Week 5: Graphing comparisons

EMSE 4197 | John Paul Helveston | February 12, 2020

R tip of the week

1) Quick shortcuts

Insert a `<-` 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'))
federal_spending <- read_csv(here('data', 'federal_spending_long.csv'))
gapminder        <- read_csv(here('data', 'gapminder.csv'))
marathon         <- read_csv(here('data', 'marathon.csv'))
milk_production  <- read_csv(here('data', 'milk_production.csv'))
```

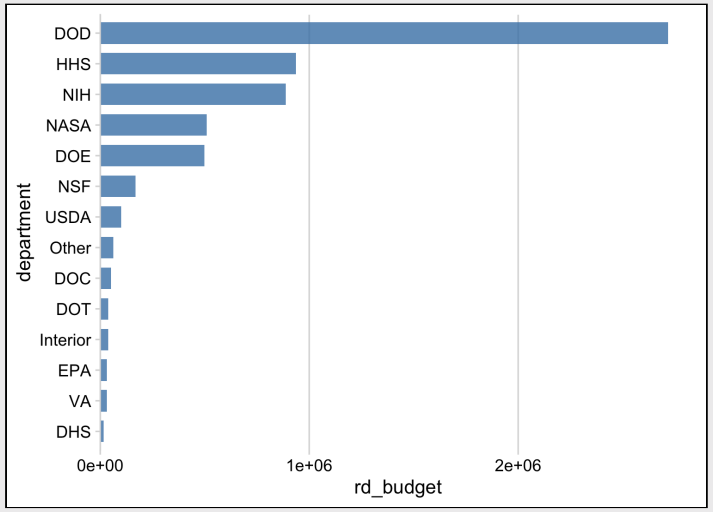
Graphing comparisons

1. Ranking things
2. Comparing things to a reference
3. Comparing two things
4. Comparing distributions

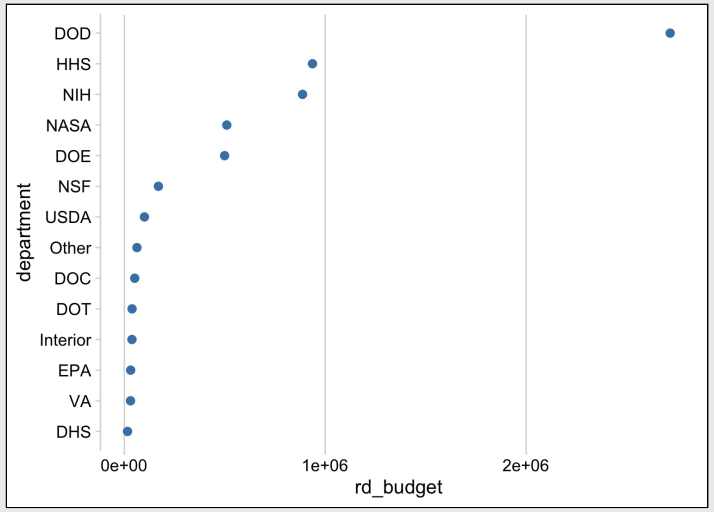
Graphing comparisons

1. Ranking things
2. Comparing things to a reference
3. Comparing two things
4. Comparing distributions

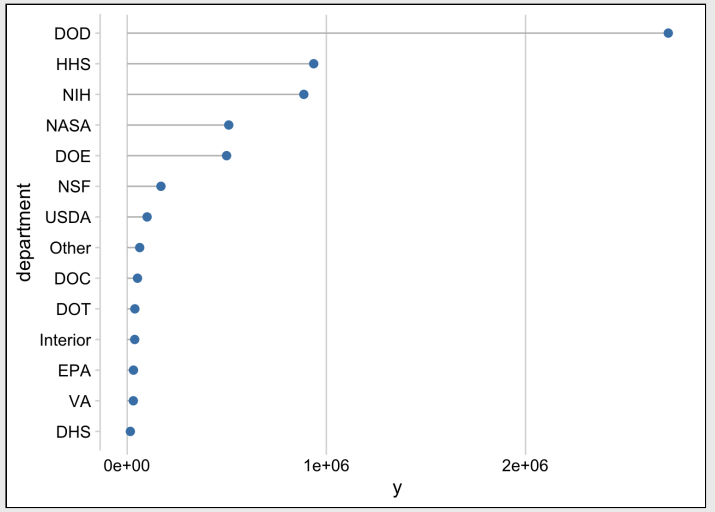
Bar chart



Dot chart



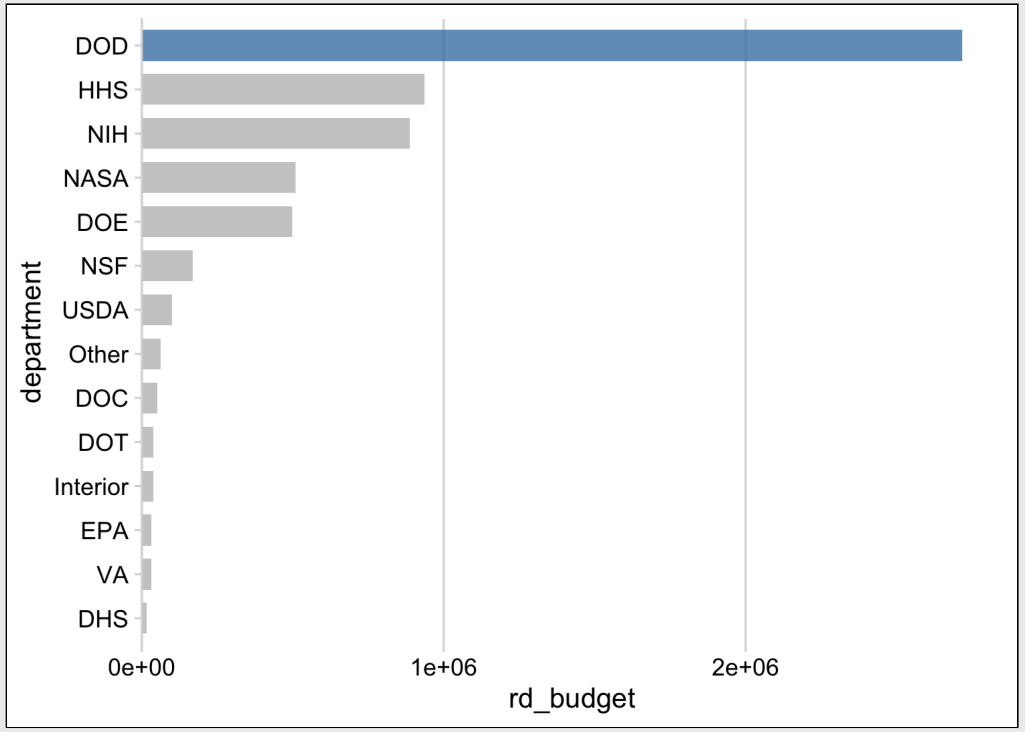
Lollipop chart



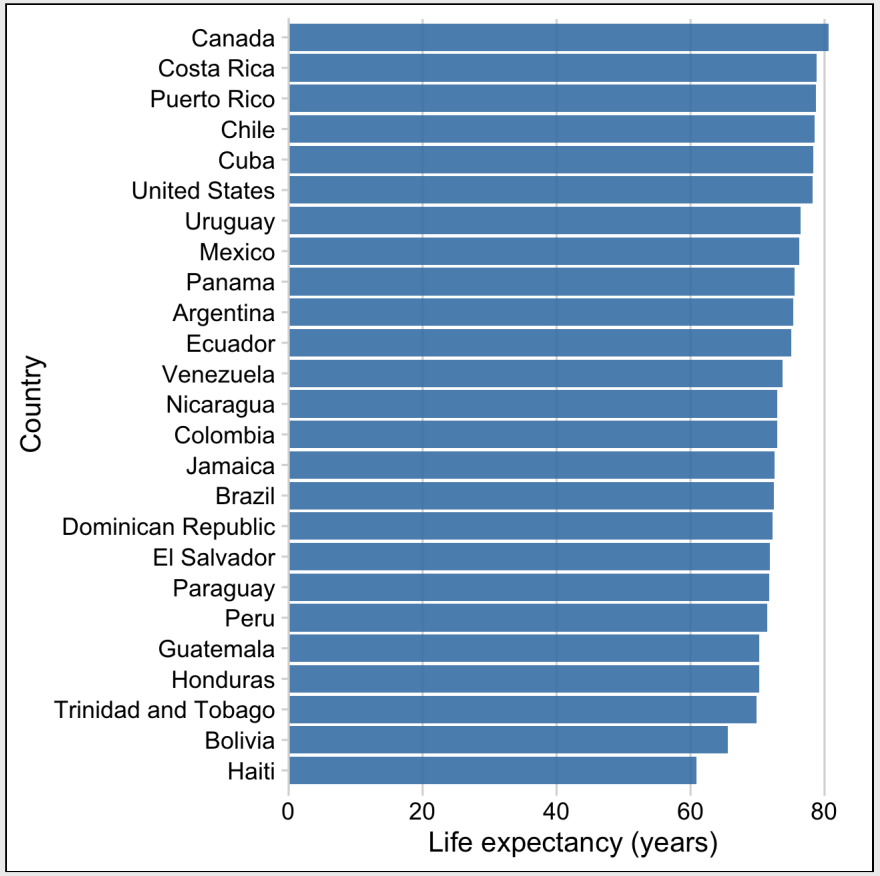
For ranking,
bars are a great starting point

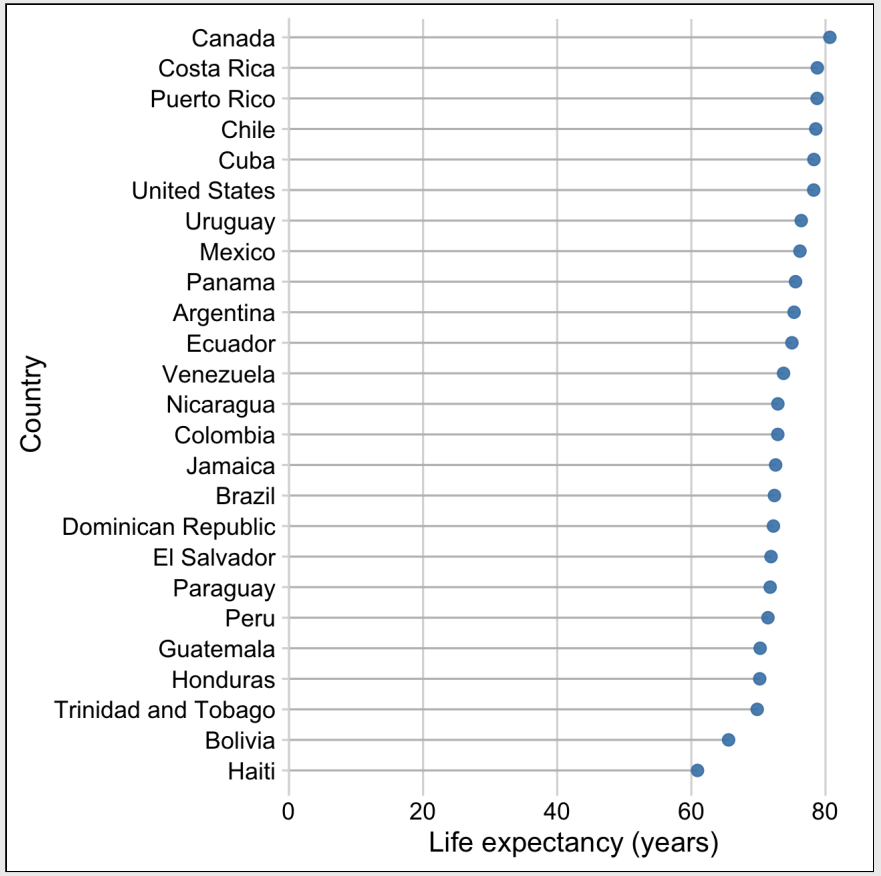
Bars are good for highlighting categories w/color

The DOD's R&D budget is nearly the
same as all other departments combined

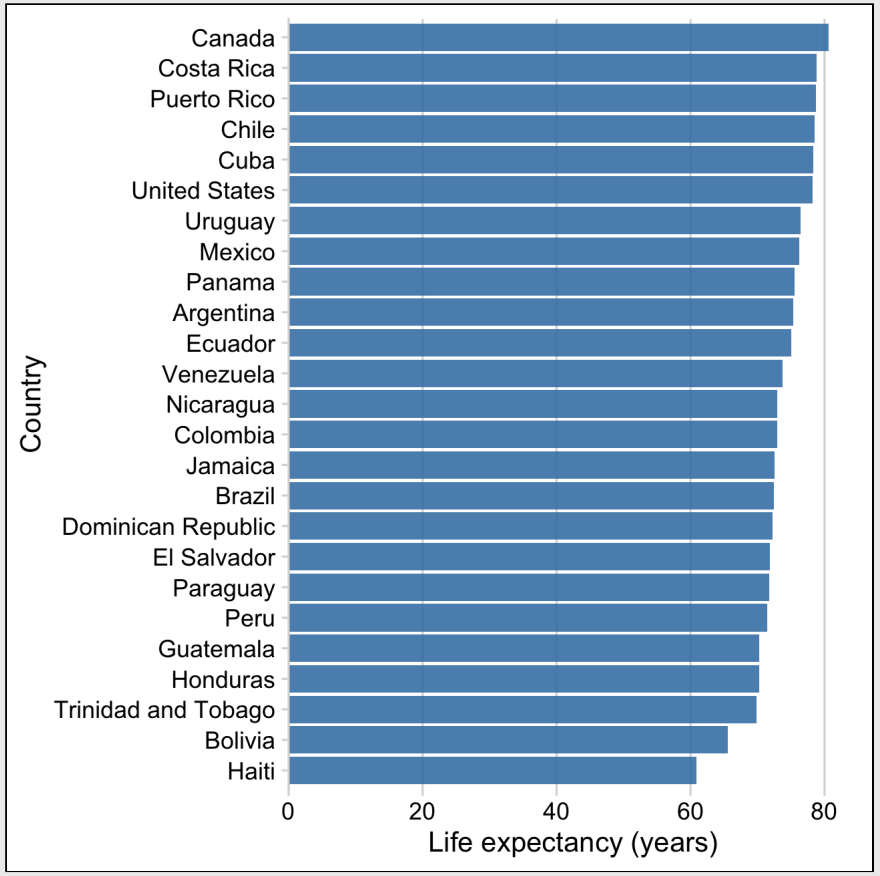


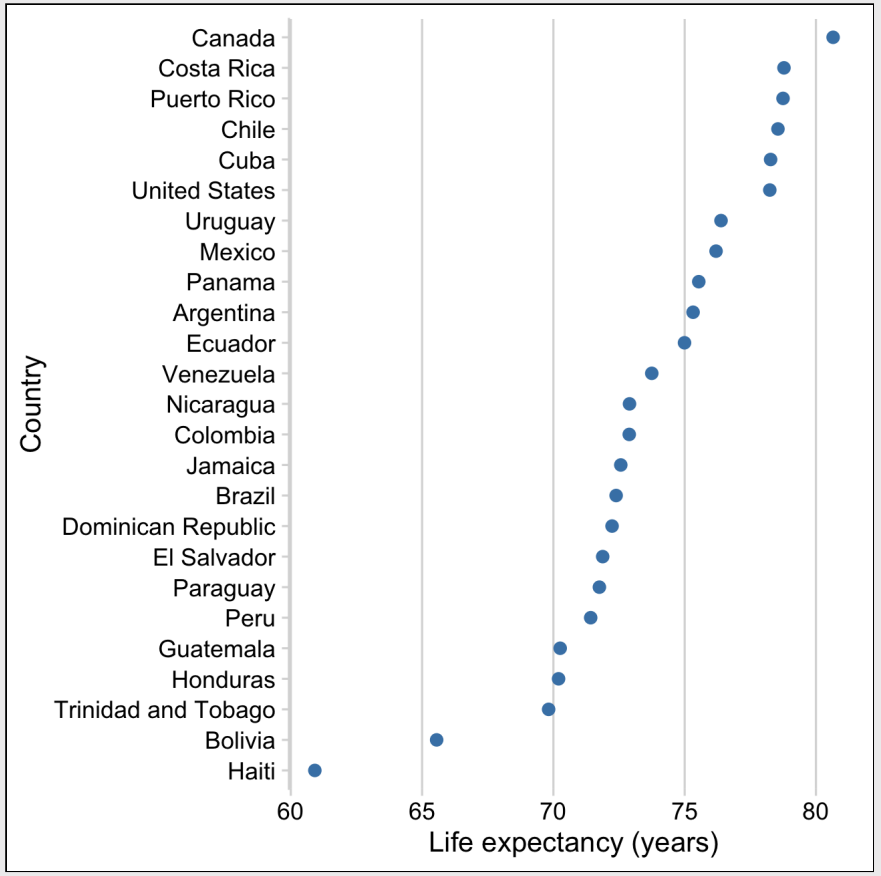
Use lollipops when the bars are overwhelming





Or use dots and don't set axis to 0





How to make a **Bar chart**

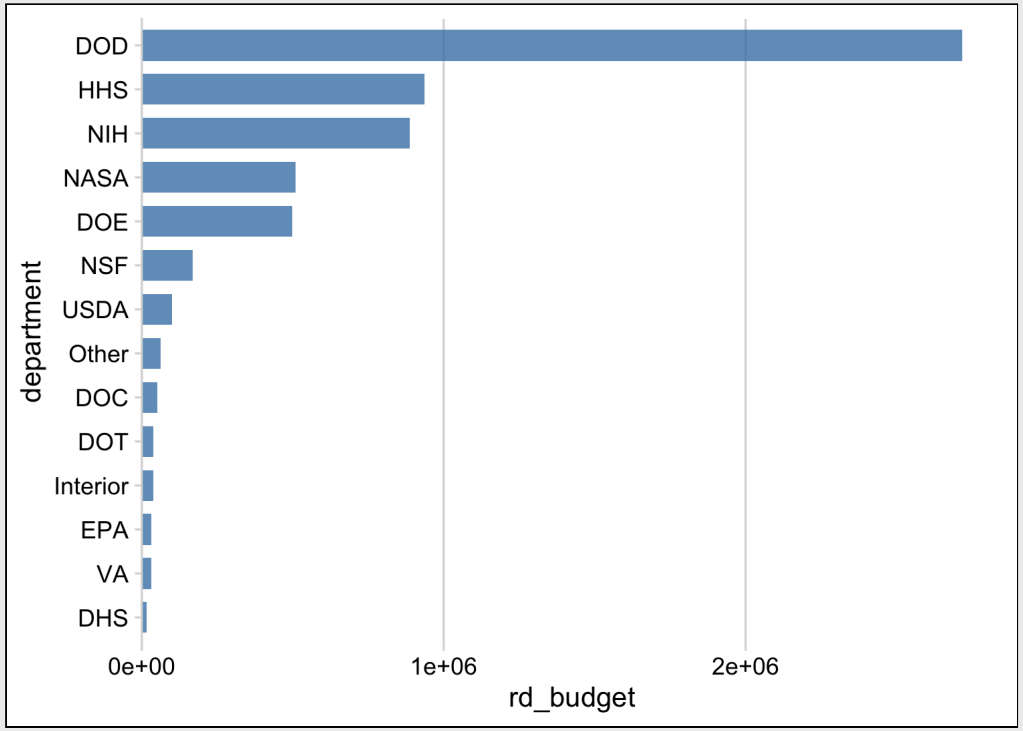
Summarize data frame:

```
federal_spending_summary <- federal_spending %>%
  group_by(department) %>%
  summarise(rd_budget = sum(rd_budget)) %>%
  mutate(
    department = fct_reorder(department, rd_budget))
```

Make chart:

```
ggplot(federal_spending_summary) +
  geom_col(aes(x = department, y = rd_budget),
           width = 0.7, alpha = 0.8,
           fill = 'steelblue') +
  scale_y_continuous(
    expand = expand_scale(mult = c(0, 0.05))) +
  coord_flip() +
  theme_minimal_vgrid()
```

Bar chart of federal R&D spending by department



How to make a **Bar chart** with color

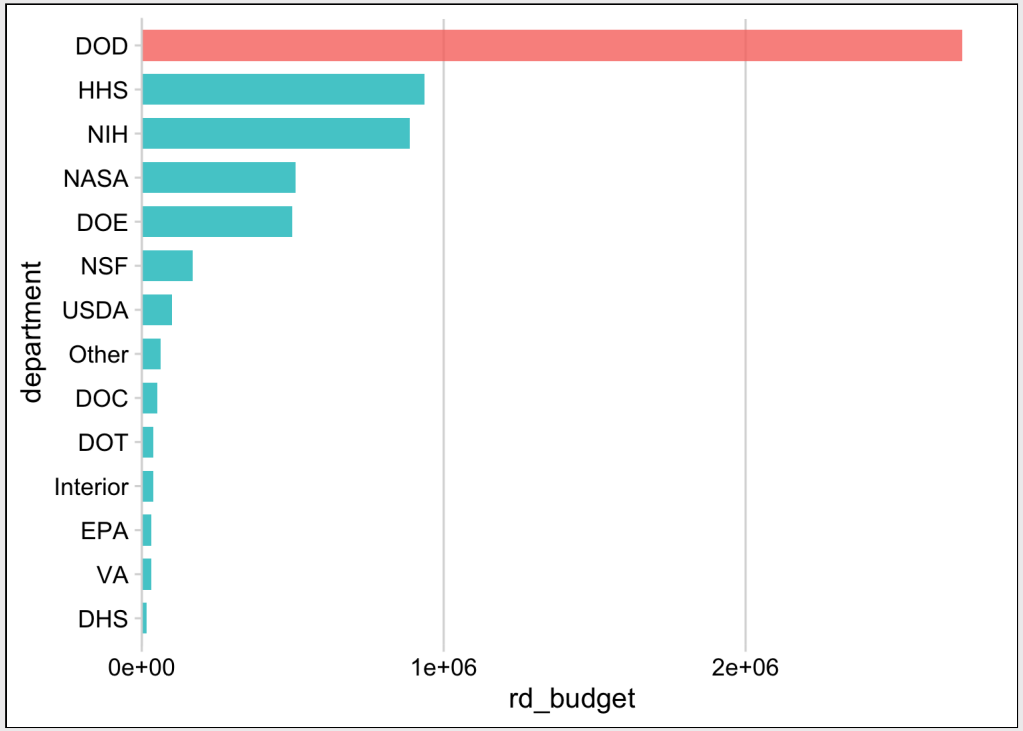
Summarize data frame:

```
federal_spending_summary <- federal_spending %>%
  group_by(department) %>%
  summarise(rd_budget = sum(rd_budget)) %>%
  mutate(
    department = fct_reorder(department, rd_budget),
    dept_color = if_else(
      department == 'DOD', 'DOD', 'Other'))
```

Make chart:

```
ggplot(federal_spending_summary) +
  geom_col(aes(x = department, y = rd_budget,
               fill = dept_color),
           width = 0.7, alpha = 0.8) +
  scale_y_continuous(
    expand = expand_scale(mult = c(0, 0.05))) +
  coord_flip() +
  theme_minimal_vgrid() +
  theme(legend.position = 'none')
```

The DOD's R&D budget is nearly the same as all other departments combined



How to make a **Bar chart** with color

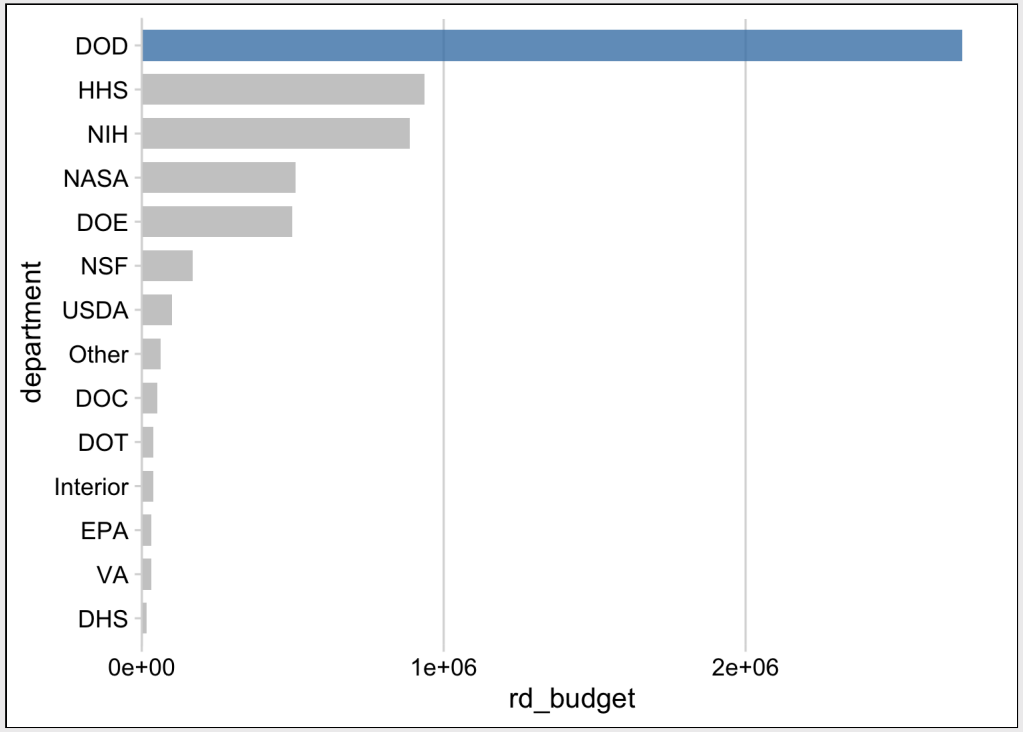
Summarize data frame:

```
federal_spending_summary <- federal_spending %>%
  group_by(department) %>%
  summarise(rd_budget = sum(rd_budget)) %>%
  mutate(
    department = fct_reorder(department, rd_budget),
    dept_color = if_else(
      department == 'DOD', 'DOD', 'Other'))
```

Make chart:

```
ggplot(federal_spending_summary) +
  geom_col(aes(x = department, y = rd_budget,
               fill = dept_color),
           width = 0.7, alpha = 0.8) +
  scale_fill_manual(values = c('steelblue', 'grey')) +
  scale_y_continuous(
    expand = expand_scale(mult = c(0, 0.05))) +
  coord_flip() +
  theme_minimal_vgrid() +
  theme(legend.position = 'none')
```

The DOD's R&D budget is nearly the same as all other departments combined



How to make a **Dot chart**

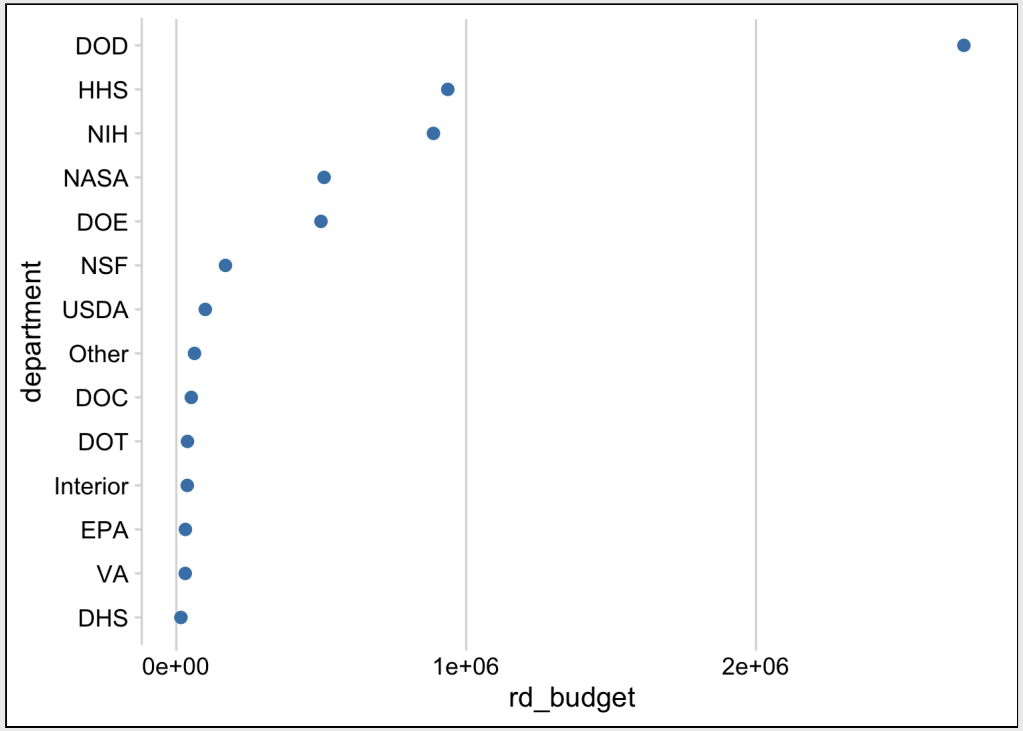
Summarize data frame:

```
federal_spending_summary <- federal_spending %>%
  group_by(department) %>%
  summarise(rd_budget = sum(rd_budget)) %>%
  mutate(
    department = fct_reorder(department, rd_budget))
```

Make chart:

```
ggplot(federal_spending_summary) +
  geom_point(aes(x = department, y = rd_budget),
             size = 2.5, color = 'steelblue') +
  coord_flip() +
  theme_minimal_vgrid()
```

Dot chart of federal R&D spending by department



How to make a **Lollipop chart**

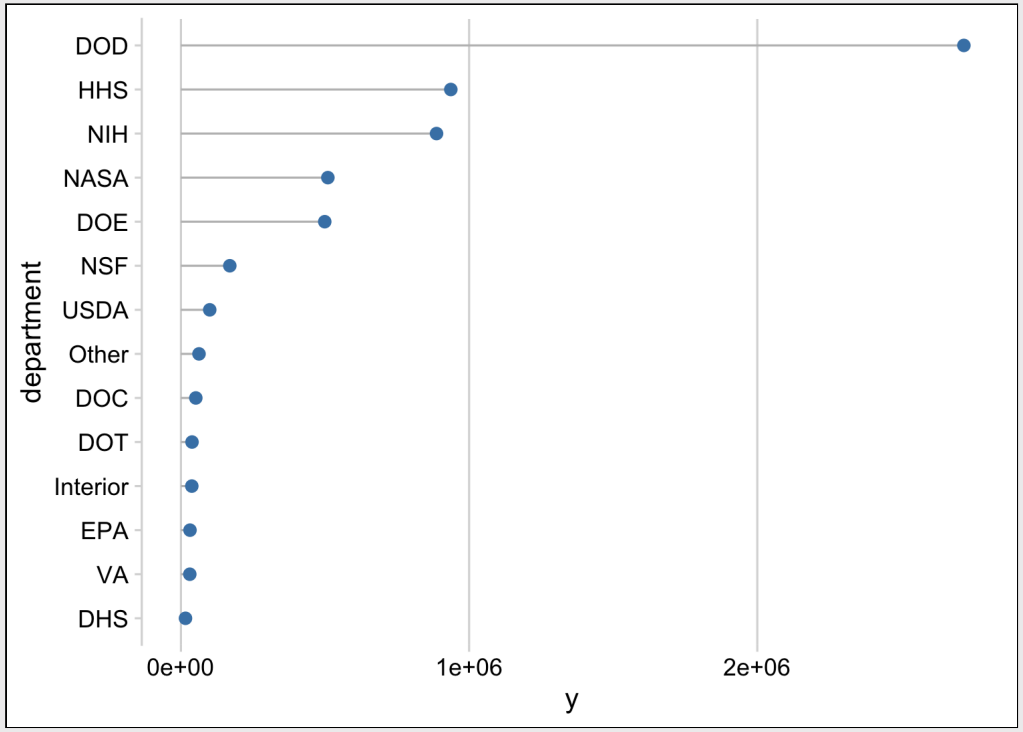
Summarize data frame:

