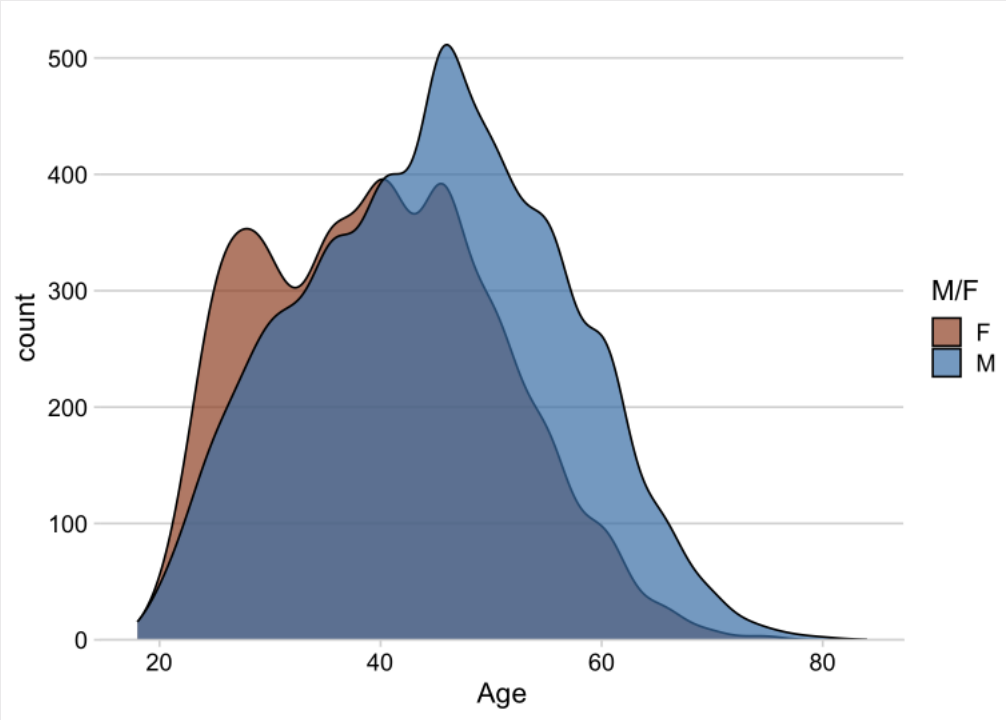
3. Comparing distributions

# Overlapping histograms have issues

Bad

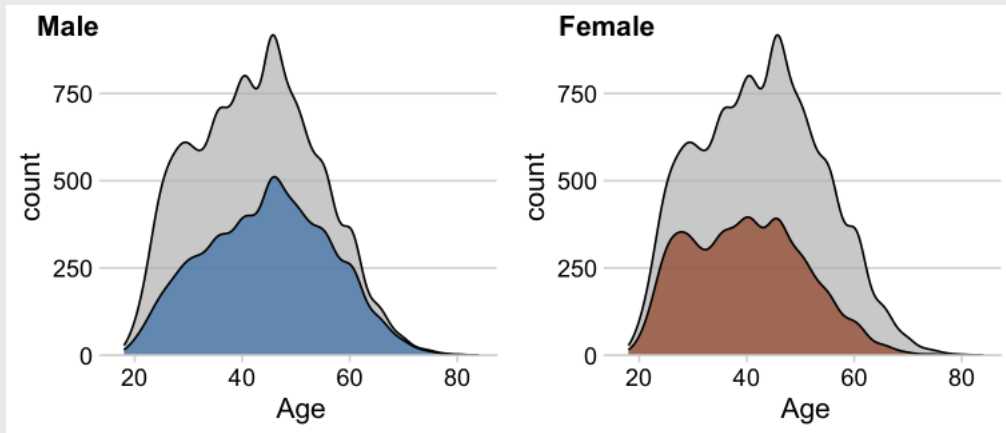


Slightly better

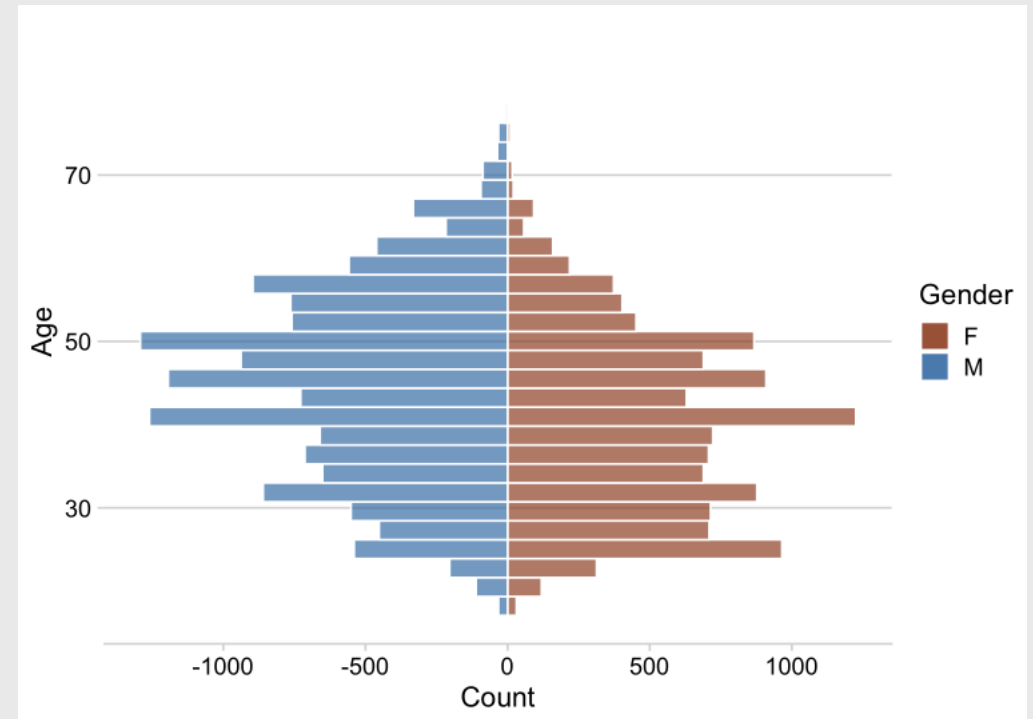


# Good when number of categories is **small**

## Density facets

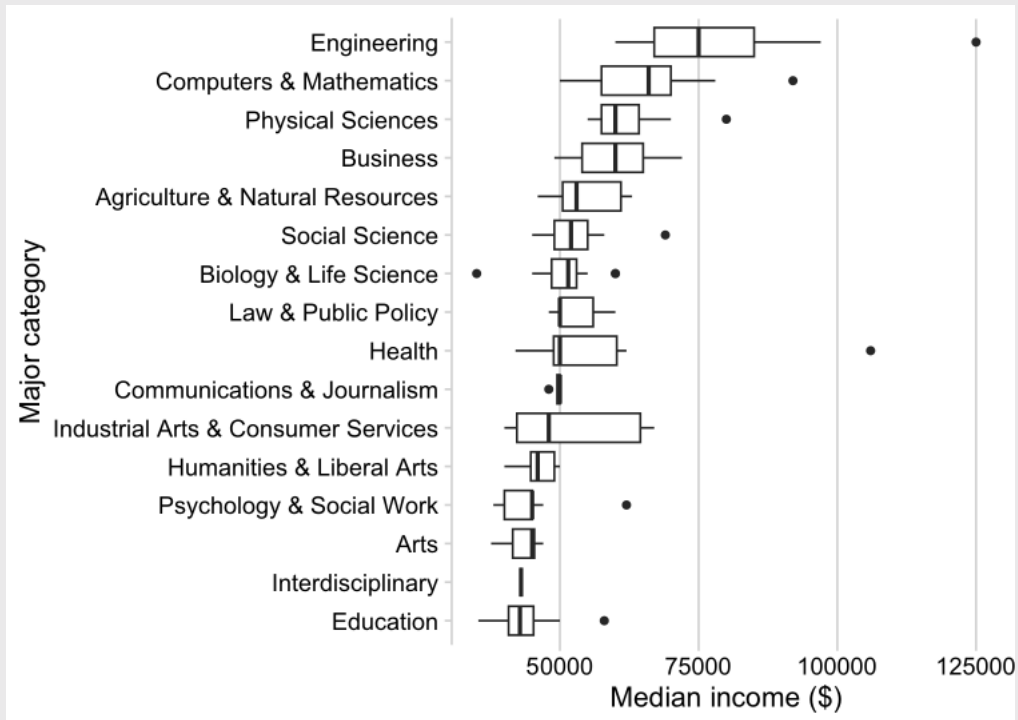


