McGill | School of Information Studies

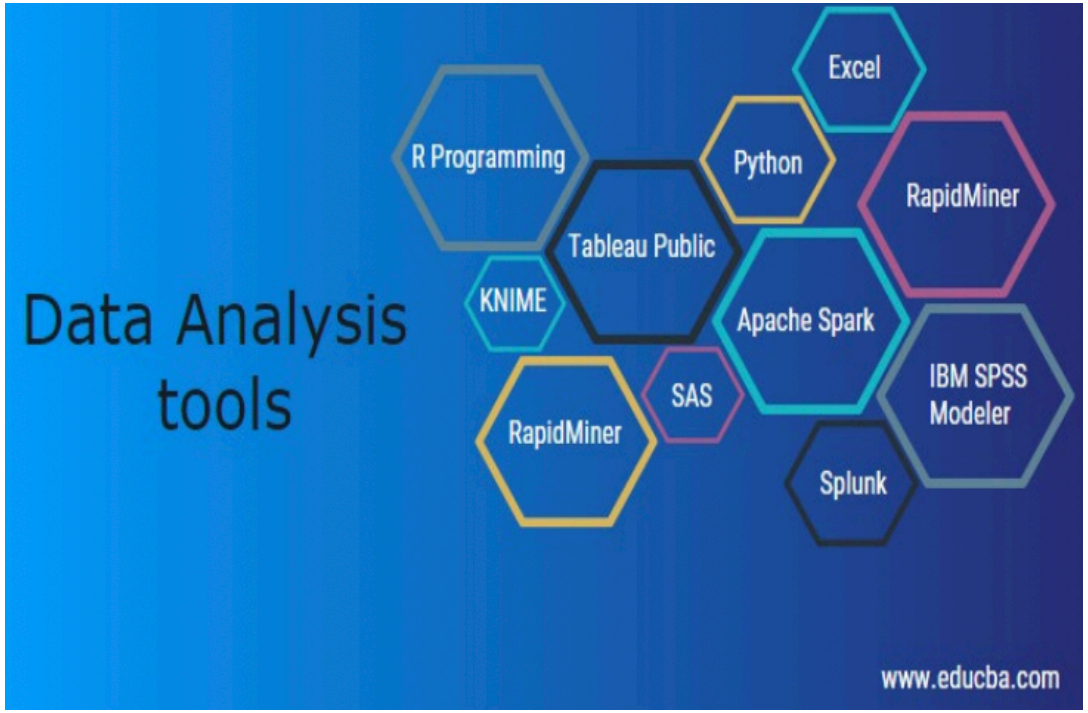
Data Visualization with R using ggplot2

WEEK 3

19 Sept 2022

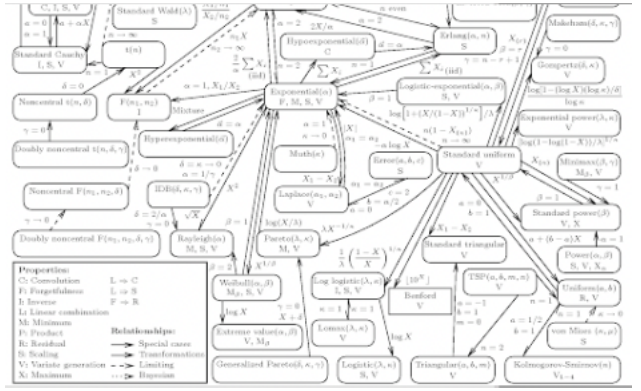
How do you usually make graphics?

1. Quick and accessible (excel).
2. Statistical software (e.g., JMP, SAS)
3. Tableau, PowerBI, Domo, Sisense
4. R, Python and/or others

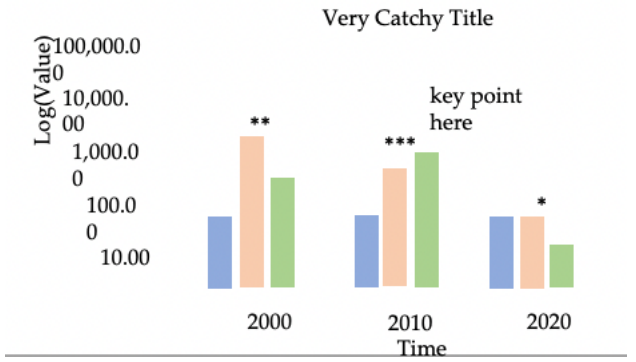


What makes for a good graphics?