```
federal_spending_summary <- federal_spending %>%
  group_by(department) %>%
  summarise(rd_budget = sum(rd_budget)) %>%
  mutate(
    department = fct_reorder(department, rd_budget))
```

Make chart:

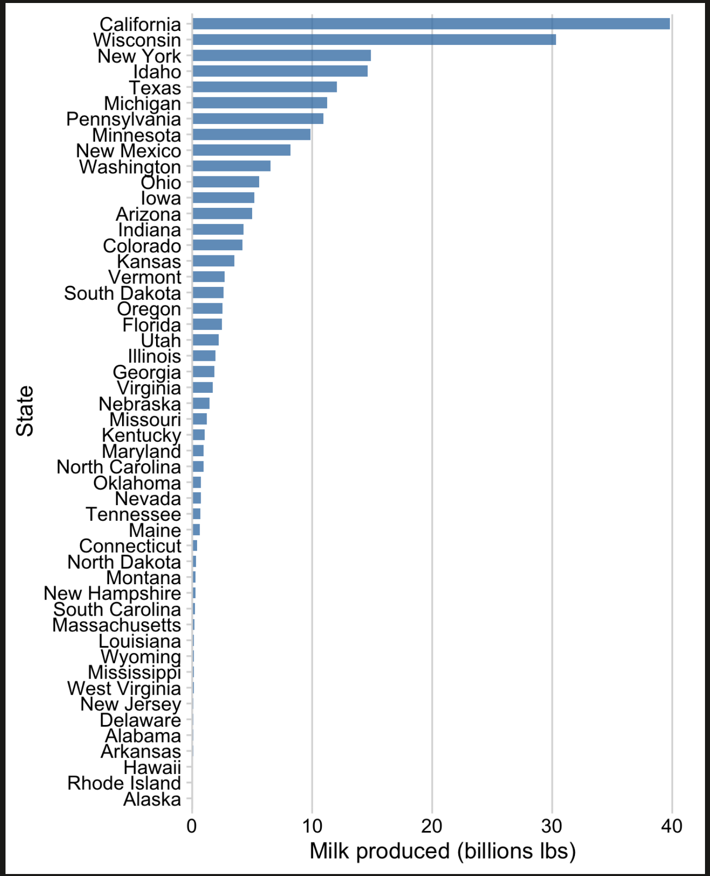
```
ggplot(federal_spending_summary) +
  geom_segment(aes(x    = department,
                   xend = department,
                   y    = 0,
                   yend = rd_budget),
               color = 'grey') +
  geom_point(aes(x = department, y = rd_budget),
             size = 2.5, color = 'steelblue') +
  coord_flip() +
  theme_minimal_vgrid()
```

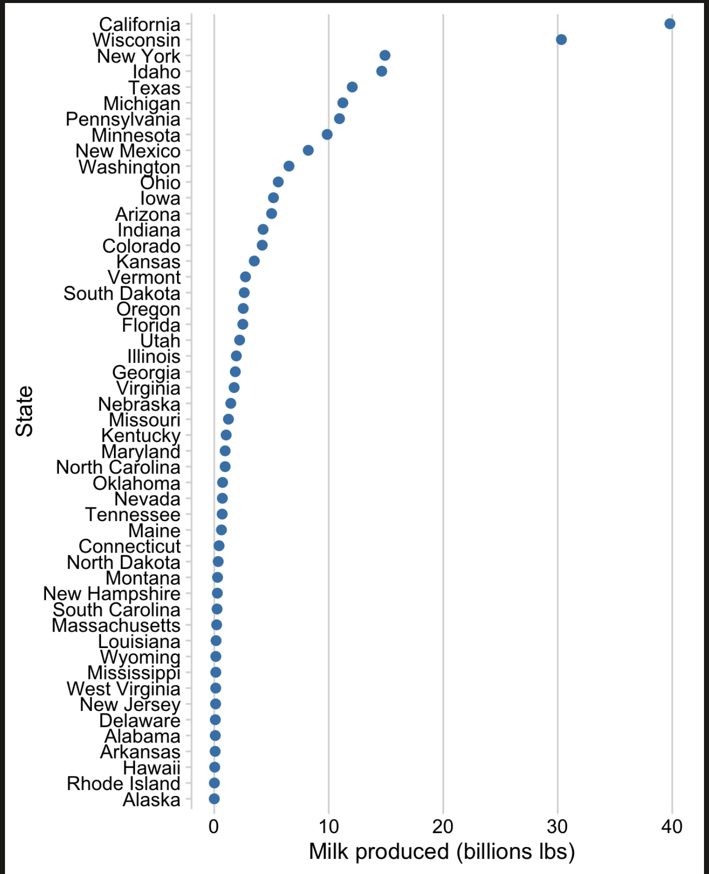
Lollipop chart of federal R&D spending by department

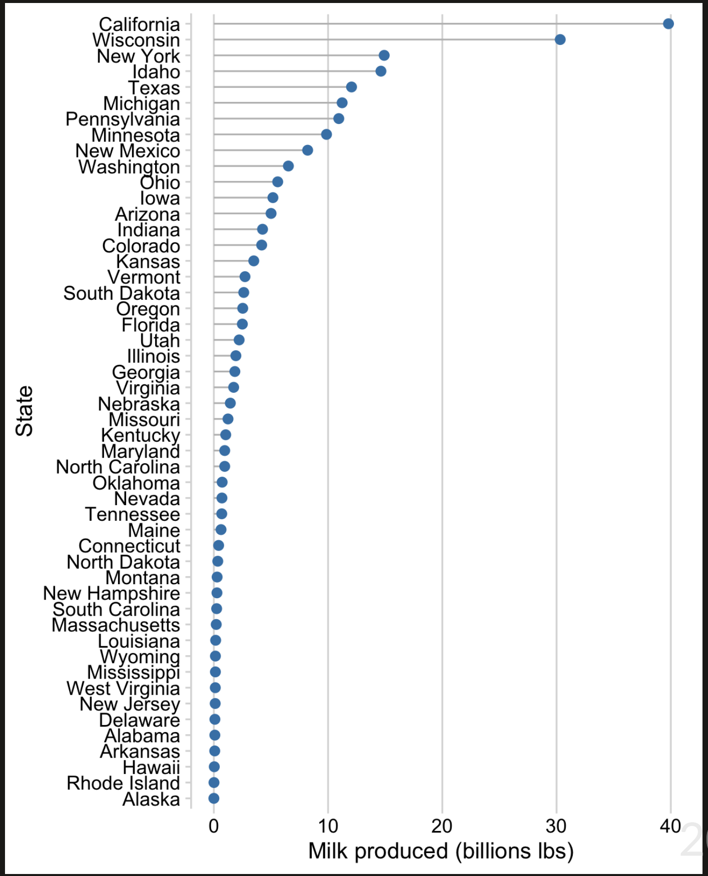


Your turn

Use the `milk_production.csv` data to create the following charts ranking states by 2017 milk production.



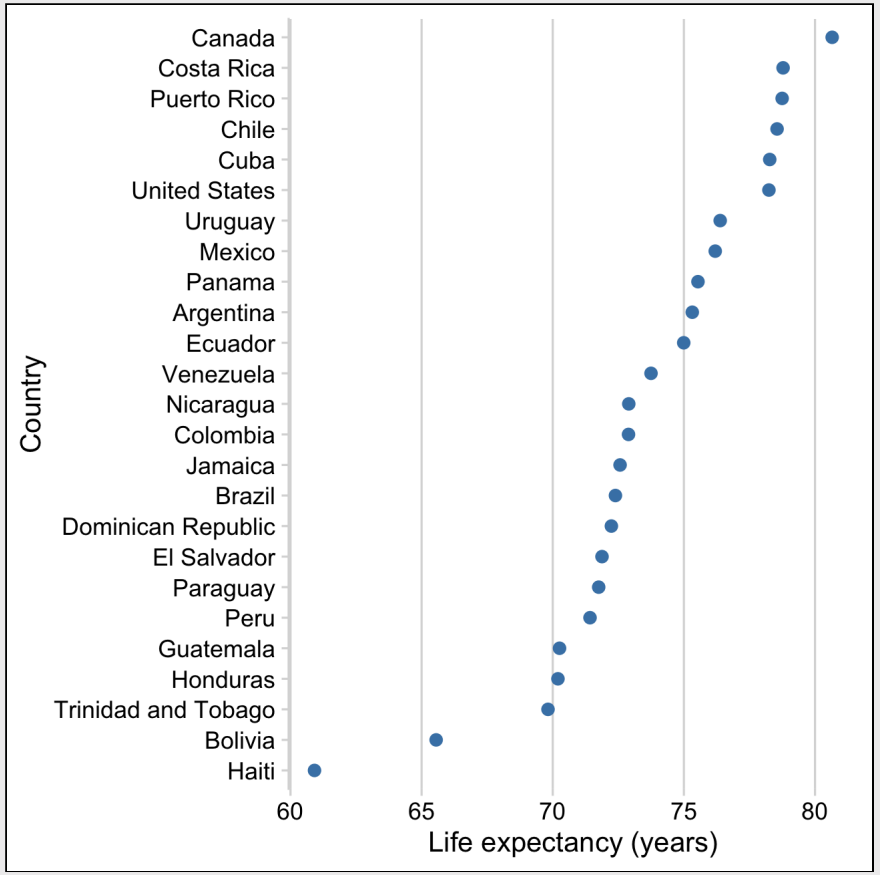


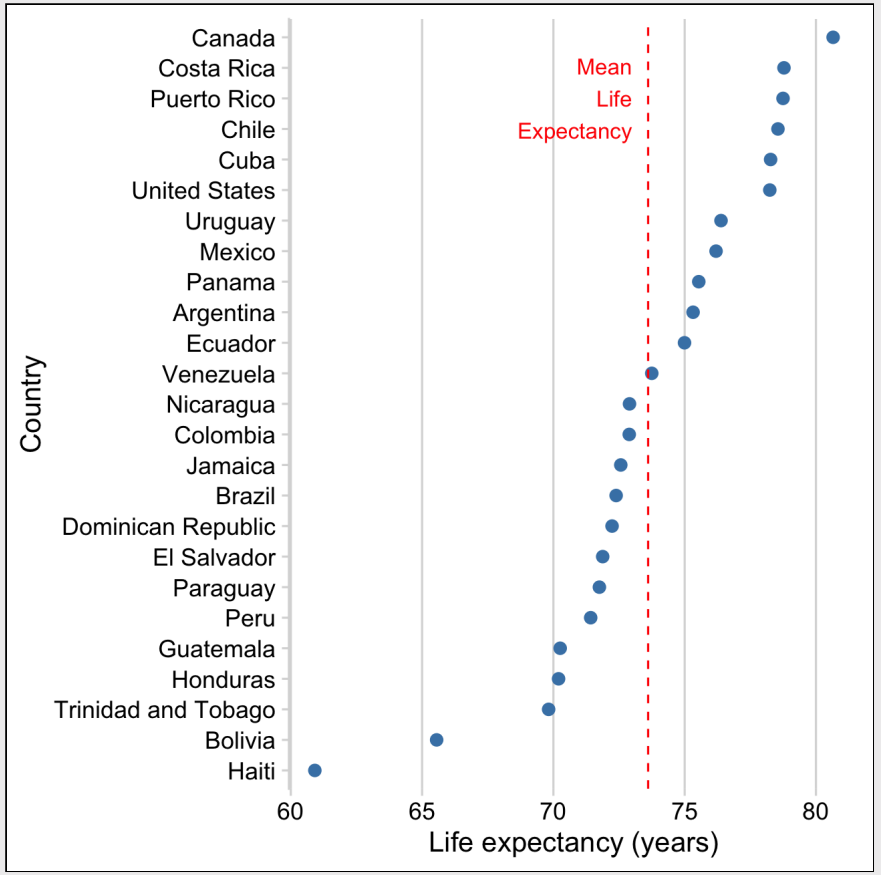


Graphing comparisons

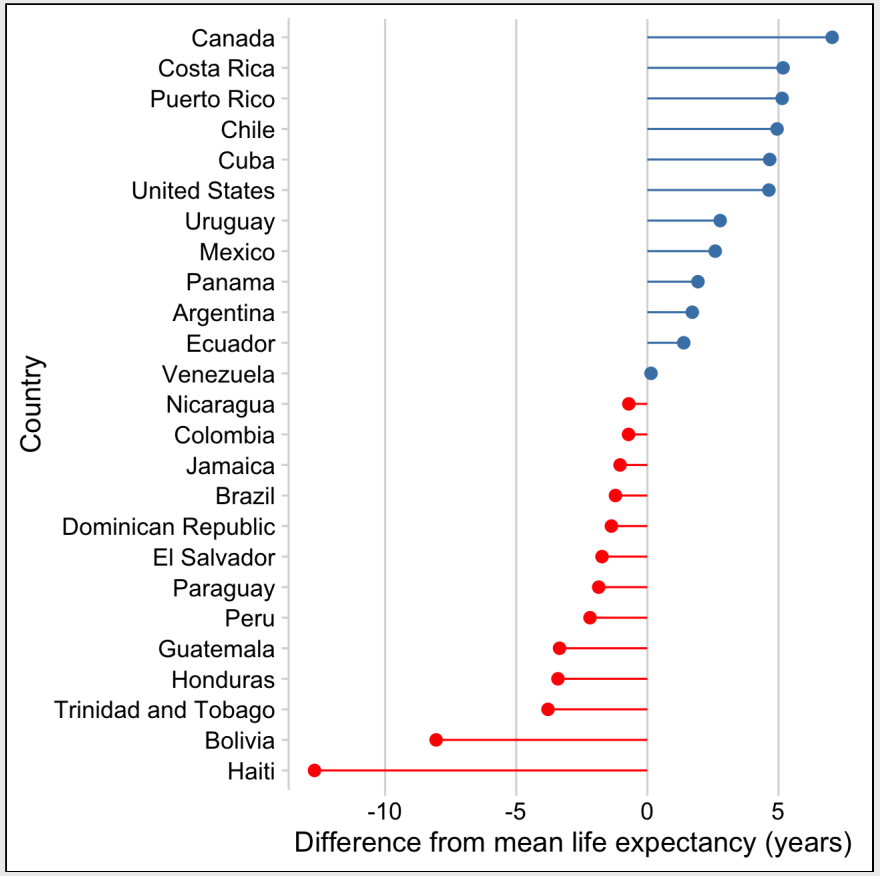
1. Ranking things
2. Comparing things to a reference
3. Comparing two things
4. Comparing distributions

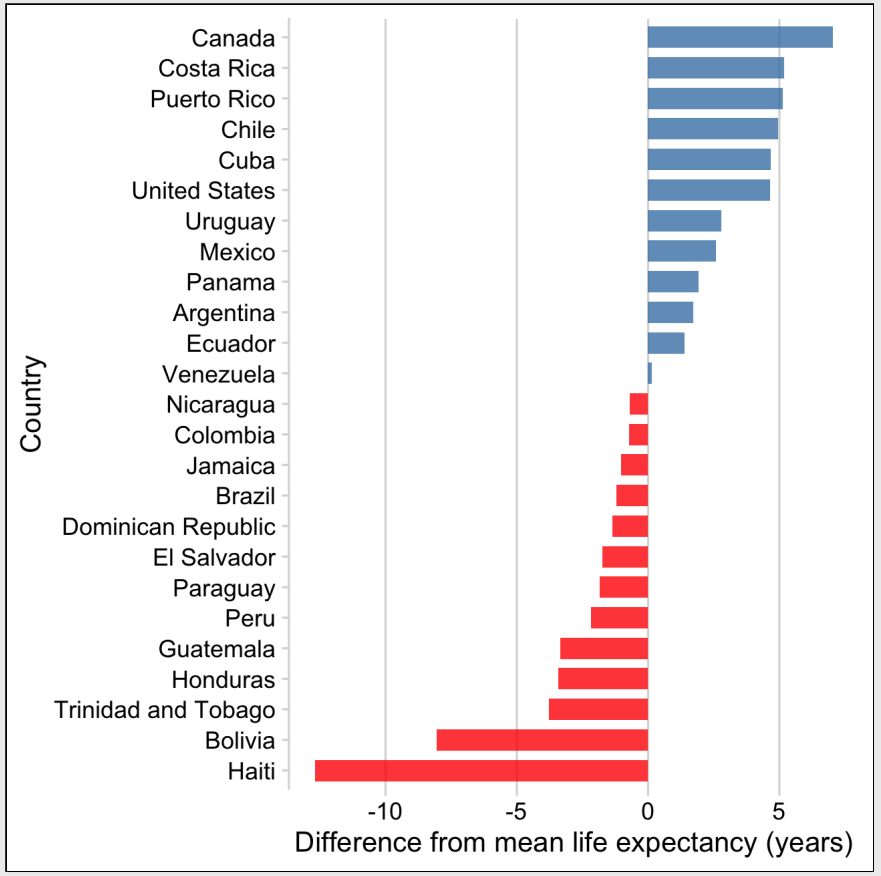
Use reference lines to add context to chart



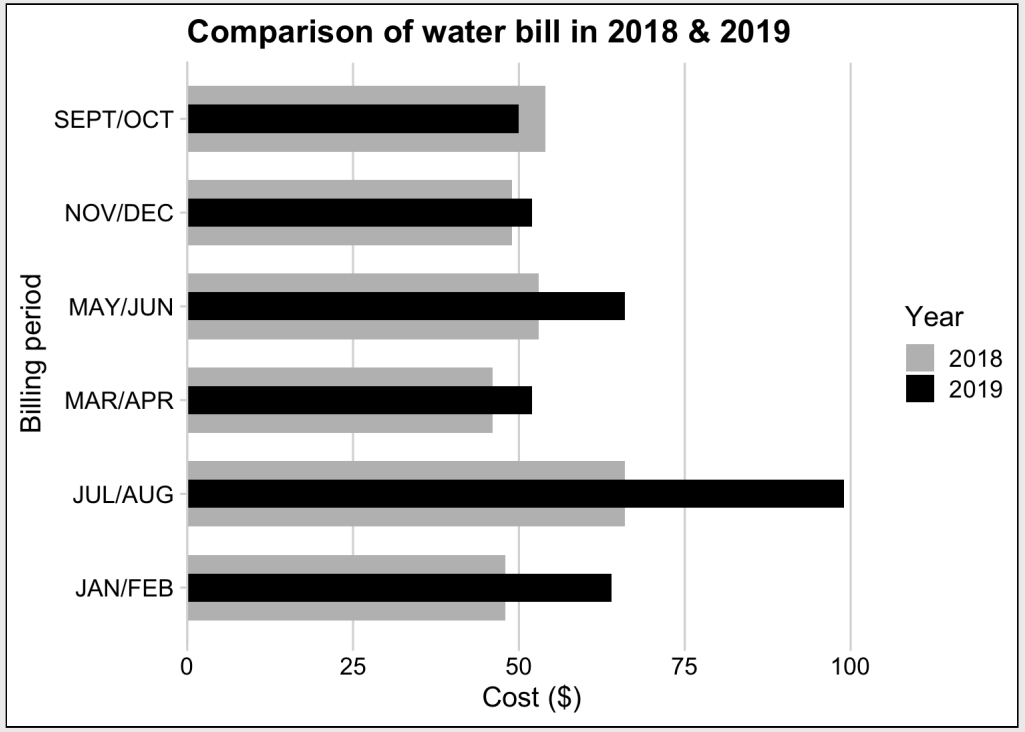


Or make zero the reference line





Compare multiple references with overlapping bars

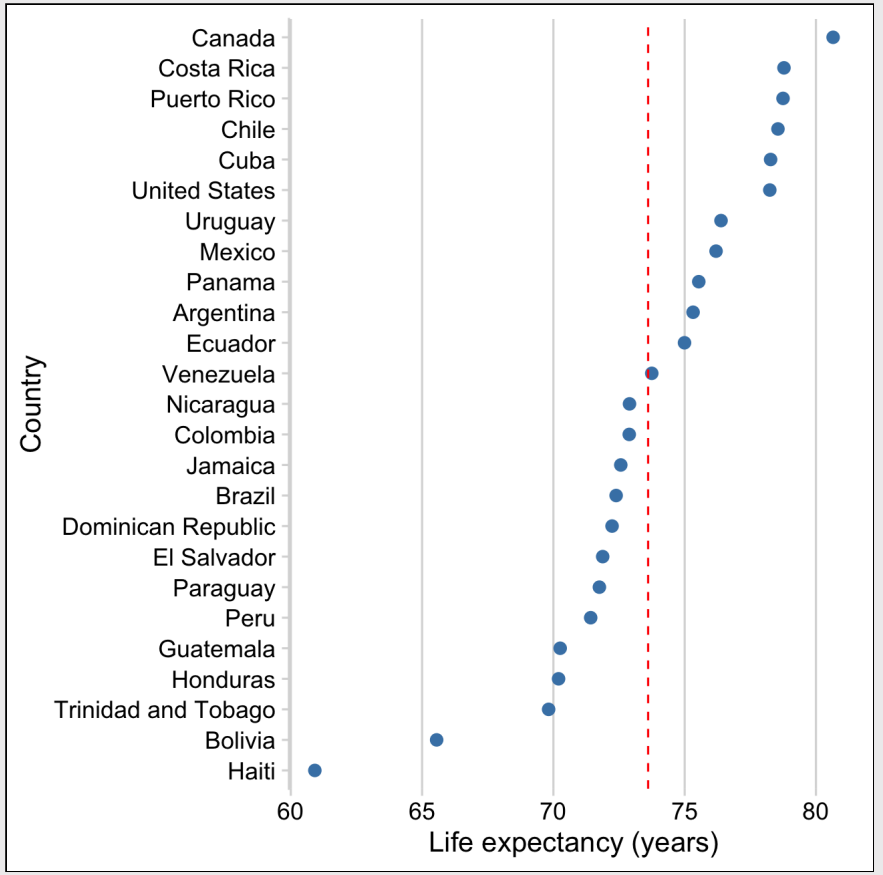


How to add a reference line

Add horizontal line with `geom_hline()`

Add vertical line with `geom_vline()`

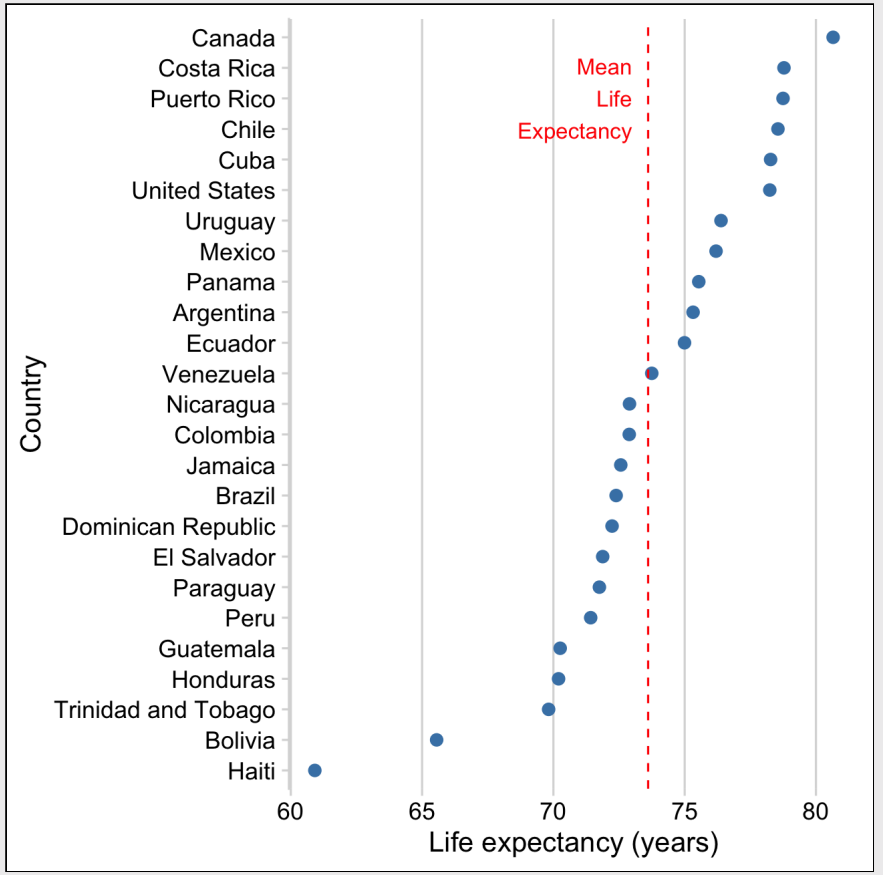
```
ggplot(gapminder_americas) +
  geom_point(aes(x = lifeExp, y = country),
             color = 'steelblue', size = 2.5) +
  geom_vline(
      xintercept = mean(gapminder_americas$lifeExp),
      color = 'red', linetype = 'dashed') +
  theme_minimal_vgrid() +
  labs(x = 'Life expectancy (years)',
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



How to add a reference line

Add text with `annotate()`

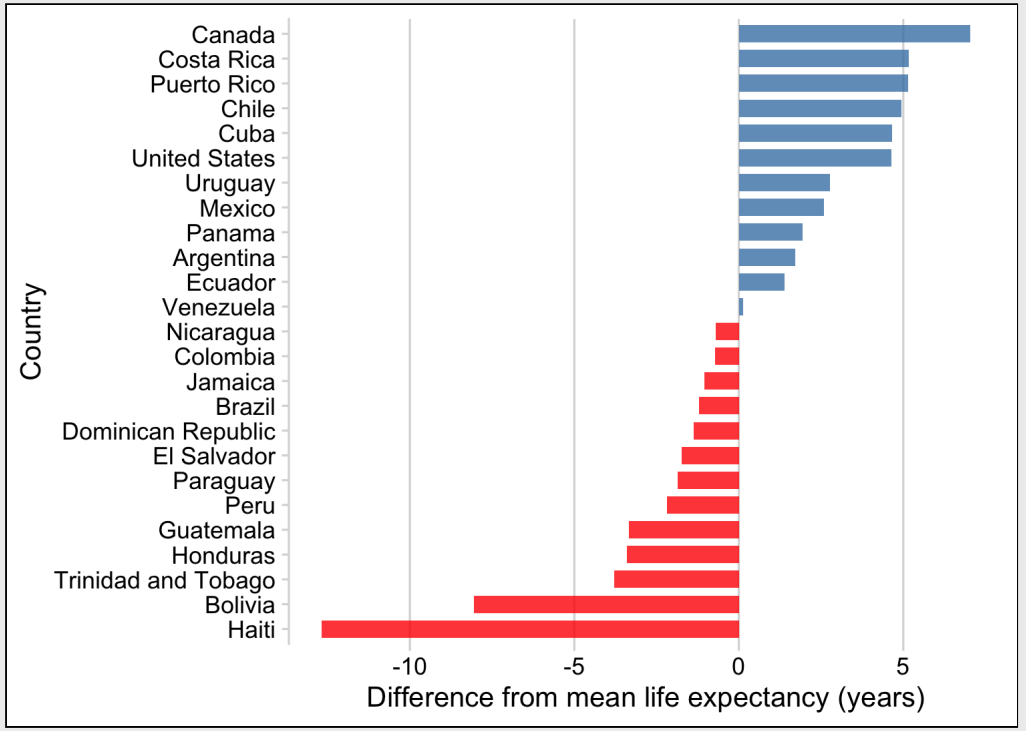
```
ggplot(gapminder_americas) +
  geom_point(aes(x = lifeExp, y = country),
             color = 'steelblue', size = 2.5) +
  geom_vline(
      xintercept = mean(gapminder_americas$lifeExp),
      color = 'red', linetype = 'dashed') +
  annotate('text', x = 73, y = 'Puerto Rico',
           color = 'red', hjust = 1,
           label = 'Mean\nLife\nExpectancy') +
  theme_minimal_vgrid() +
  labs(x = 'Life expectancy (years)',
       y = 'Country')
```