## Diverging histograms

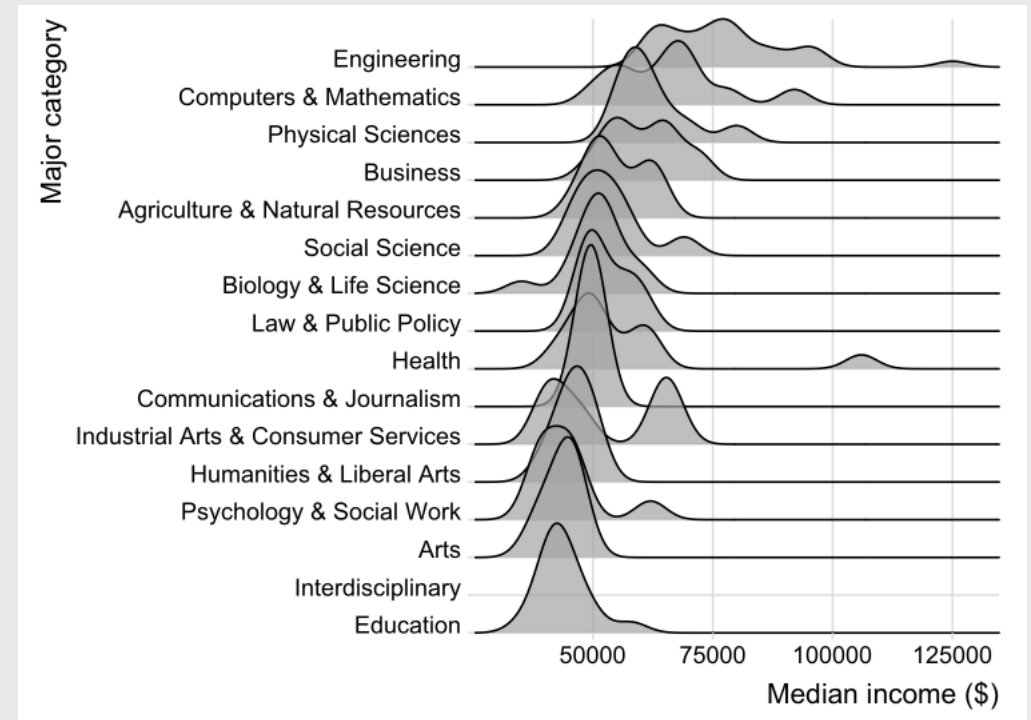


# Good when number of categories is **large**

## Boxplot



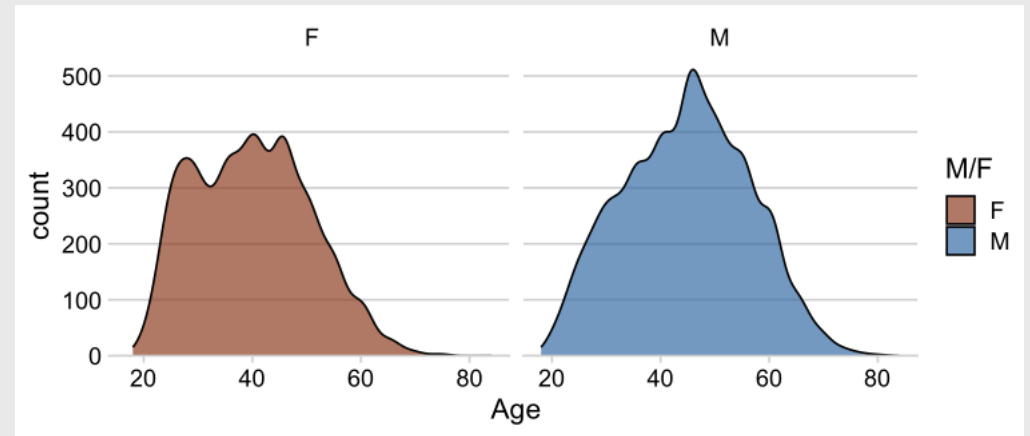
## Ridgeplot



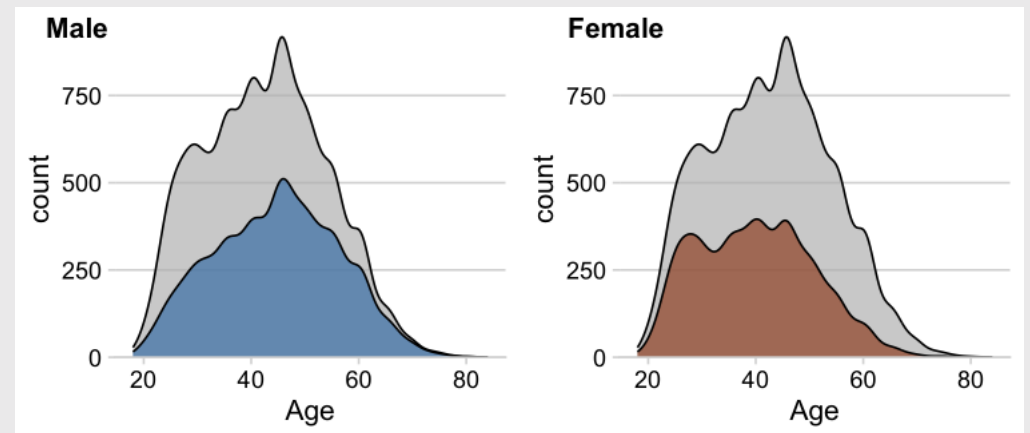
# How to make density facets

You can use `facet_wrap()`, but you won't get the full density overlay

```
ggplot(marathon,