How to make zero the reference point

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

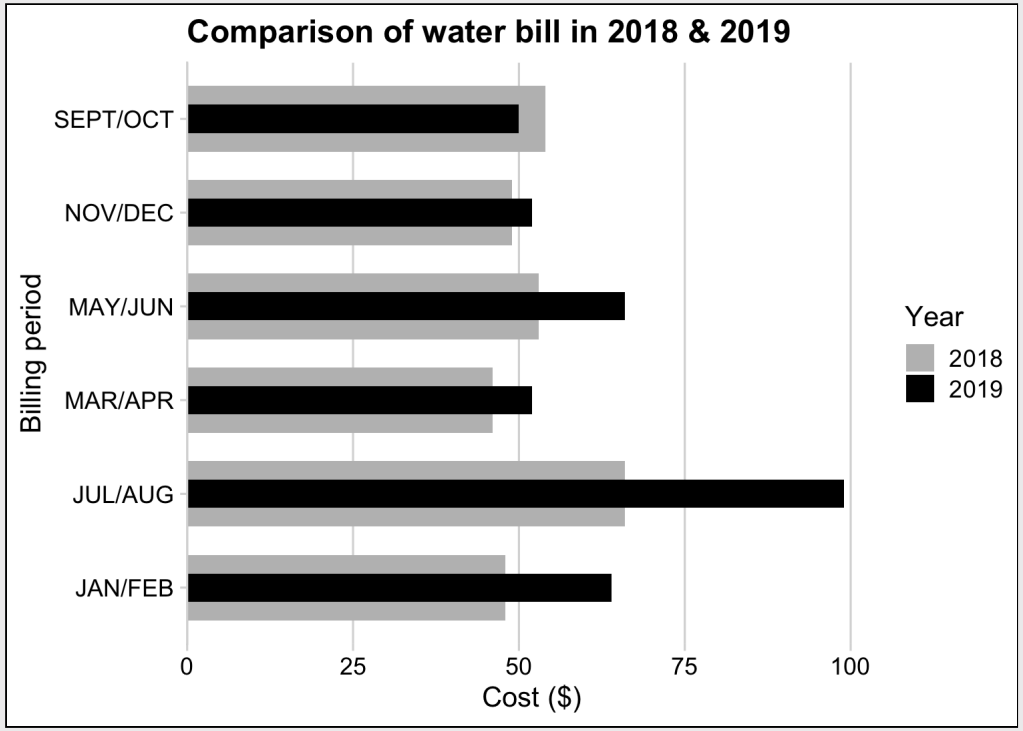
```
ggplot(gapminder_diverging) +
  geom_col(aes(x = country, y = lifeExp,
               fill = color),
           width = 0.7, alpha = 0.8) +
  scale_fill_manual(values = c('steelblue', 'red')) +
  coord_flip() +
  theme_minimal_vgrid() +
  theme(legend.position = 'none') +
  labs(
    x = 'Country',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



How to make overlapping bars

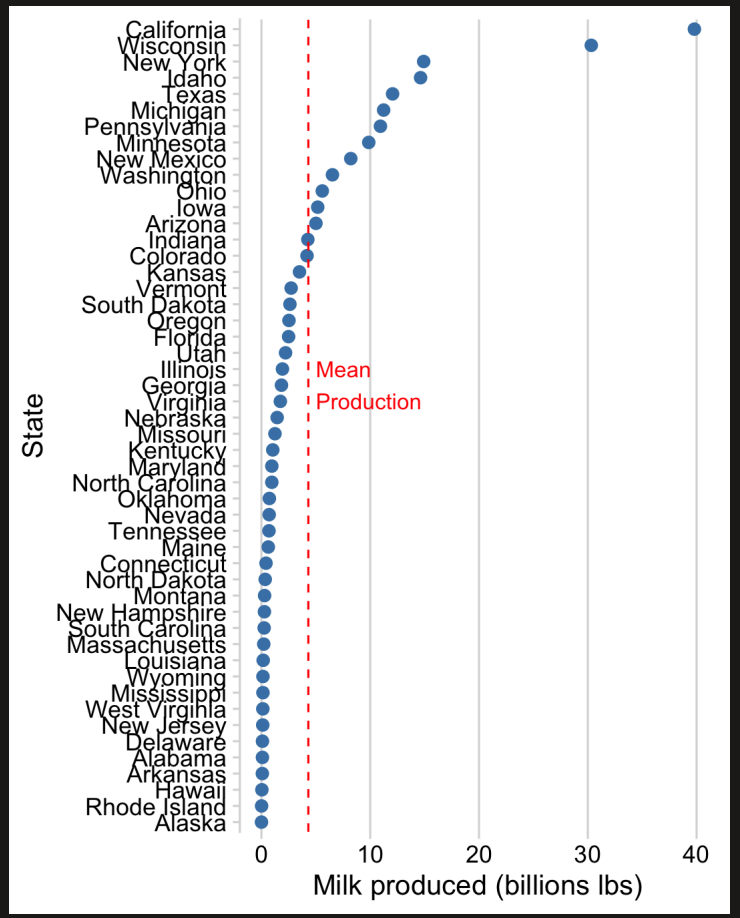
```
## # A tibble: 6 x 3
##   period   current previous
##   <chr>      <dbl>    <dbl>
## 1 JAN/FEB       64       48
## 2 MAR/APR       52       46
## 3 MAY/JUN       66       53
## 4 JUL/AUG       99       66
## 5 SEPT/OCT      50       54
## 6 NOV/DEC       52       49
```

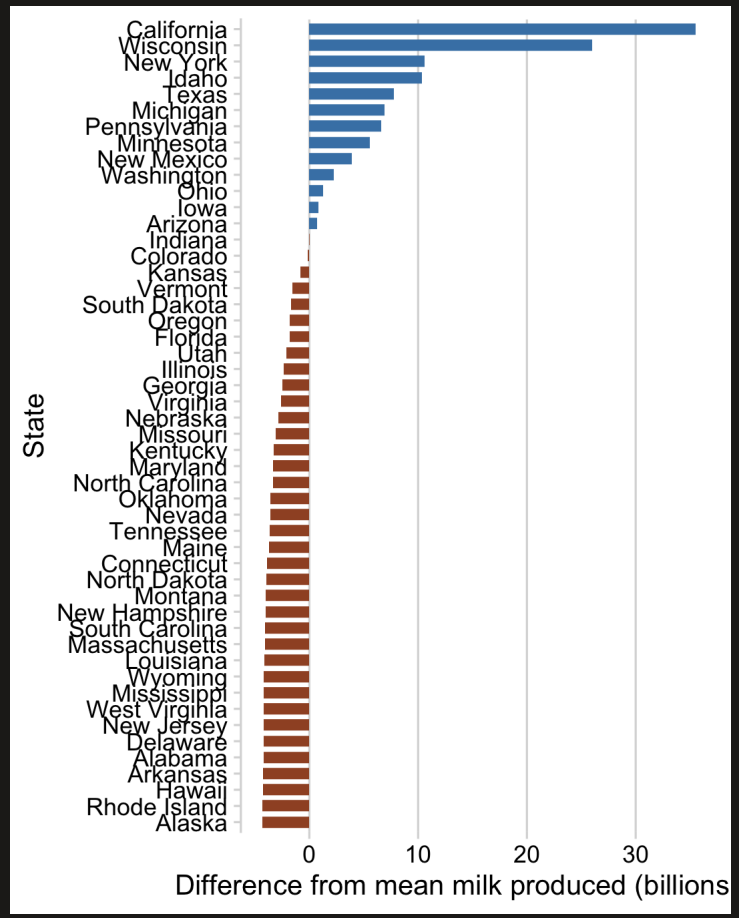
```
ggplot(water_usage) +
    geom_col(aes(x = period, y = previous,
                 fill = '2018'),
             width = 0.7) +
    geom_col(aes(x = period, y = current,
                 fill = '2019'),
             width = 0.3) +
    scale_fill_manual(values = c('grey', 'black')) +
    scale_y_continuous(
        expand = expand_scale(mult = c(0, 0.05))) +
    coord_flip() +
    theme_minimal_vgrid() +
    labs(x = 'Billing period',
         y = 'Cost ($)',
         fill = 'Year',
         title = 'Comparison of water bill in 2018 & 2019')
```



Your turn

Use the `milk_production.csv` data to create the following charts showing differences from the mean state milk production in 2017.





5 minute break!

Stand up

Move around

Stretch!

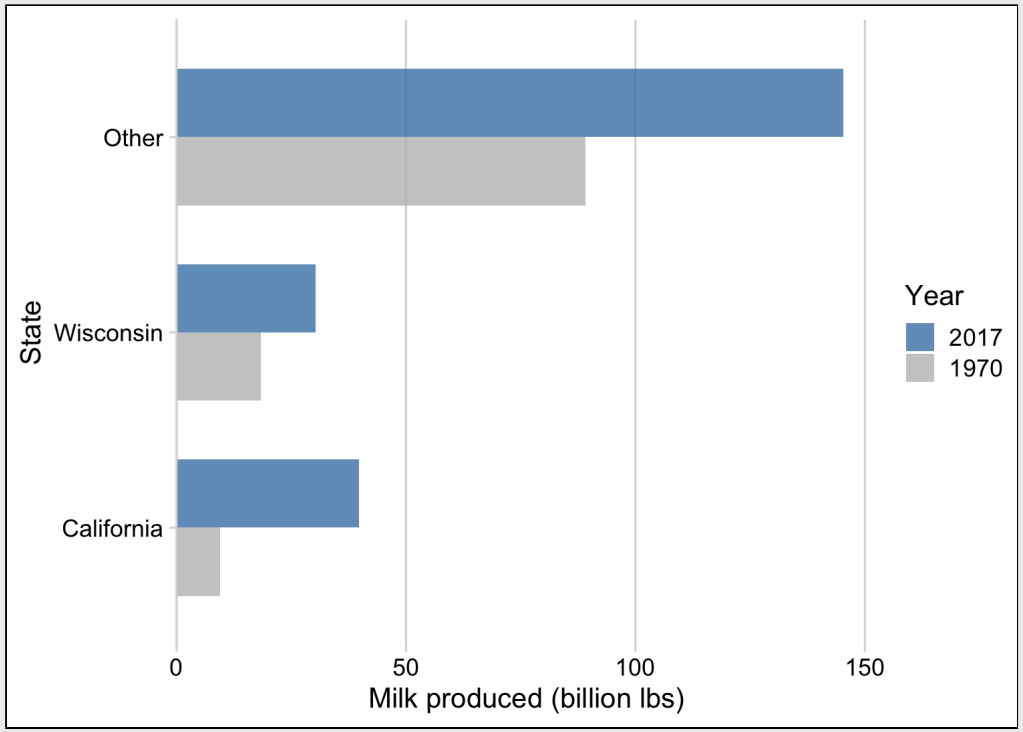
Graphing comparisons

1. Ranking things
2. Comparing things to a reference
3. Comparing two things
4. Comparing distributions

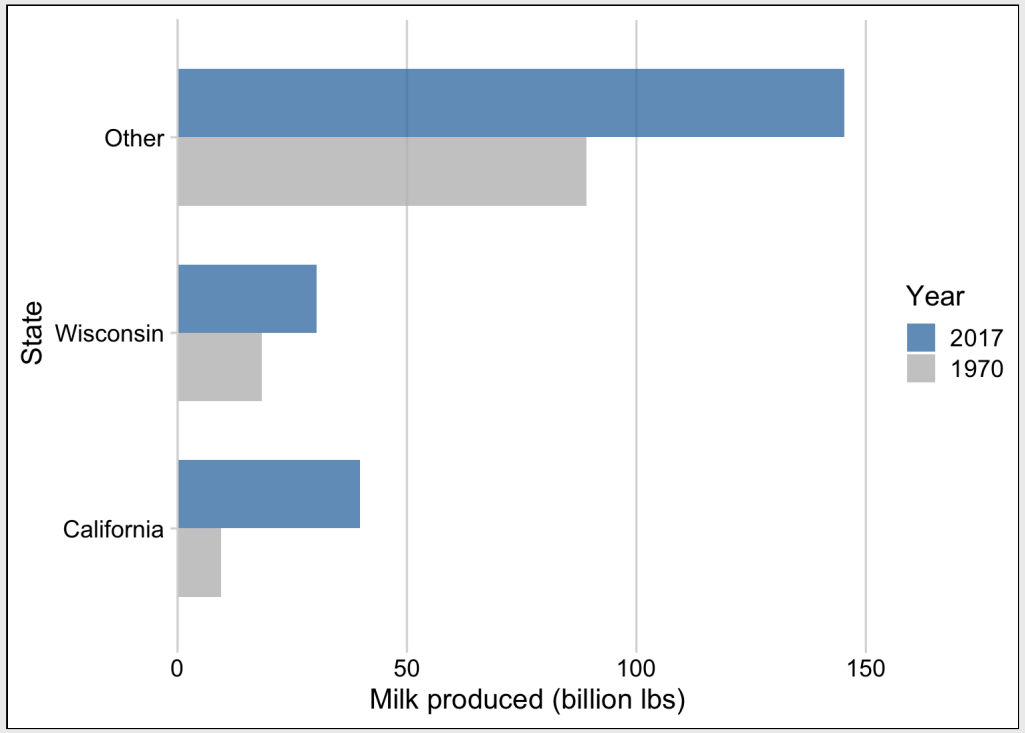
When comparing **only 2** things, dodged bars are a good starting point

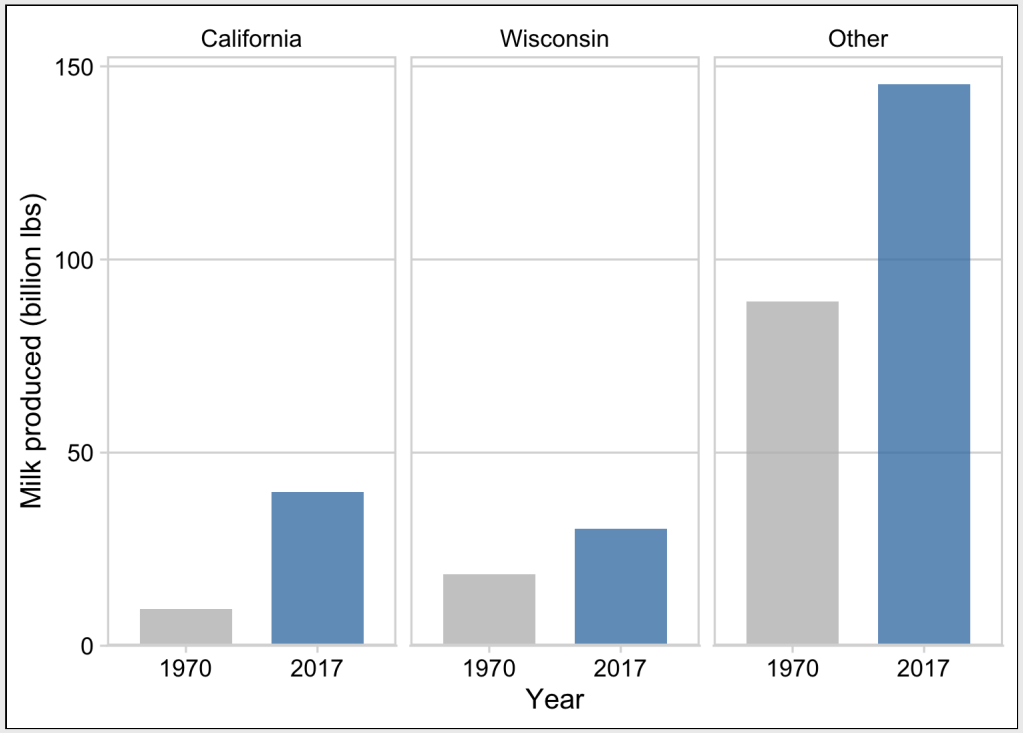
```
milk_compare <- milk_production %>%
  filter(year %in% c(1970, 2017)) %>%
  mutate(state = fct_other(state,
    keep = c('California', 'Wisconsin'))) %>%
  group_by(year, state) %>%
  summarise(milk_produced = sum(milk_produced) / 10^9)