  aes(x = Age, y = ..count..,
    fill = `M/F`)) +
  geom_density(alpha = 0.7) +
  facet_wrap(vars(`M/F`)) +
  scale_fill_manual(
    values = c('sienna', 'steelblue')) +
  scale_y_continuous(
    expand = expansion(mult = c(0, 0.05)))
  theme_minimal_hgrid()
```



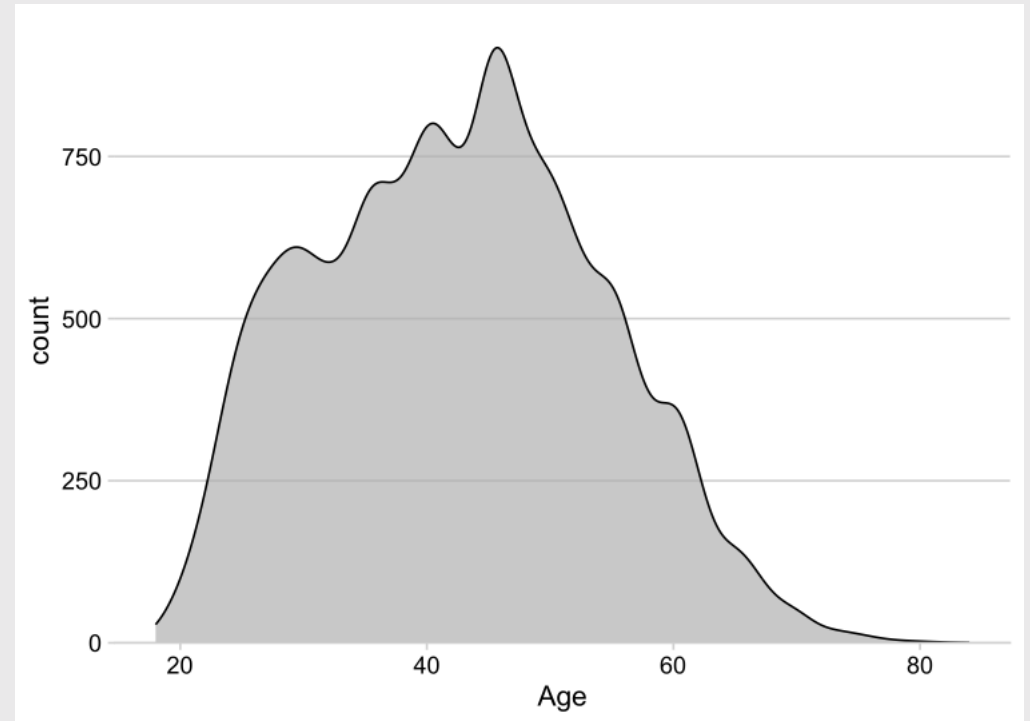
If you want the full density overlay, you have to hand-make the facets



# How to make density facets

Make the full density plot first

```
base <- ggplot(marathon,  
              aes(x = Age, y = ..count..)) +  
  geom_density(fill = 'grey', alpha = 0.7) +  
  scale_y_continuous(  
    expand = expansion(mult = c(0, 0.05)))  
  theme_minimal_hgrid()
```

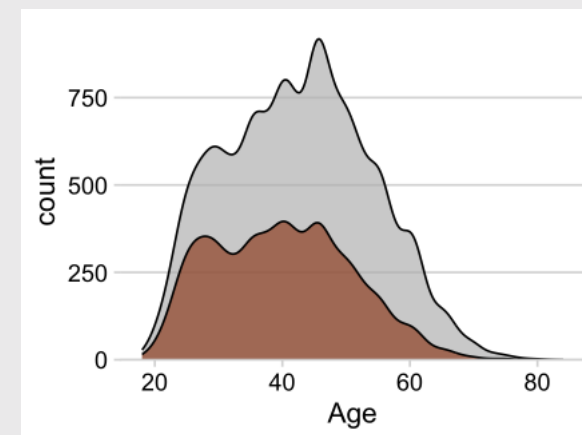
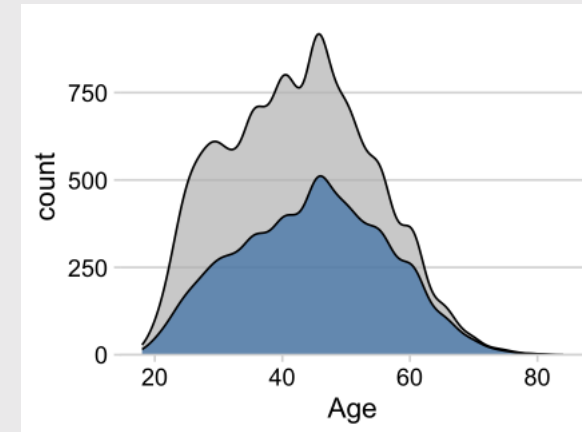


# How to make density facets

Separately create each sub-plot

```
male <- base +  
  geom_density(  
    data = marathon %>%  
      filter(`M/F` == 'M'),  
    fill = 'steelblue', alpha = 0.7) +  
  theme(legend.position = 'none')
```

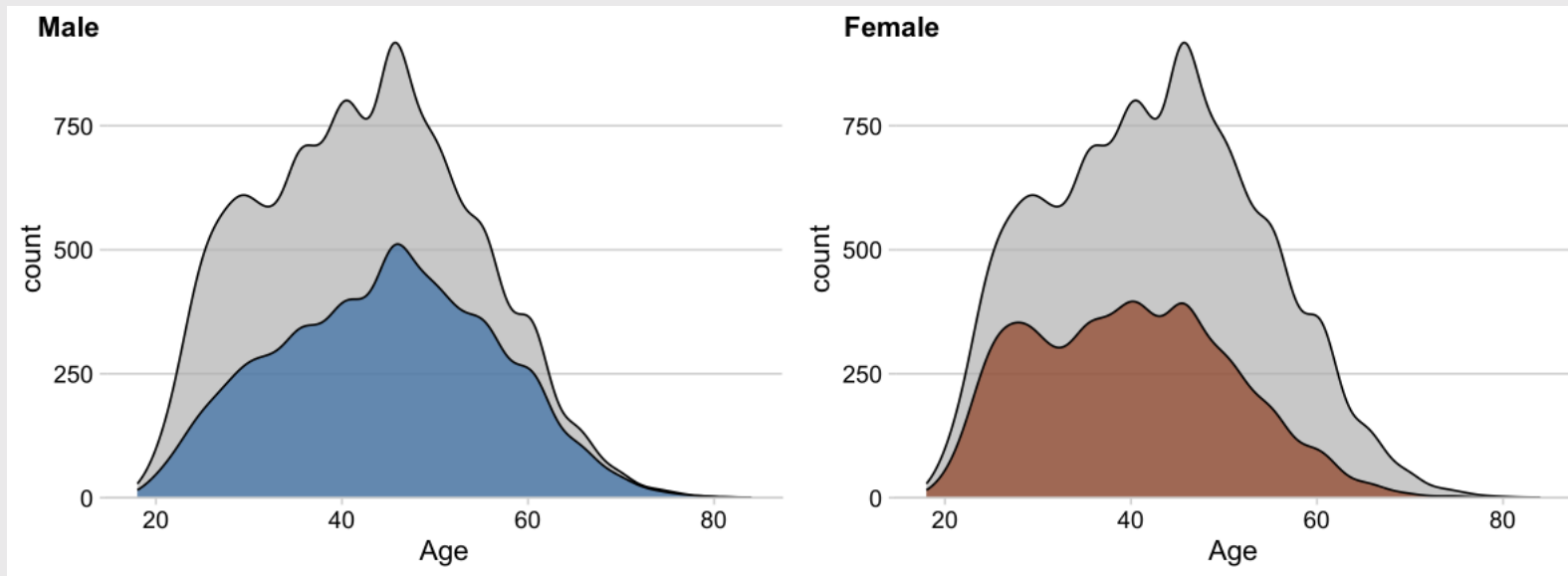
```
female <- base +  
  geom_density(  
    data = marathon %>%  
      filter(`M/F` == 'F'),  
    fill = 'sienna', alpha = 0.7) +  
  theme(legend.position = 'none')
```