milk_compare
```

```
## # A tibble: 6 x 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.
```



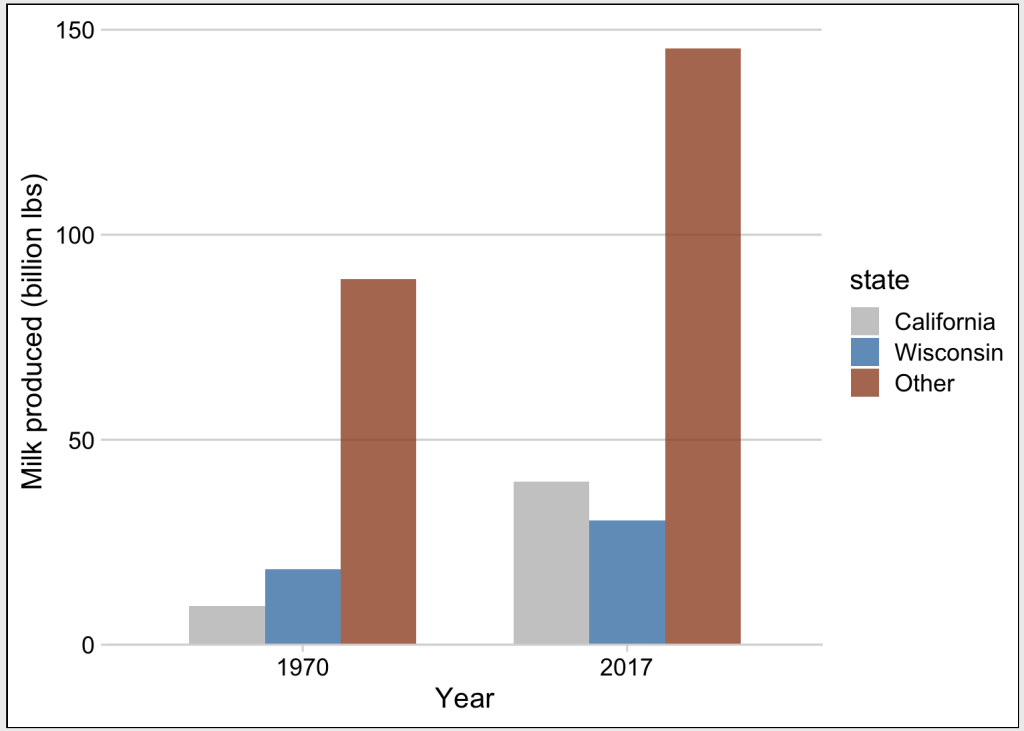
When comparing **only 2** things, dodged bars are a good starting point ...and facets can be helpful



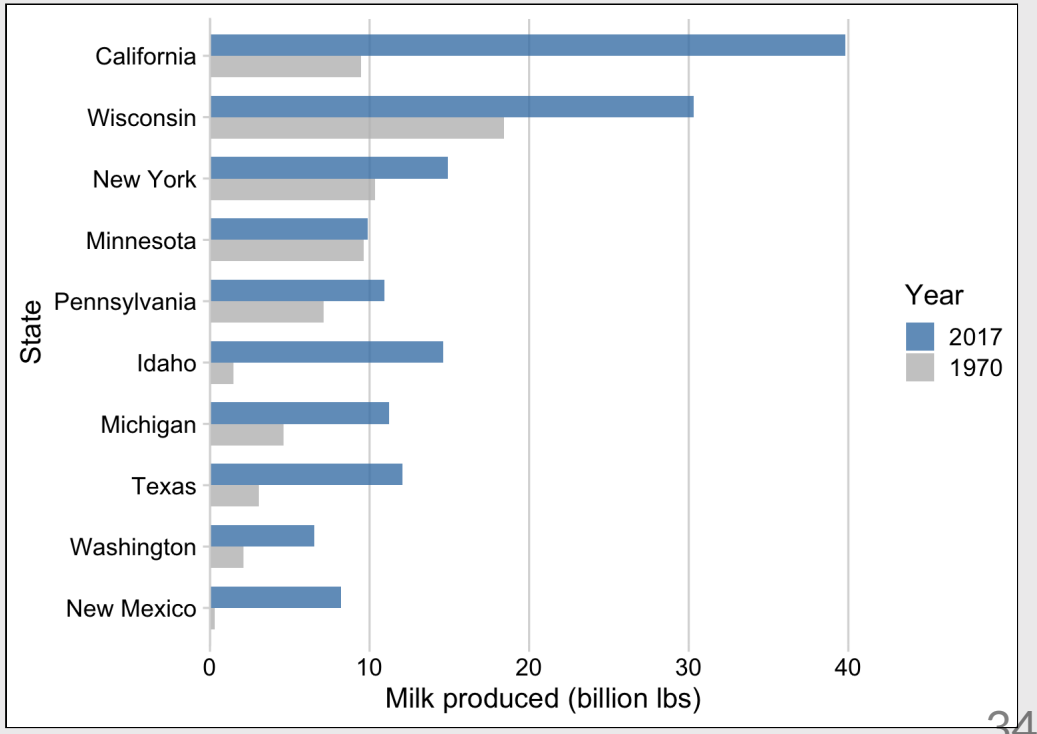


With **more than 2** things, dodged bars start to get confusing

Still comparing 2 time periods, but in groups of **3**

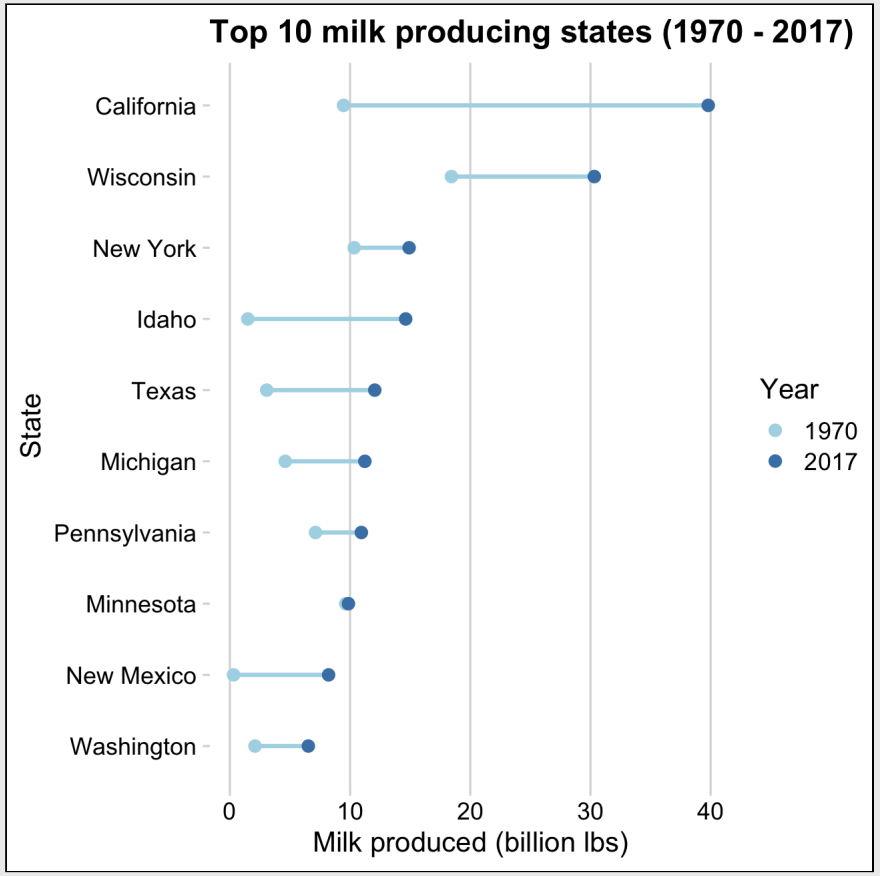


Still comparing 2 time periods, but across **10** categories

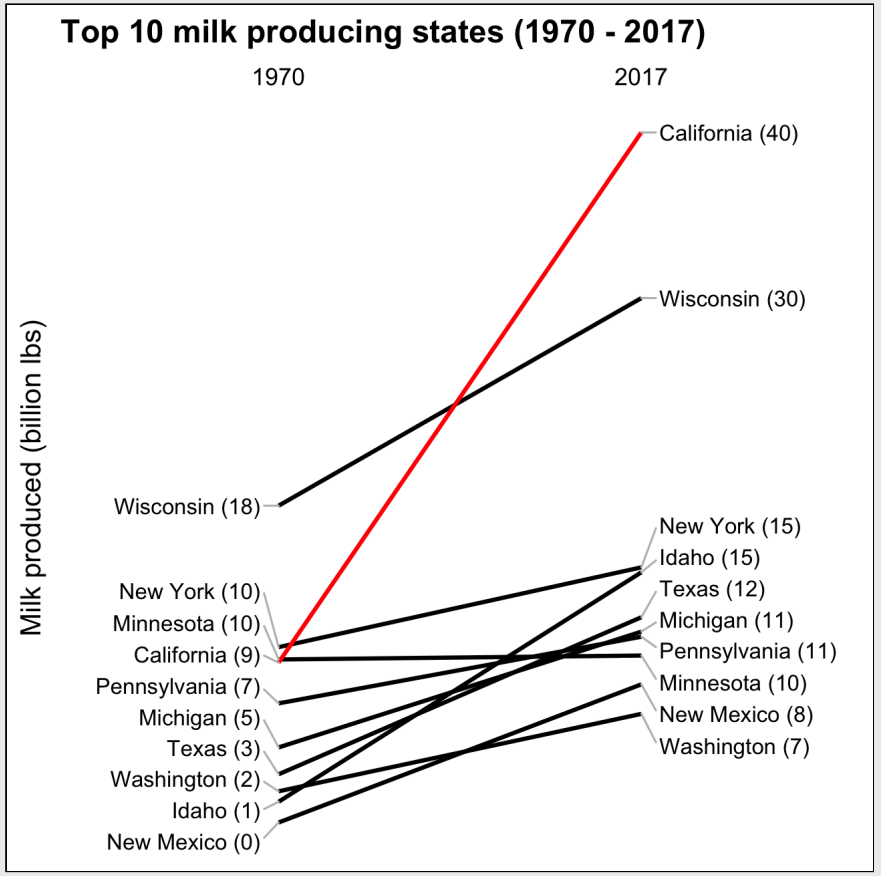


Charts for comparing change between 2 things across *more than* 2 categories

Dumbbell chart

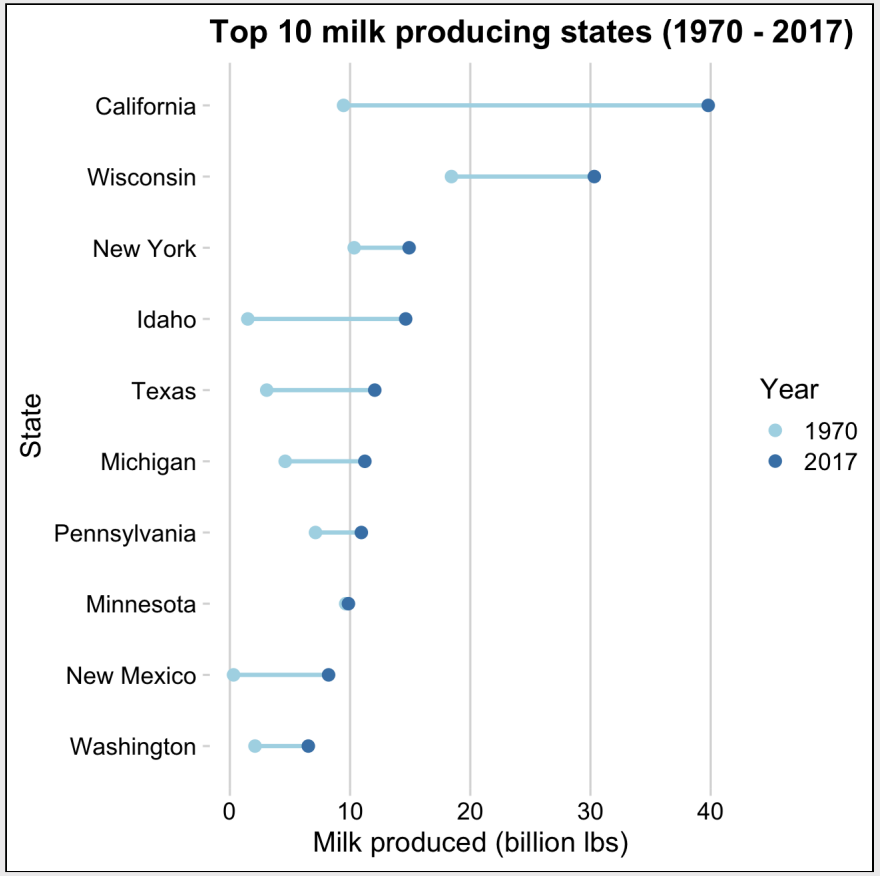


Slope chart



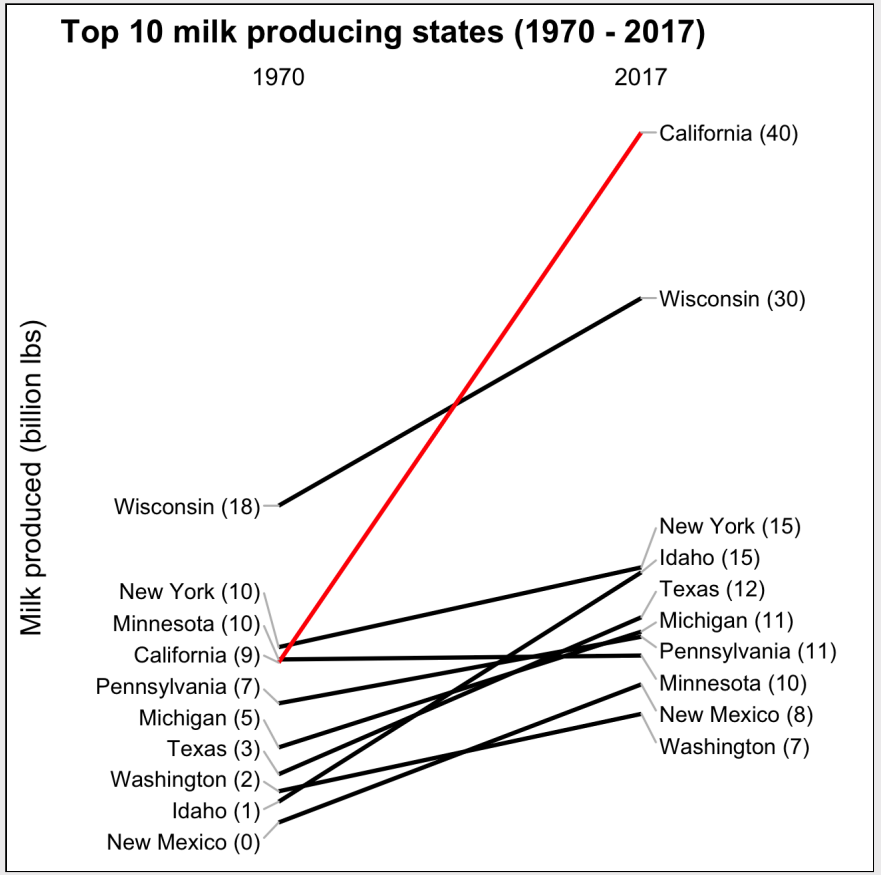
Dumbbell charts highlight...

- ...change in *magnitudes* across two periods / groups



Slope charts highlight...

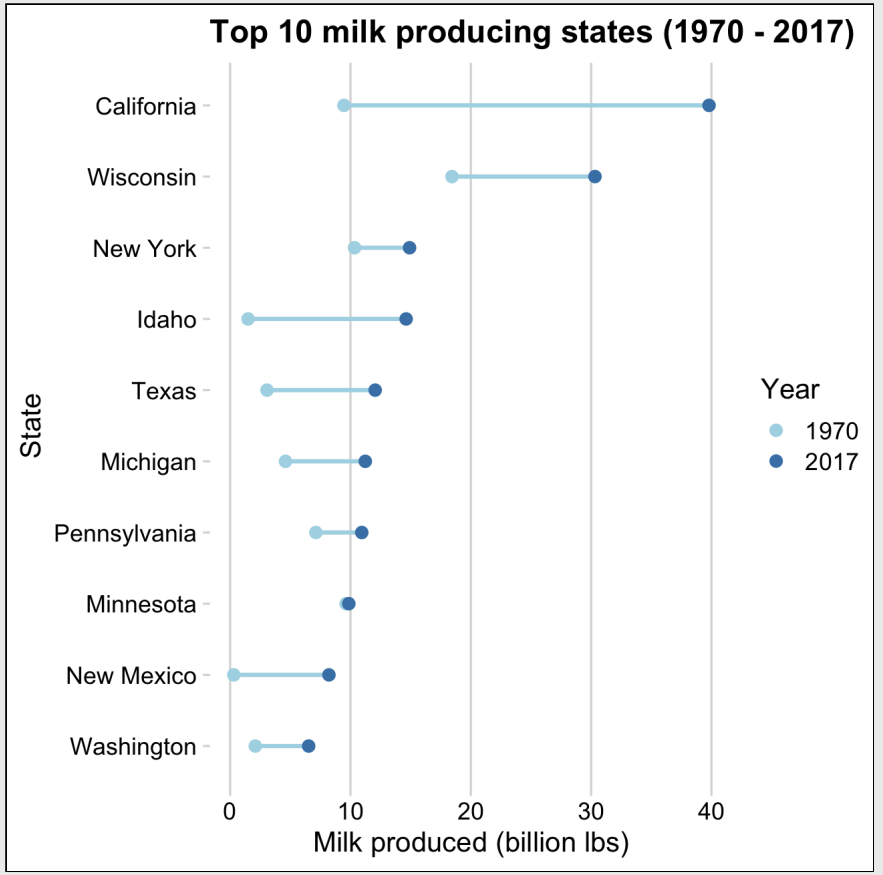
- ...change in *ranking* across two periods / groups
- ...how one thing is different from another



How to make a **Dumbbell chart**

Create data frame for plotting

```
# Identify the top 10 states (by 2017 production)
top10states <- milk_production %>%
    filter(year == 2017) %>%
    arrange(desc(milk_produced)) %>%
    slice(1:10)

# Now make the plot data frame
milk_summary_dumbbell <- milk_production %>%
    filter(
        year %in% c(1970, 2017),
        state %in% top10states$state) %>%
    mutate(
        # Reorder state variables
        state = fct_relevel(
            state, rev(top10states$state)),
        # Convert year to discrete variable
        year = as.factor(year),
        # Modify the units
        milk_produced = milk_produced / 10^9)
```

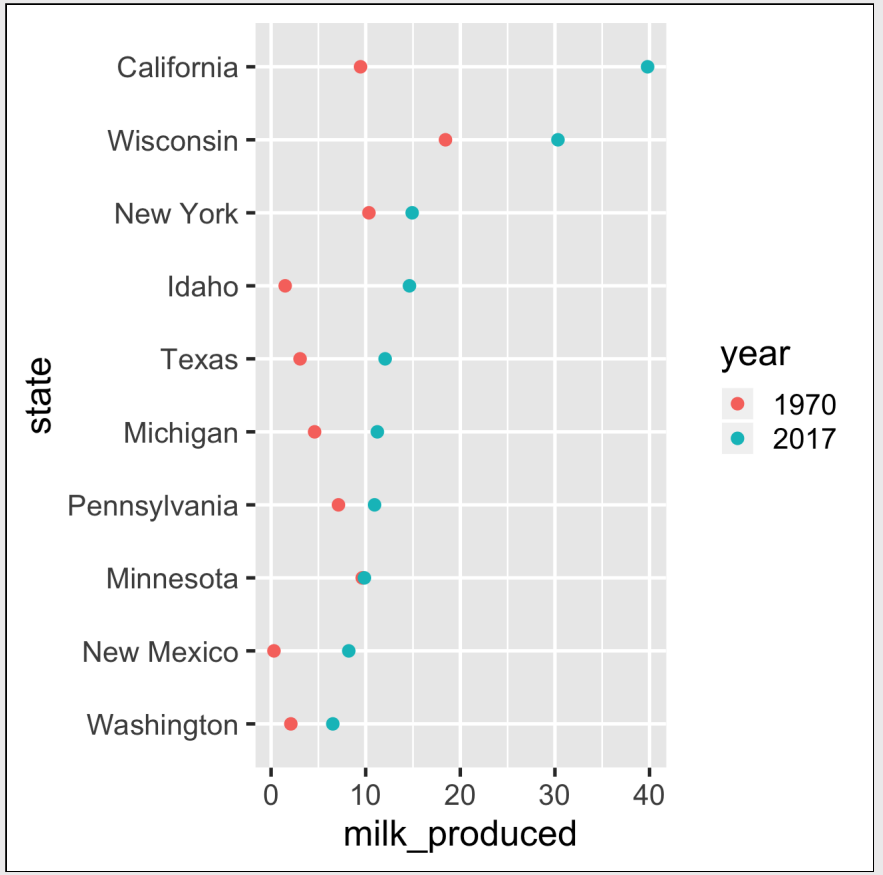


How to make a **Dumbbell chart**

Make the plot

Start with points:

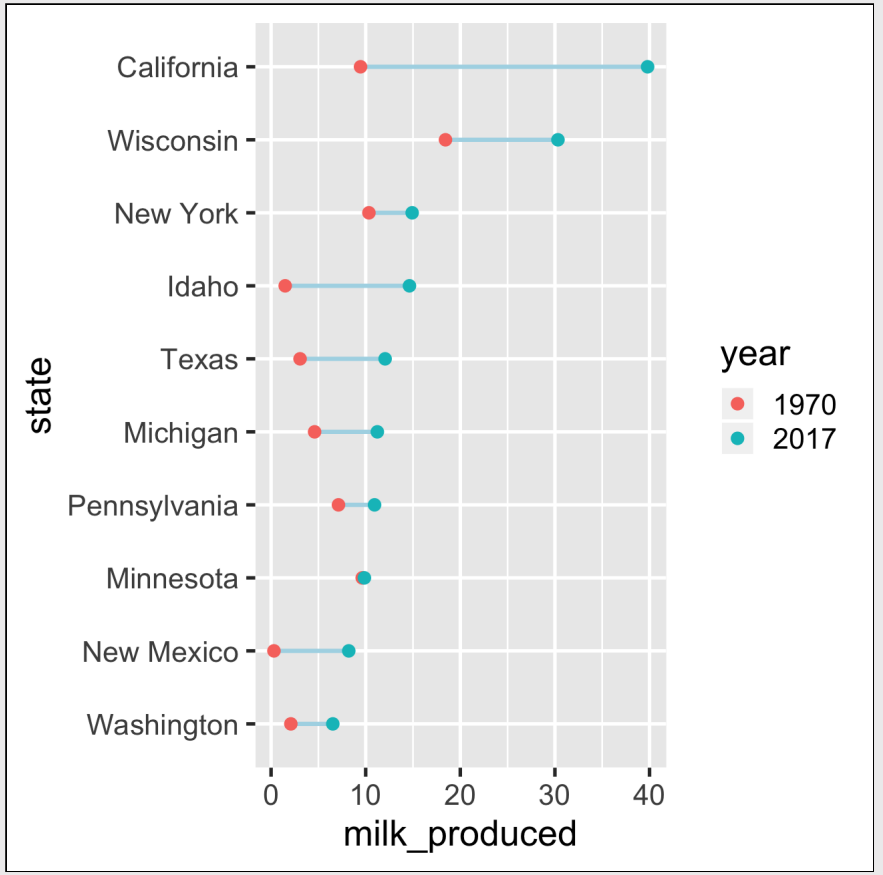
```
ggplot(milk_summary_dumbbell,
       aes(x = milk_produced, y = state)) +
  geom_point(aes(color = year), size = 2.5)
```



How to make a **Dumbbell chart**

Add lines - note the `group` variable:

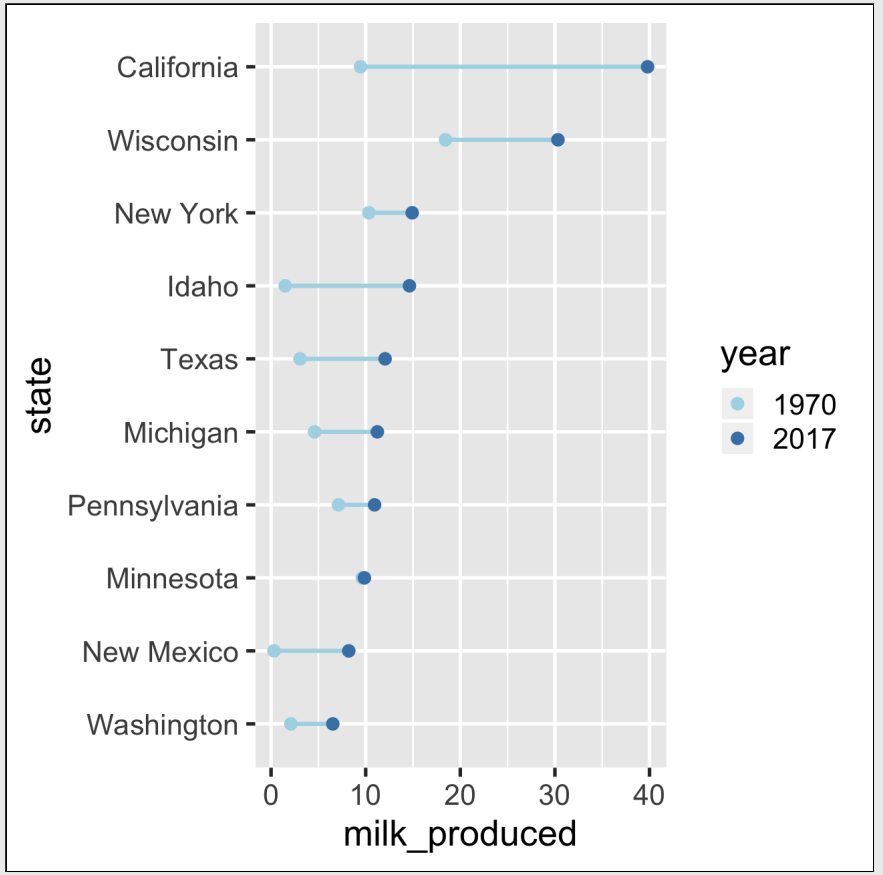
```
ggplot(milk_summary_dumbbell,
       aes(x = milk_produced, y = state)) +
  geom_line(aes(group = state),
            color = 'lightblue', size = 1) +
  geom_point(aes(color = year), size = 2.5)
```



How to make a **Dumbbell chart**

Change the colors:

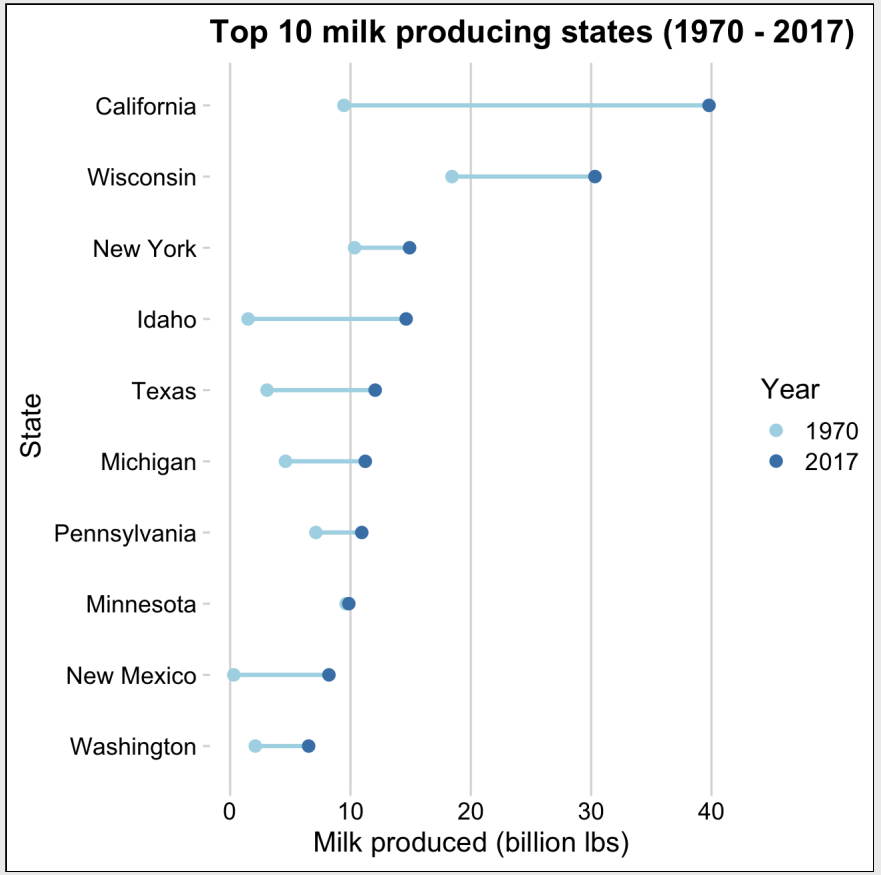
```
ggplot(milk_summary_dumbbell,
       aes(x = milk_produced, y = state)) +
  geom_line(aes(group = state),
            color = 'lightblue', size = 1) +
  geom_point(aes(color = year), size = 2.5) +
  scale_color_manual(
      values = c('lightblue', 'steelblue'))
```



How to make a **Dumbbell chart**

Adjust the theme and annotate:

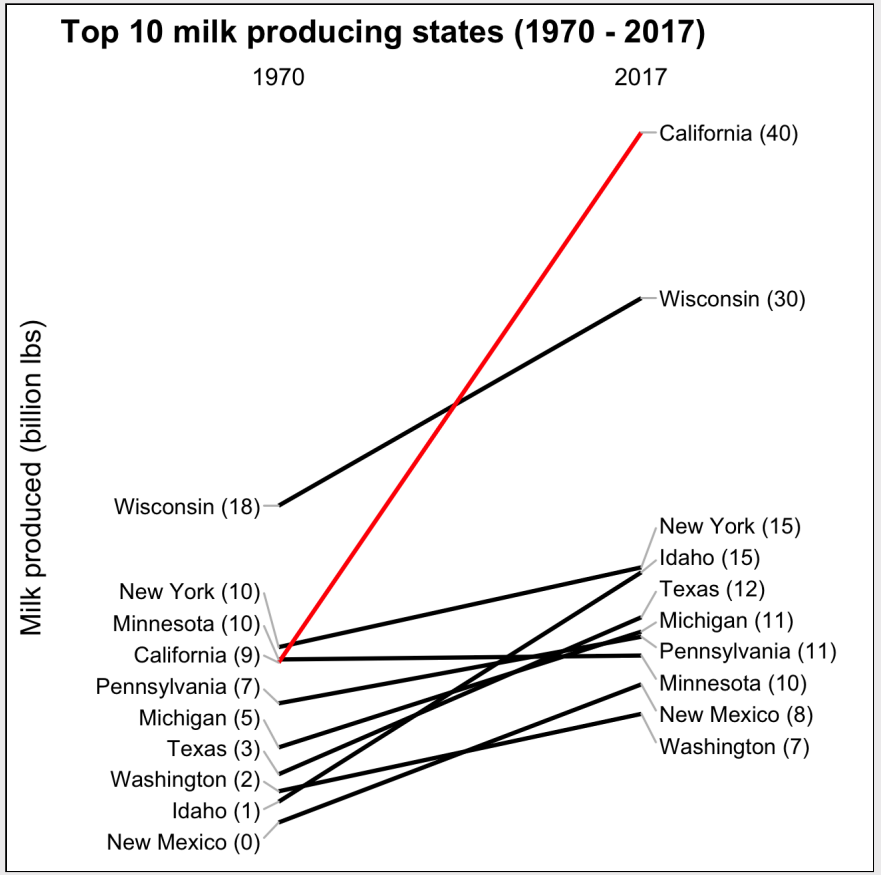
```
ggplot(milk_summary_dumbbell,
       aes(x = milk_produced, y = state)) +
  geom_line(aes(group = state),
            color = 'lightblue', size = 1) +
  geom_point(aes(color = year), size = 2.5) +
  scale_color_manual(
      values = c('lightblue', 'steelblue')) +
  theme_minimal_vgrid() +
  # Remove y axis line
  theme(axis.line.y = element_blank()) +
  labs(x = 'Milk produced (billion lbs)',
       y = 'State',
       color = 'Year',
       title = 'Top 10 milk producing states (1970 - 2
```



Create data frame for plotting