# How to make density facets

Combine into single plot

```
plot_grid(male, female,  
         labels = c('Male', 'Female'))
```

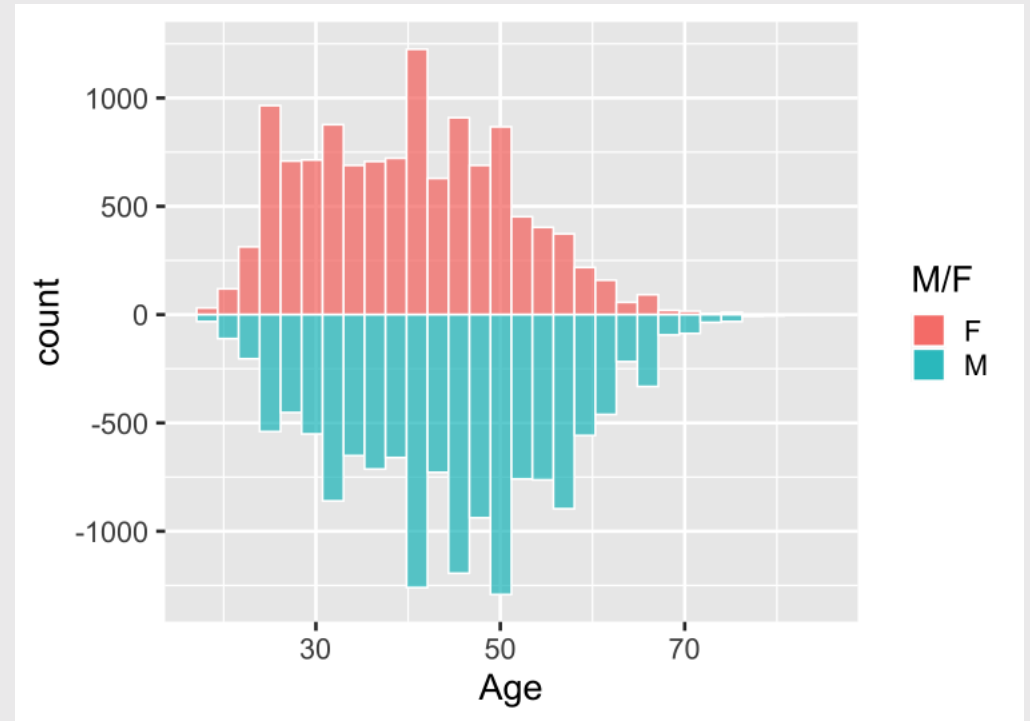




# How to make diverging histograms

Make the histograms by filtering the data

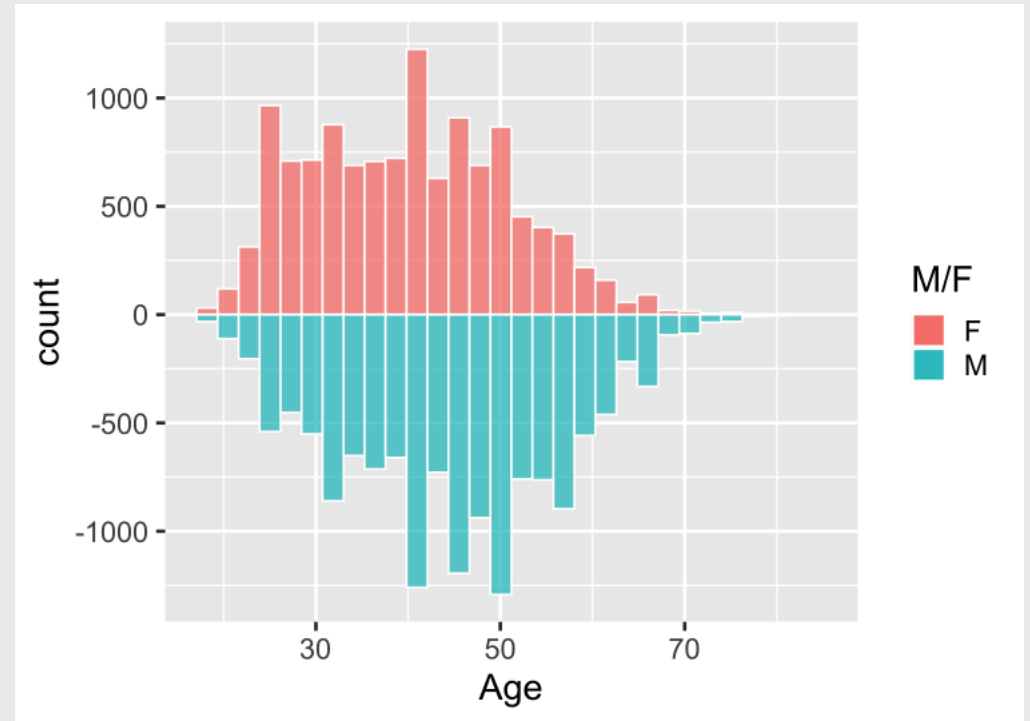
```
ggplot(marathon, aes(x = Age)) +  
  # Add histogram for Female runners:  
  geom_histogram(  
    data = marathon %>%  
      filter(`M/F` == 'F'),  
    aes(fill = `M/F`, y=..count..),  
    alpha = 0.7, color = 'white') +  
  # Add negative histogram for Male runners:  
  geom_histogram(  
    data = marathon %>%  
      filter(`M/F` == 'M'),  