```
# Identify the top 10 states (by 2017 production)
top10states <- milk_production %>%
    filter(year == 2017) %>%
    arrange(desc(milk_produced)) %>%
    slice(1:10)

# Now make the plot data frame
milk_summary_slope <- milk_production %>%
    filter(
        year %in% c(1970, 2017),
        state %in% top10states$state) %>%
    mutate(
        # Modify the units
        milk_produced = milk_produced / 10^9,
        # Reorder state variables to follow top 10 states
        state = fct_relevel(state, rev(top10states$state)),
        # Make year a factor
        year = as.factor(year),
        # Create variable for the line color
        lineColor = if_else(
            state == 'California', 'CA', 'other'))
        # Create labels
        label = str_c(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

```
# Identify the top 10 states (by 2017 production)
top10states <- milk_production %>%
    filter(year == 2017) %>%
    arrange(desc(milk_produced)) %>%
    slice(1:10)

# Now make the plot data frame
milk_summary_slope <- milk_production %>%
    filter(
        year %in% c(1970, 2017),
        state %in% top10states$state) %>%
    mutate(
        # Modify the units
        milk_produced = milk_produced / 10^9,
        # Reorder state variables to follow top 10 states
        state = fct_relevel(state, rev(top10states$state)),
        # Make year a factor
        year = as.factor(year),
        # Create variable for the line color
        lineColor = if_else(
            state == 'California', 'CA', 'other'))
        # Create labels
        label = str_c(state, ' (',
                      round(milk_produced), ')'),
        label_left = ifelse(year == 1970, label, NA),
        label_right = ifelse(year == 2017, label, NA))
```

How to make a **Slope chart**

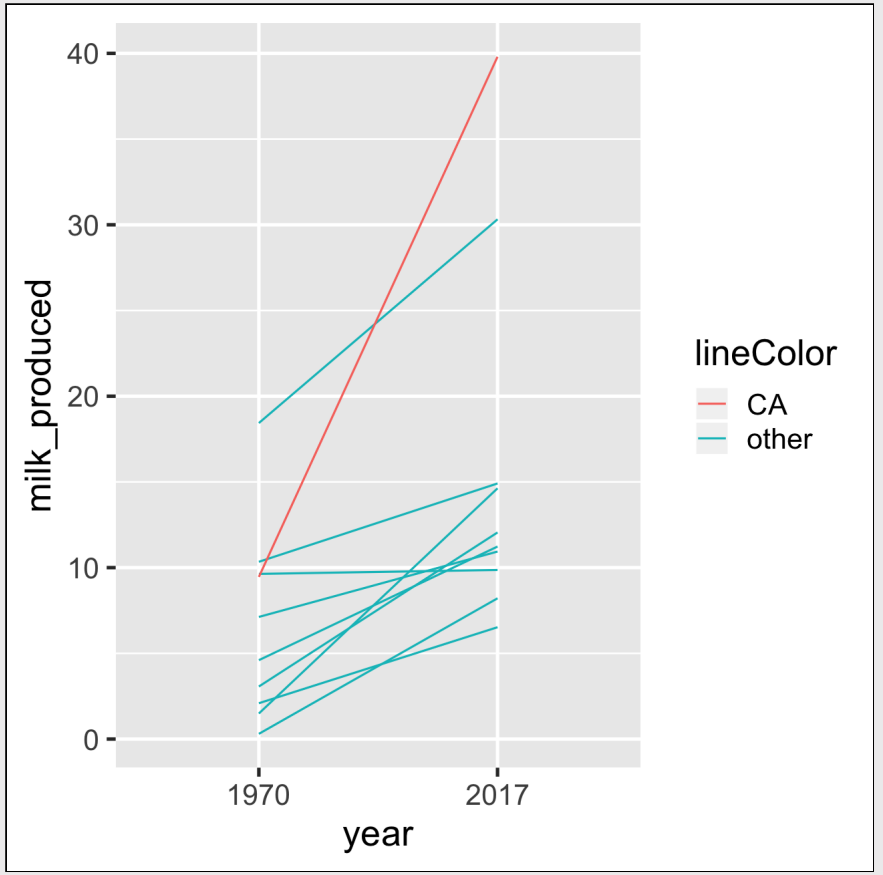
```
milk_summary_slope %>%
    select(state, year, milk_produced, label, lineColor)
```

```
## # A tibble: 20 x 5
##    state        year  milk_produced label            lineColor
##    <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:

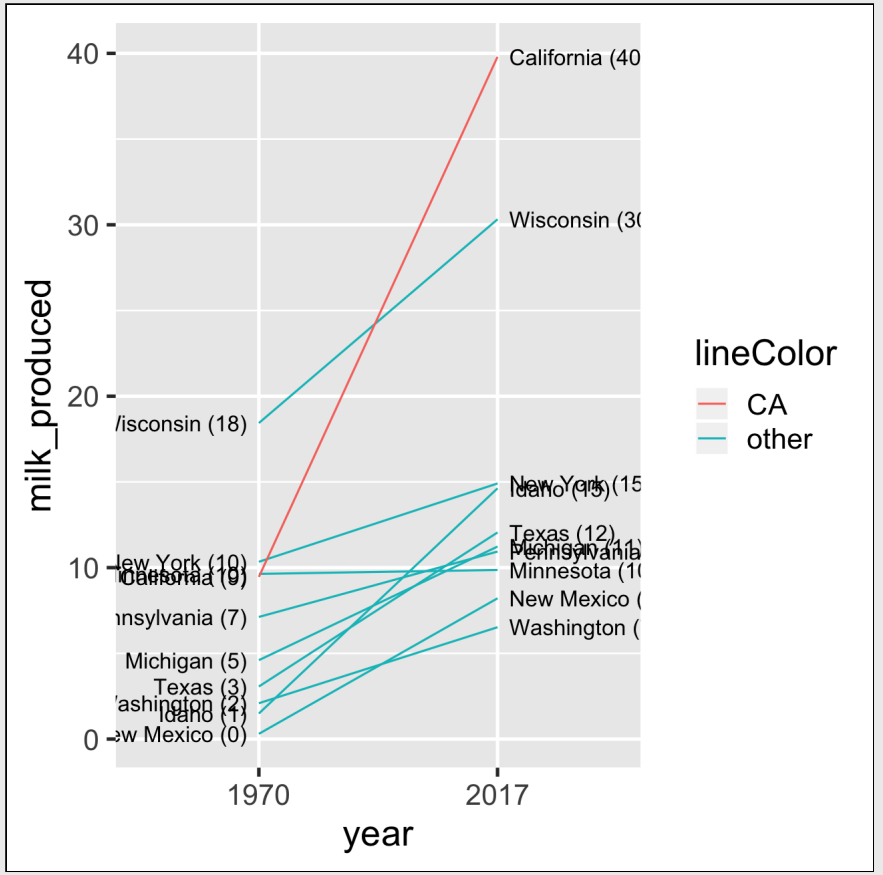
```
ggplot(milk_summary_slope,
       aes(x = year, y = milk_produced,
           group = state)) +
    geom_line(aes(color = lineColor))
```



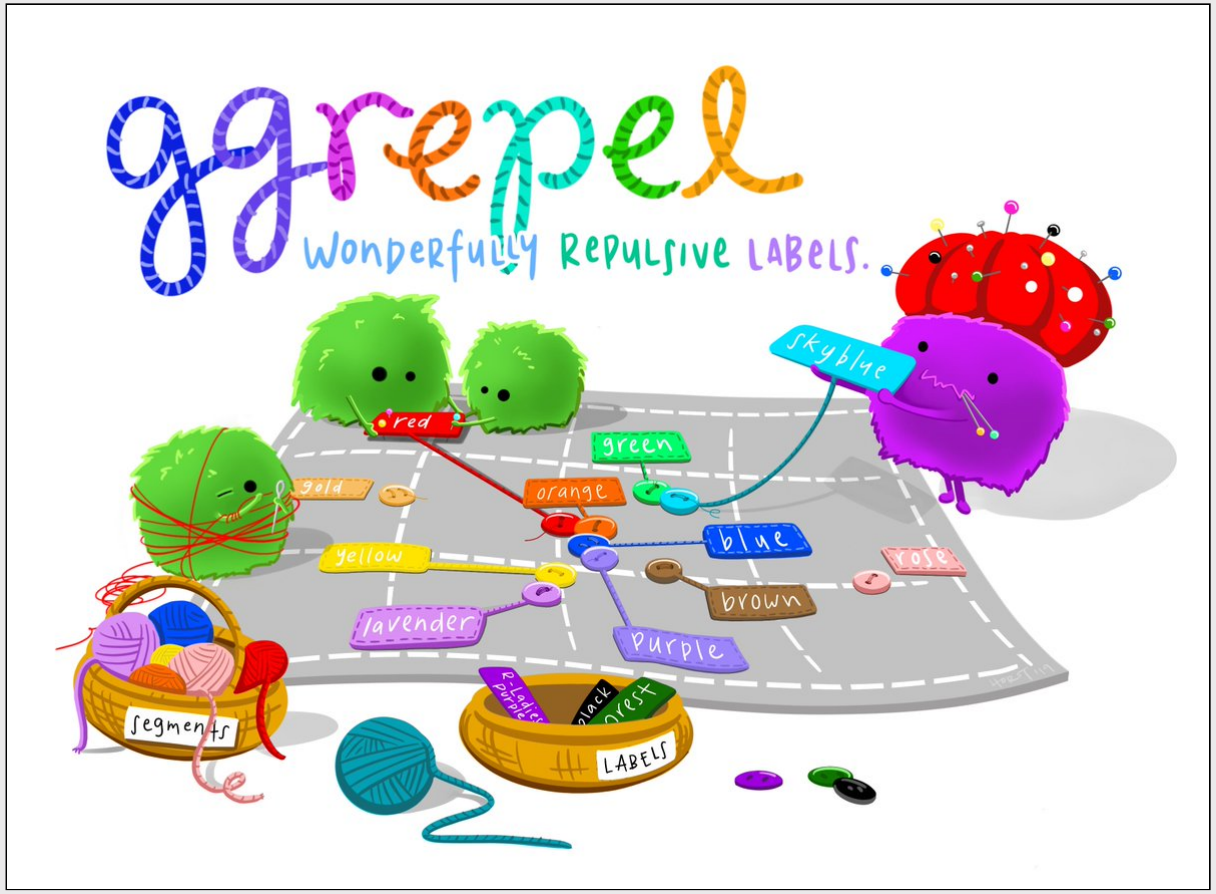
How to make a **Slope chart**

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)
```



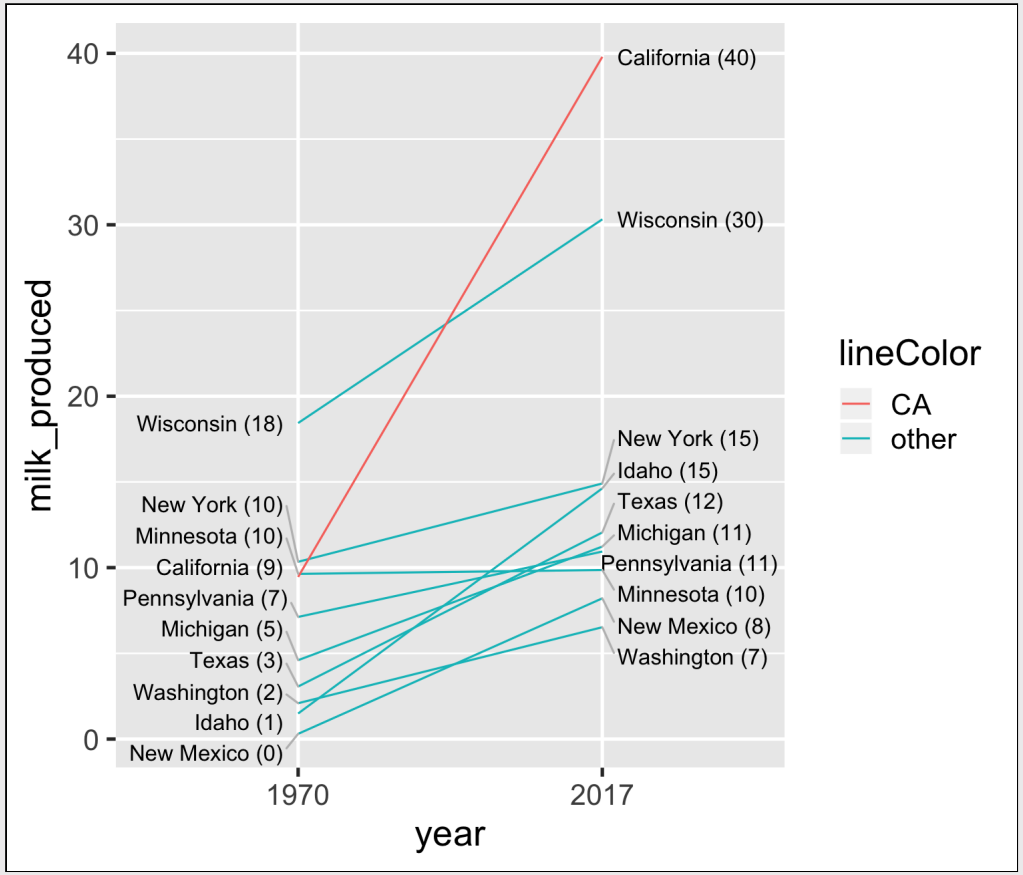
Overlapping labels?
ggrepel library to the rescue!