    aes(fill = `M/F`, y=..count..*(-1)),  
    alpha = 0.7, color = 'white')
```



# How to make diverging histograms

Make the histograms by filtering the data

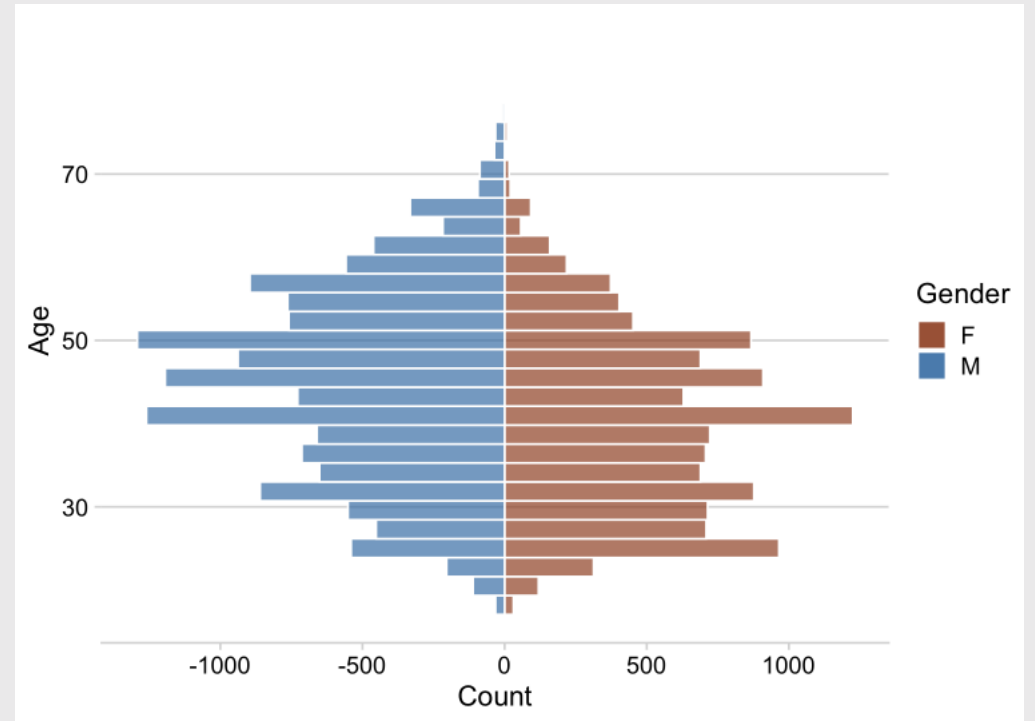
```
ggplot(marathon, aes(x = Age)) +  
  # Add histogram for Female runners:  
  geom_histogram(  
    data = marathon %>%  
      filter(`M/F` == 'F'),  
    aes(fill = `M/F`, y=..count..),  
    alpha = 0.7, color = 'white') +  
  # Add negative histogram for Male runners:  
  geom_histogram(  
    data = marathon %>%  
      filter(`M/F` == 'M'),  
    aes(fill = `M/F`, y=..count..*(-1)),  
    alpha = 0.7, color = 'white')
```