How to make a **Slope chart**

Align labels so they don't overlap:

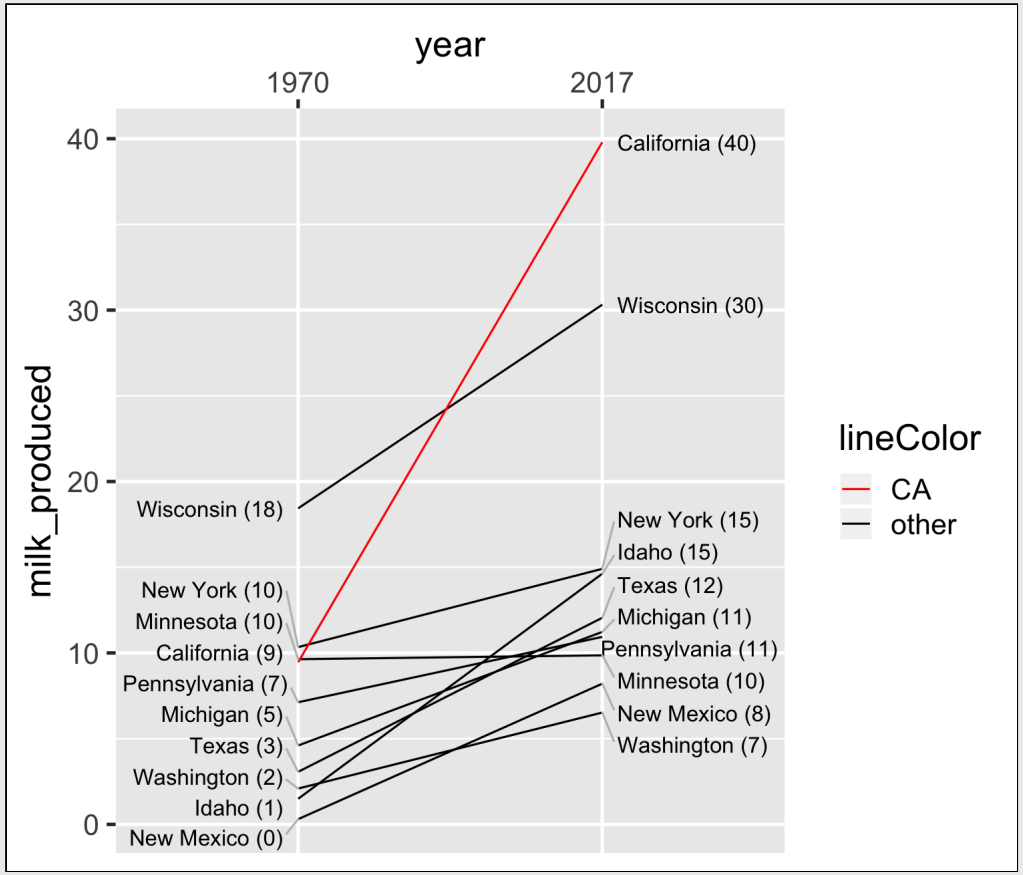
```
library(ggrepel)

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:

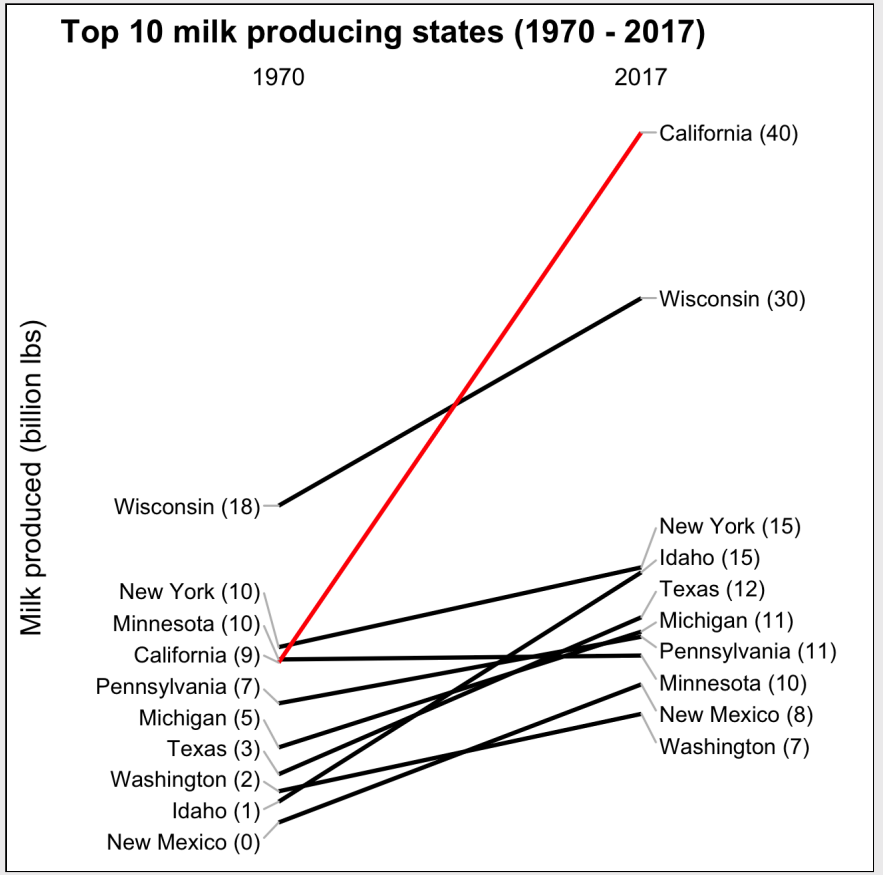
```
ggplot(milk_summary_slope,
       aes(x = year, y = milk_produced,
           group = state)) +
    geom_line(aes(color = lineColor)) +
    # Add 1970 labels (left side)
    geom_text_repel(aes(label = label_left),
                    hjust = 1, nudge_x = -0.05,
                    direction = 'y',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



How to make a **Slope chart**

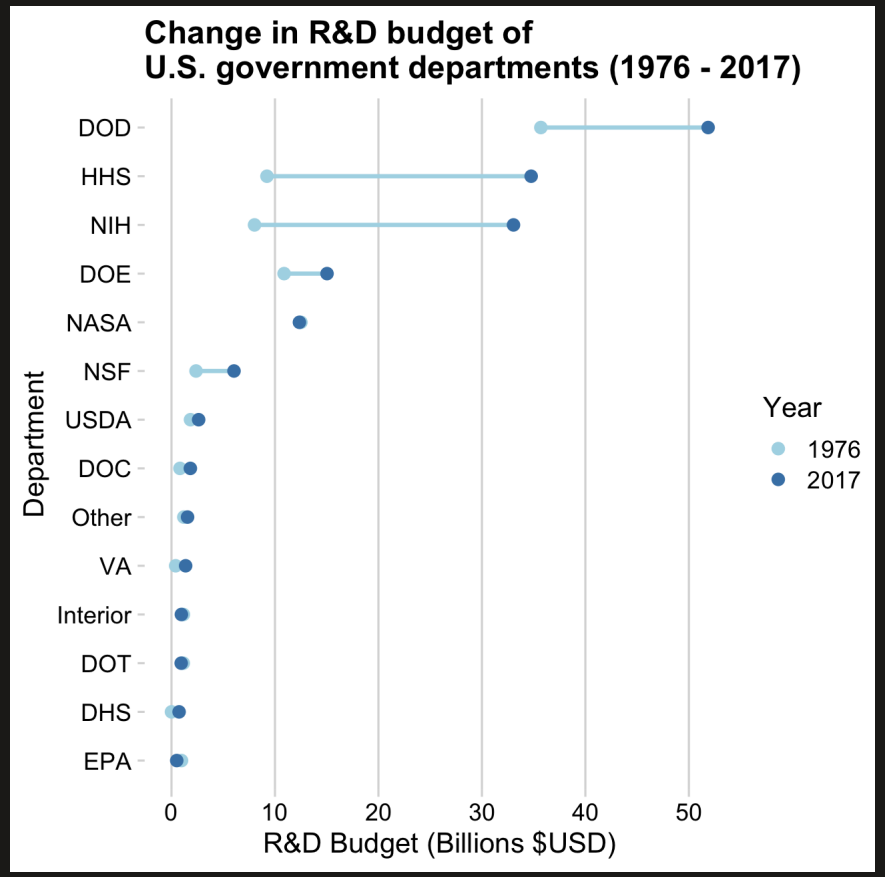
Adjust the theme and annotate

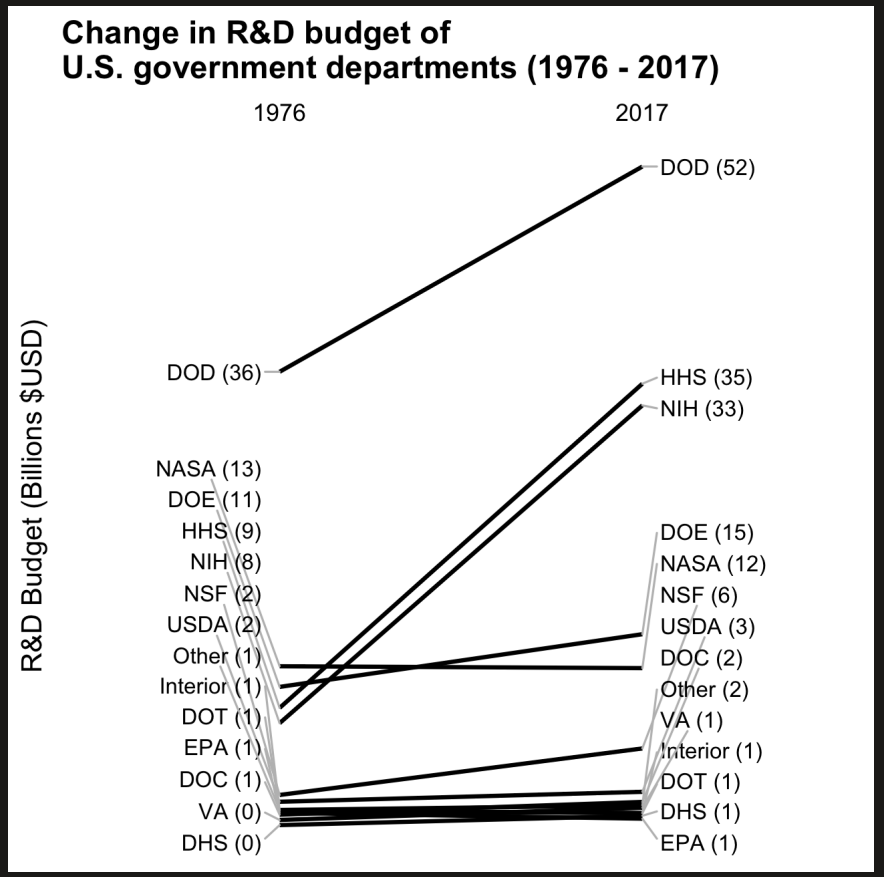
```
ggplot(milk_summary_slope,
       aes(x = year, y = milk_produced,
           group = state)) +
    geom_line(aes(color = lineColor)) +
    # Add 1970 labels (left side)
    geom_text_repel(aes(label = label_left),
                    hjust = 1, nudge_x = -0.05,
                    direction = 'y',
                    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 - 2017)') +
    theme_minimal_grid() +
    theme(panel.grid  = element_blank(),
          axis.text.y = element_blank(),
          axis.ticks = element_blank(),
          legend.position = 'none')
```



Your turn

Use the `federal_spending_long.csv` data to create the following charts.



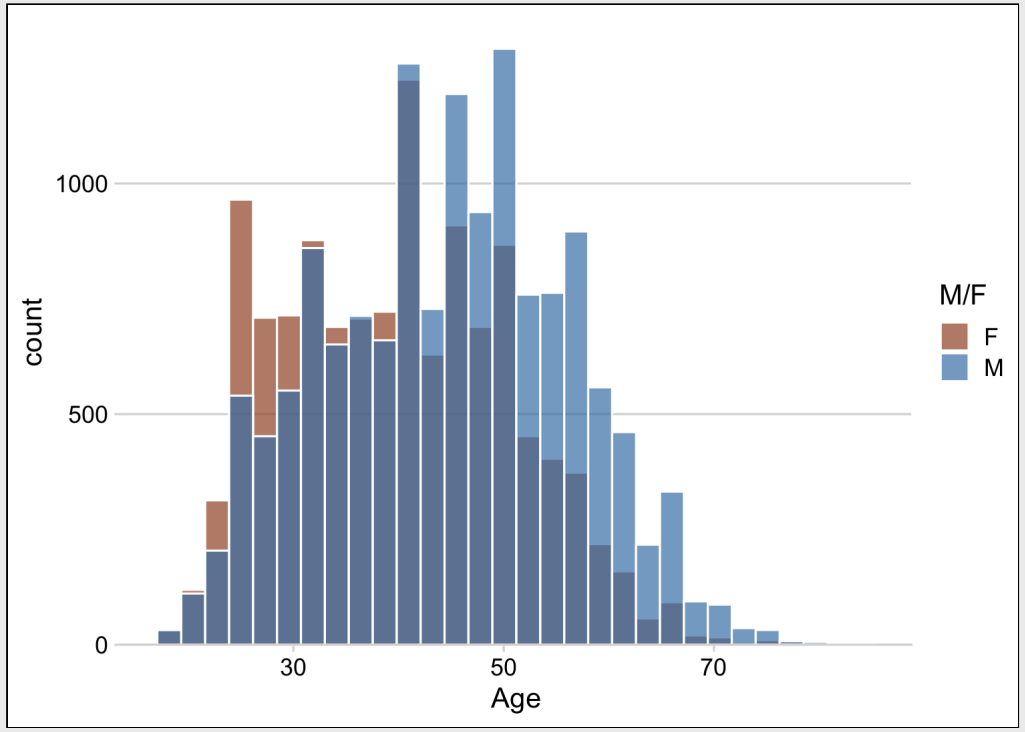


Graphing comparisons

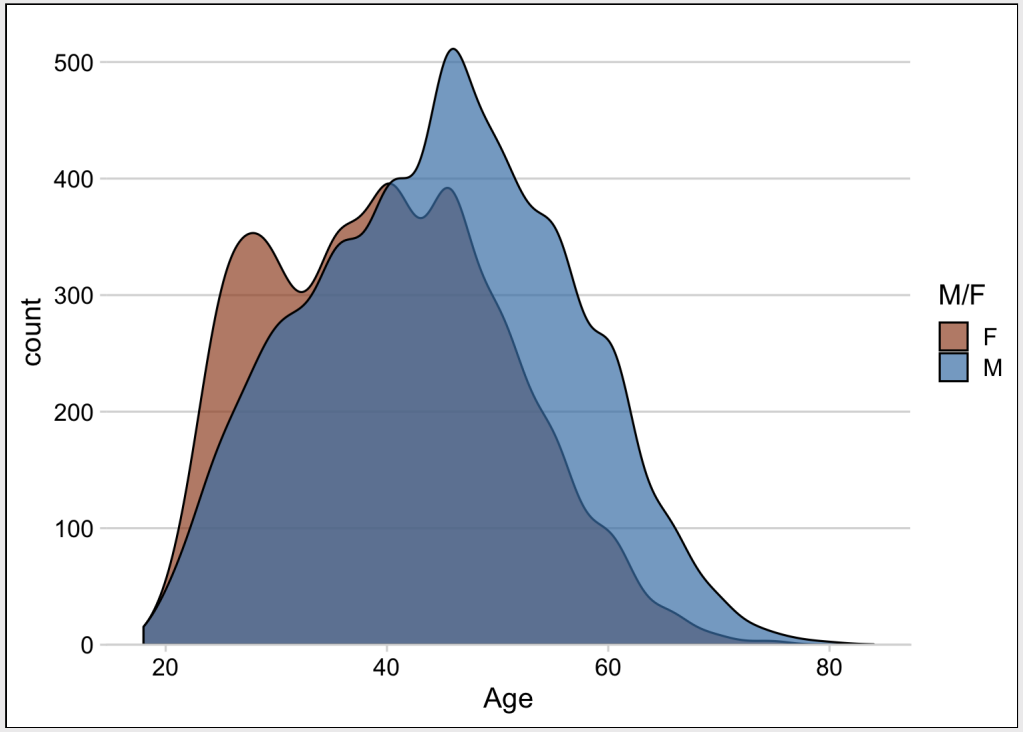
1. Ranking things
2. Comparing things to a reference
3. Comparing two things
4. Comparing distributions

Overlapping histograms have issues

Bad

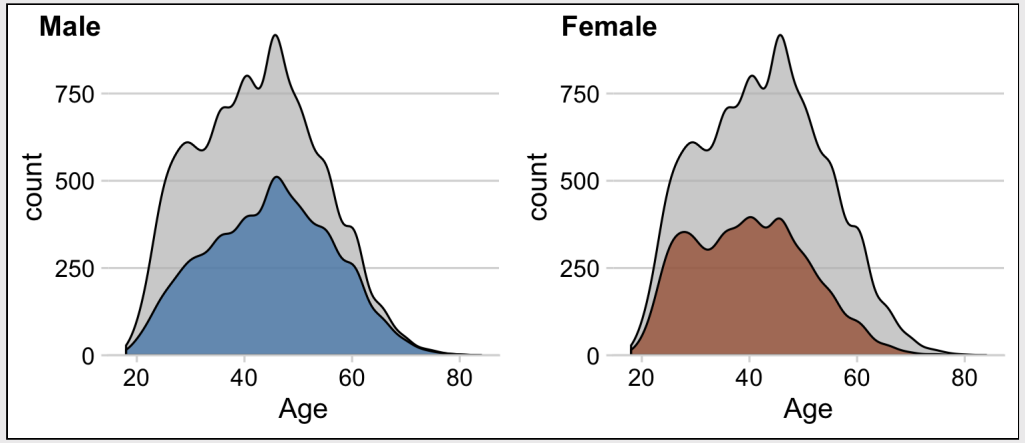


Slightly better

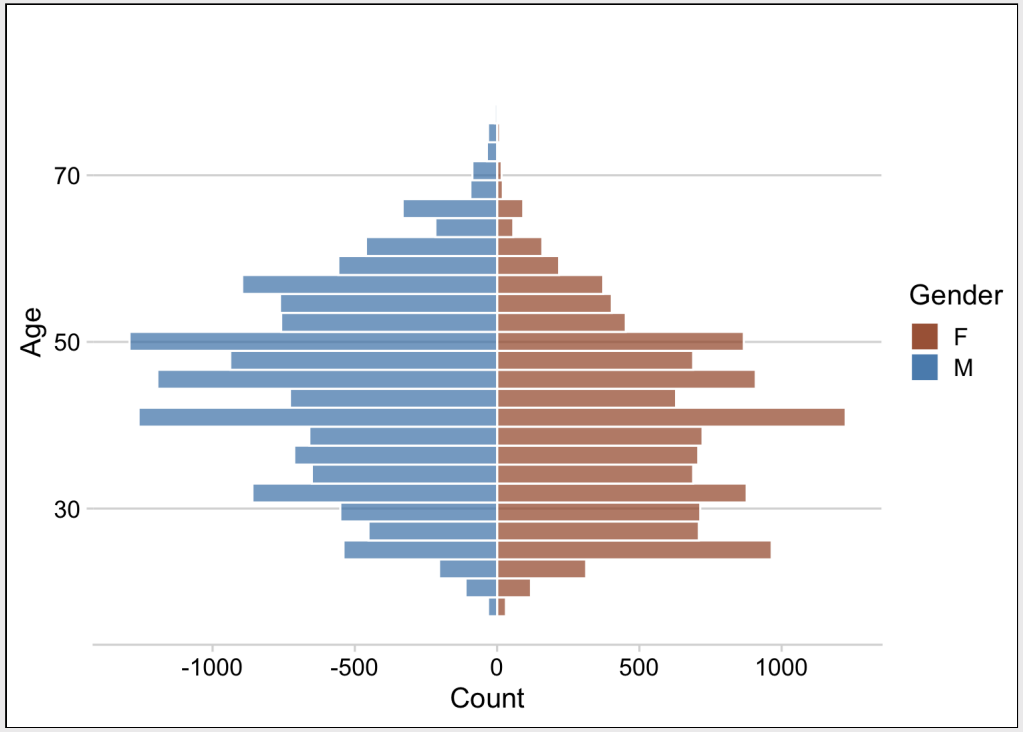


Good when # categories **small**

Density facets

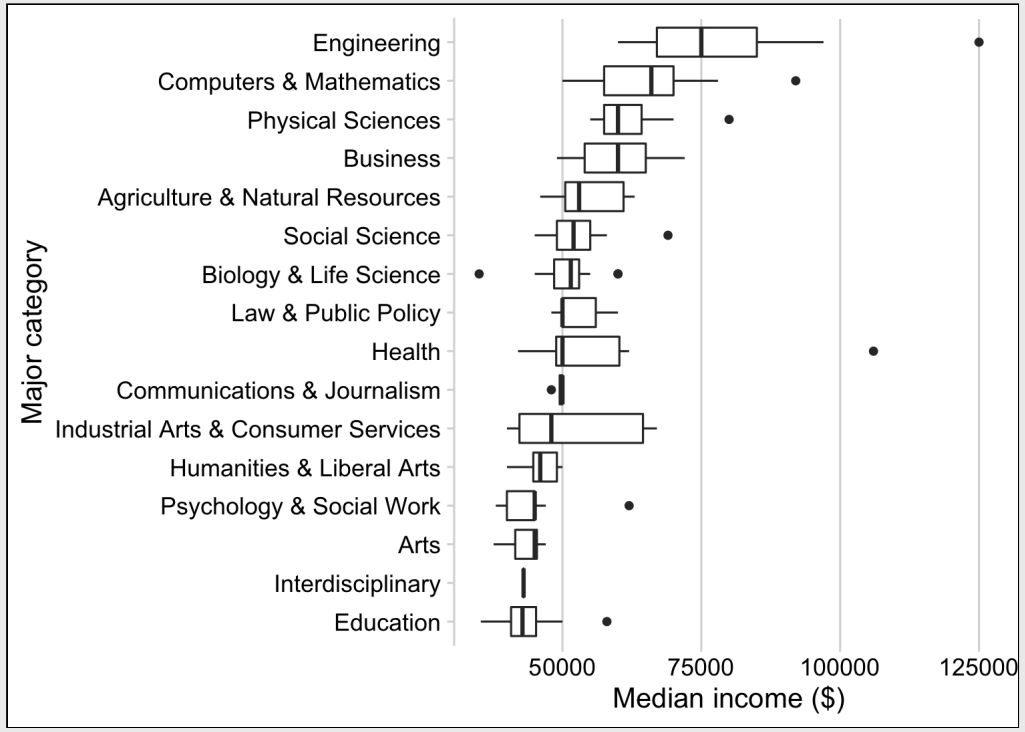


Diverging histograms

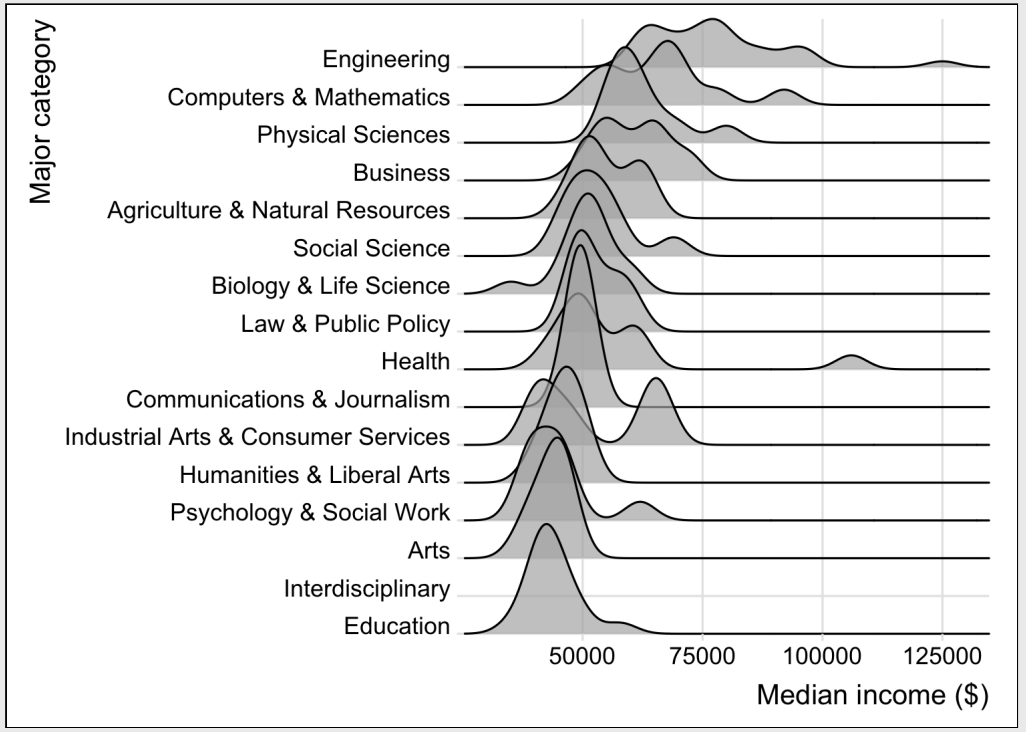


Good when # categories **large**

Boxplot



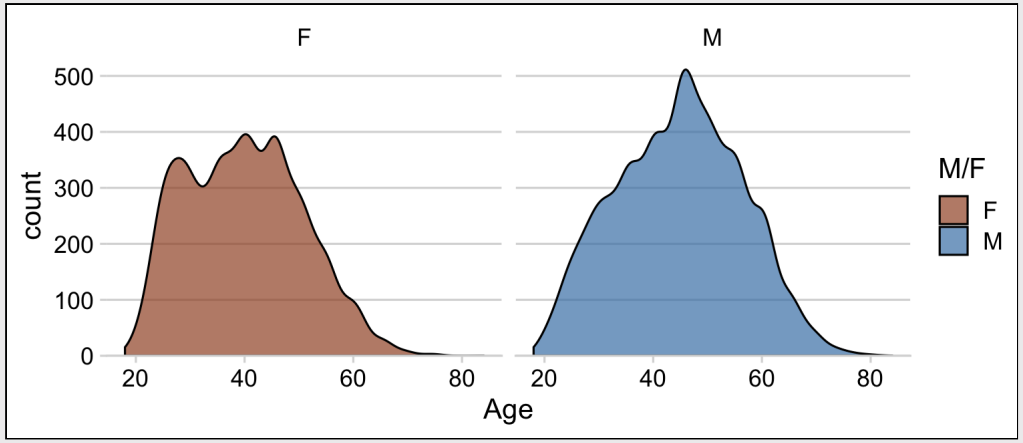
Ridgeplot



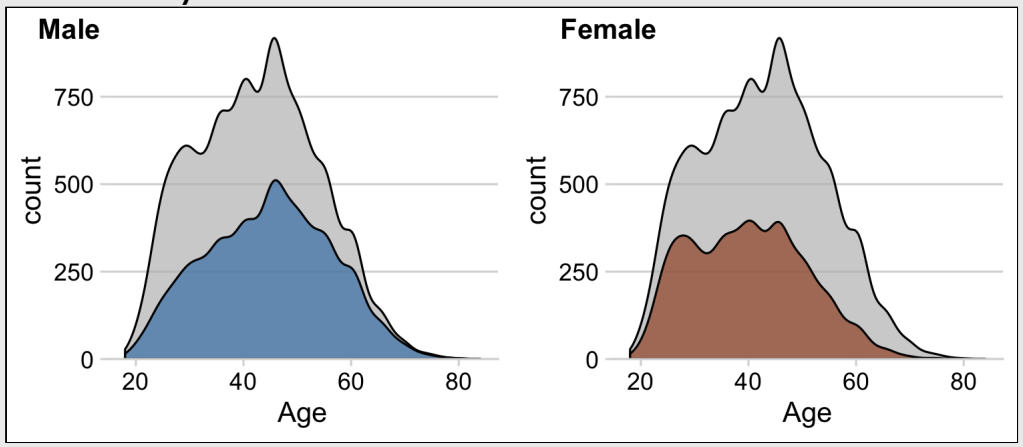
How to make density facets

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(~`M/F`) +
    scale_fill_manual(
        values = c('sienna', 'steelblue')) +
    scale_y_continuous(
        expand = expand_scale(mult = c(0, 0.05))) +
    theme_minimal_hgrid()
```



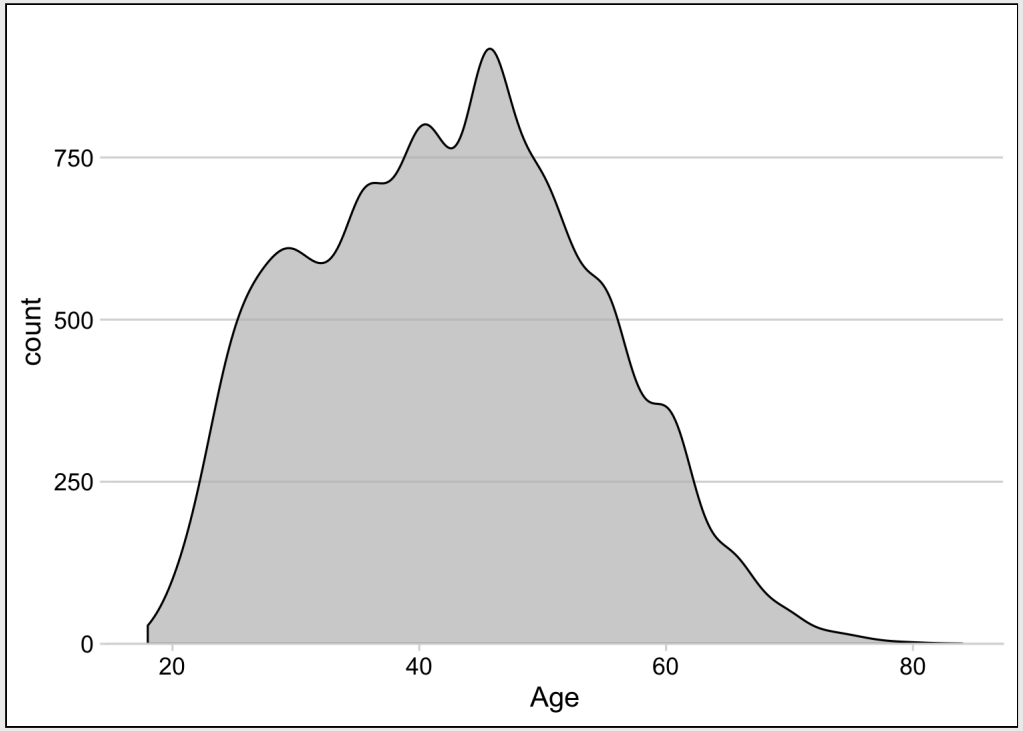
If you want the full density overlay,
you have to hand-make the facets



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 = expand_scale(mult = c(0, 0.05))) +
    theme_minimal_hgrid()
```

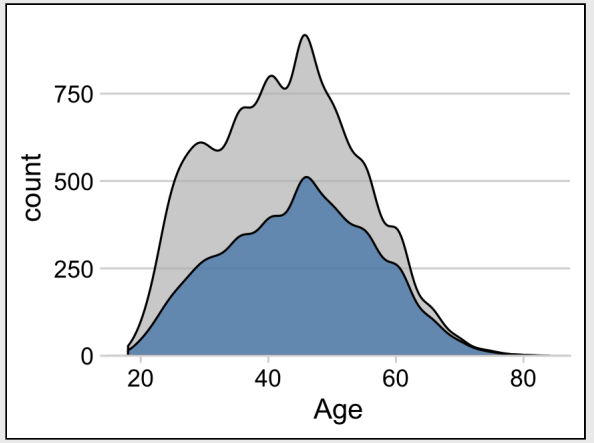


How to make density facets

Separately create each sub-plot

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

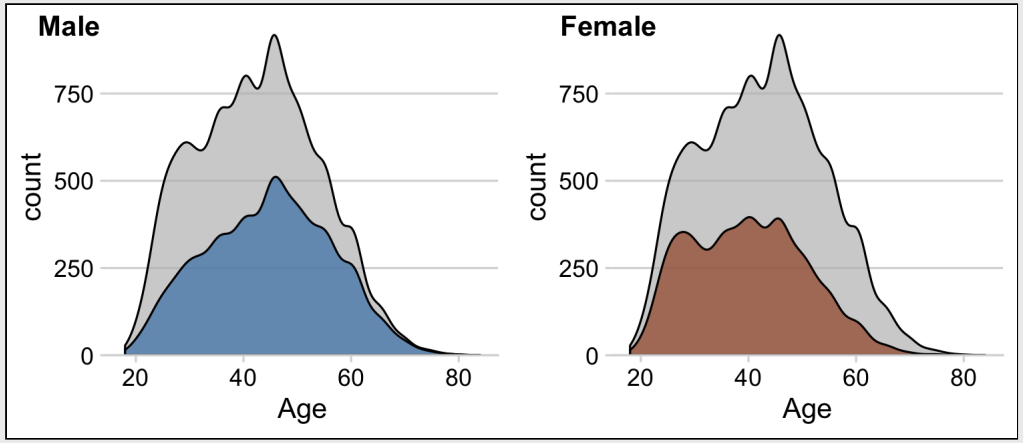
```
female <- base +
  geom_density(data = marathon %>%
                   filter(`M/F` == 'F'),
               fill = 'sienna', alpha = 0.7) +
  theme(legend.position = 'none')
```



How to make density facets

Combine into single plot

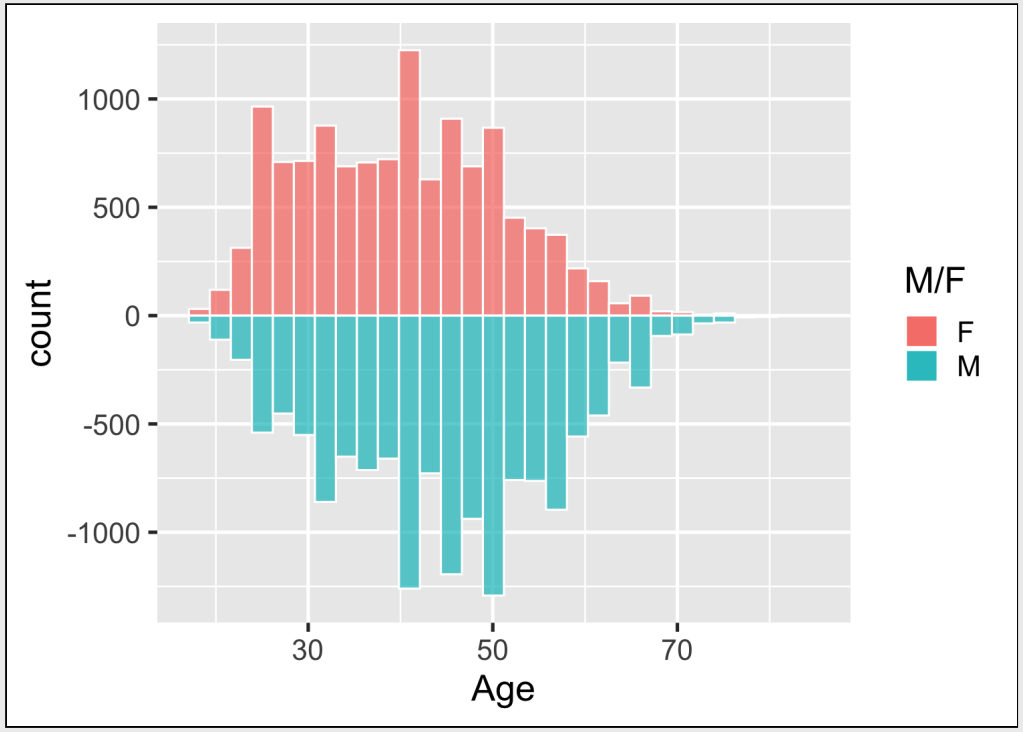
```
plot_grid(male, female,
          labels = c('Male', 'Female'))
```



How to make diverging histograms

Make the histograms by filtering the data

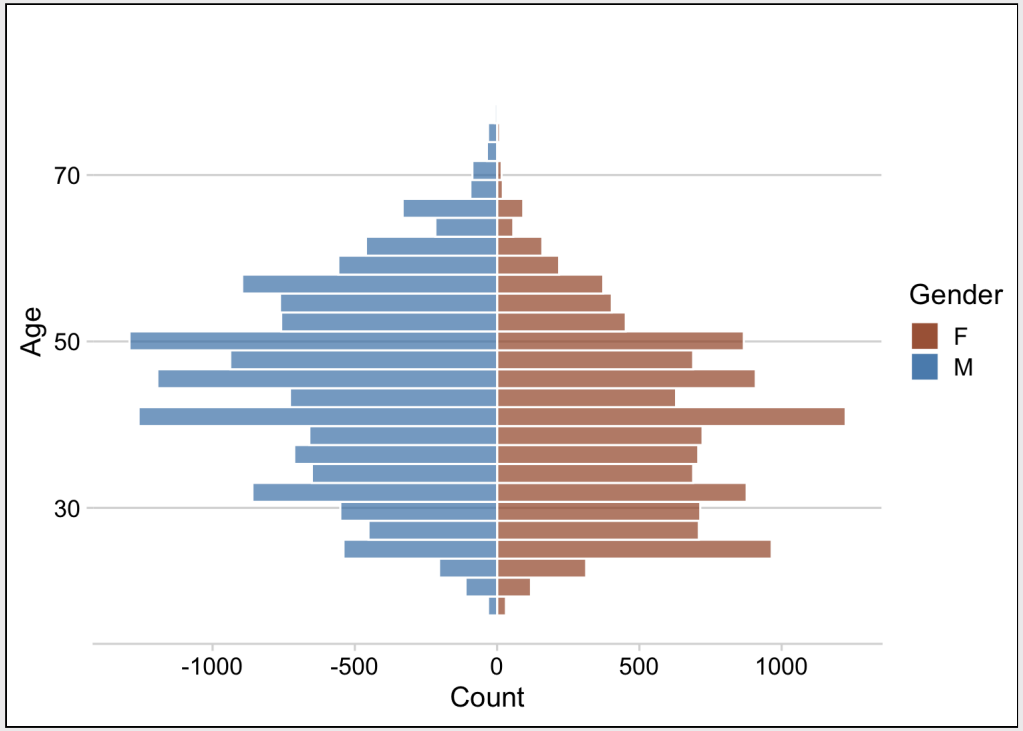
```
ggplot(marathon, aes(x = Age)) +
    # Add histogram for Female runners:
    geom_histogram(data = marathon %>%
                       filter(`M/F` == 'F'),
                   aes(fill = `M/F`, y=..count..),
                   alpha = 0.7, color = 'white') +
    # Add negative histogram for Male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

Rotate, adjust colors, theme, and 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')
```



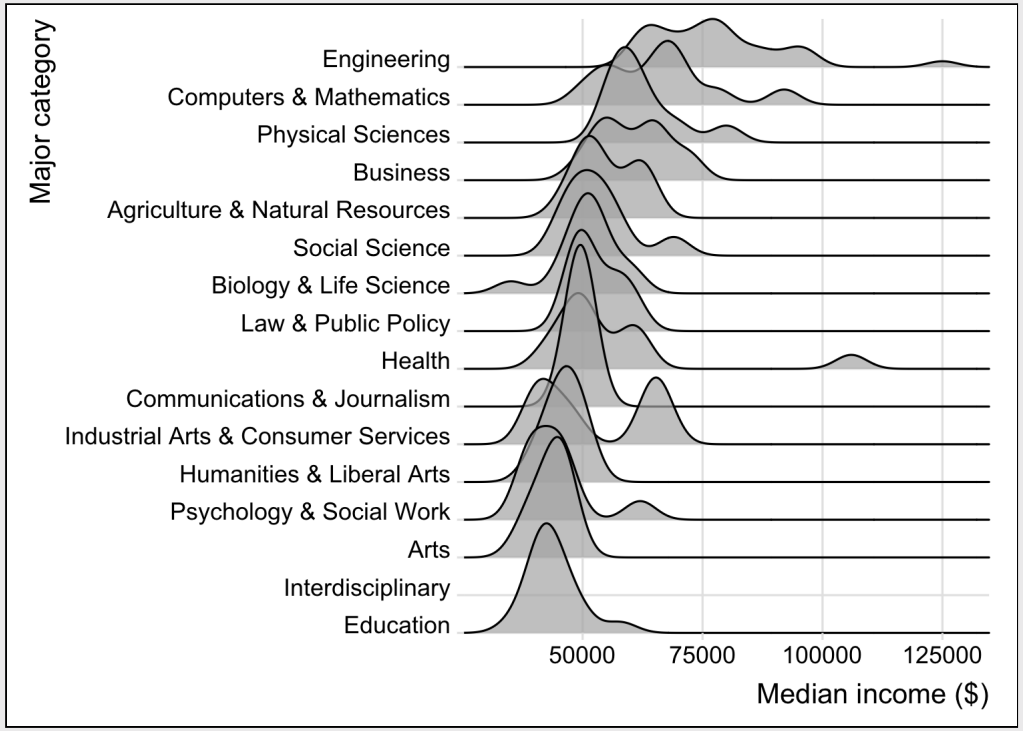
How to make ridgeplots

Reorder `major_category` variable

```
college_summary <- college_all_ages %>%
    mutate(
        major_category = fct_reorder(
            major_category, median))
```

Make the ridgeplot using the **ggridges** library

```
library(ggridges)

ggplot(college_summary) +
  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

Use the `gapminder.csv` data to create the following charts comparing the distribution of life expectancy across countries in continents in 2007.