# How to make diverging histograms

Rotate, adjust colors, theme, annotate

```
ggplot(marathon, aes(x = Age)) +  
  # Add histogram for Female runners:  
  geom_histogram(  
    data = marathon %>%  
      filter(`M/F` == 'F'),  
    aes(fill = `M/F`, y = ..count..),  
    alpha = 0.7, color = 'white') +  
  # Add negative histogram for Male runners:  
  geom_histogram(  
    data = marathon %>%  
      filter(`M/F` == 'M'),  
    aes(fill = `M/F`, y = ..count..*(-1)),  
    alpha = 0.7, color = 'white')  
  scale_fill_manual(  
    values = c('sienna', 'steelblue')) +  
  coord_flip() +  
  theme_minimal_hgrid() +  
  labs(fill = 'Gender',  
       y = 'Count')
```

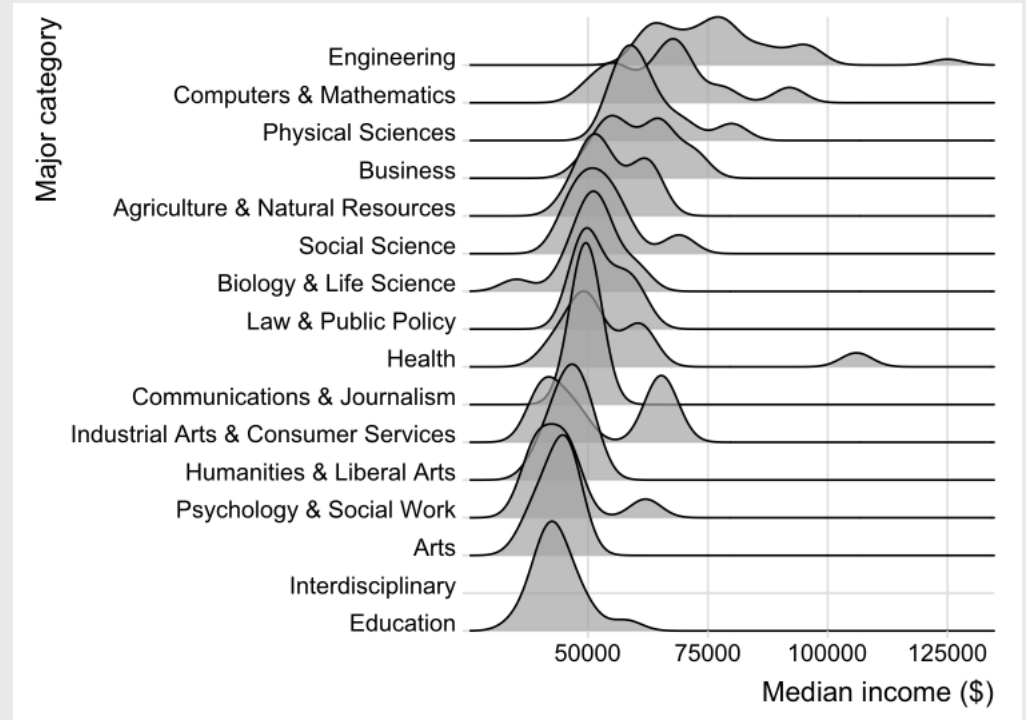


# How to make ridgeplots

Make a ridgeplot with **ggridges** library

```
library(ggridges)

college_all_ages %>%
  mutate(
    major_category = fct_reorder(
      major_category, median)) %>%
  ggplot() +
  geom_density_ridges(
    aes(x = median, y = major_category),
    scale = 4, alpha = 0.7) +
  scale_y_discrete(expand = c(0, 0)) +
  scale_x_continuous(expand = c(0, 0)) +
  coord_cartesian(clip = "off") +
  theme_ridges() +
  labs(x = 'Median income ($)',
       y = 'Major category')
```



# Your turn - comparing distributions

15:00

Use the [gapminder.csv](#) data to create the following charts comparing the distribution of life expectancy across countries in continents in 2007.