- Really busy with every detail in text…
- Flashy plotting.
- A clear message that tells a story…



Useless charts

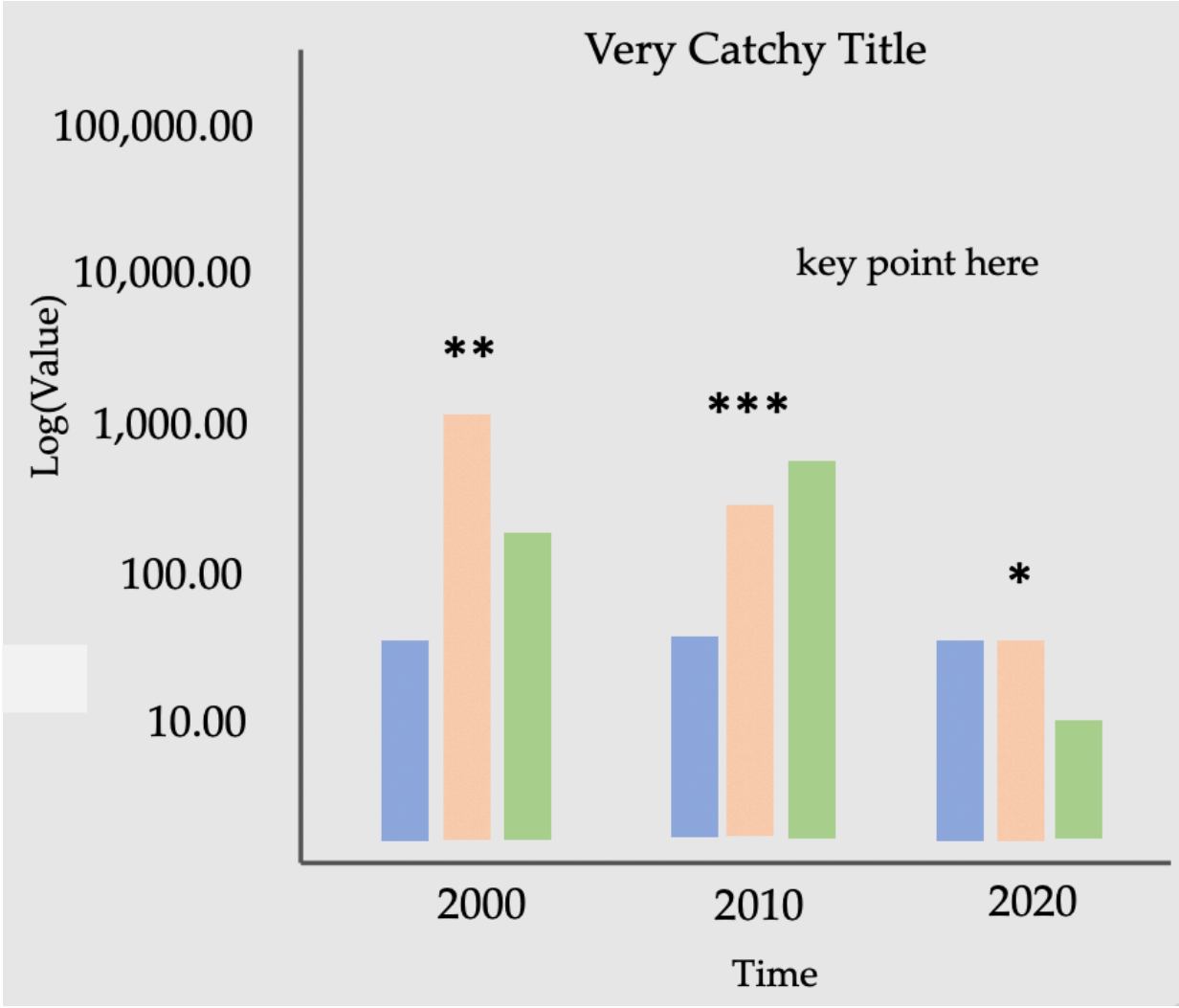


Foundations and Concepts

1. Data
   - Structure
     - Continuous
     - Discrete
   - Format
     - Wide
     - Long

2. Graph Attributes
   - Dimension
   - Axes Scale
   - Symbols, color schemes, etc.

3. Key elements
   - Title
   - Axes Labels
   - Embedded comments



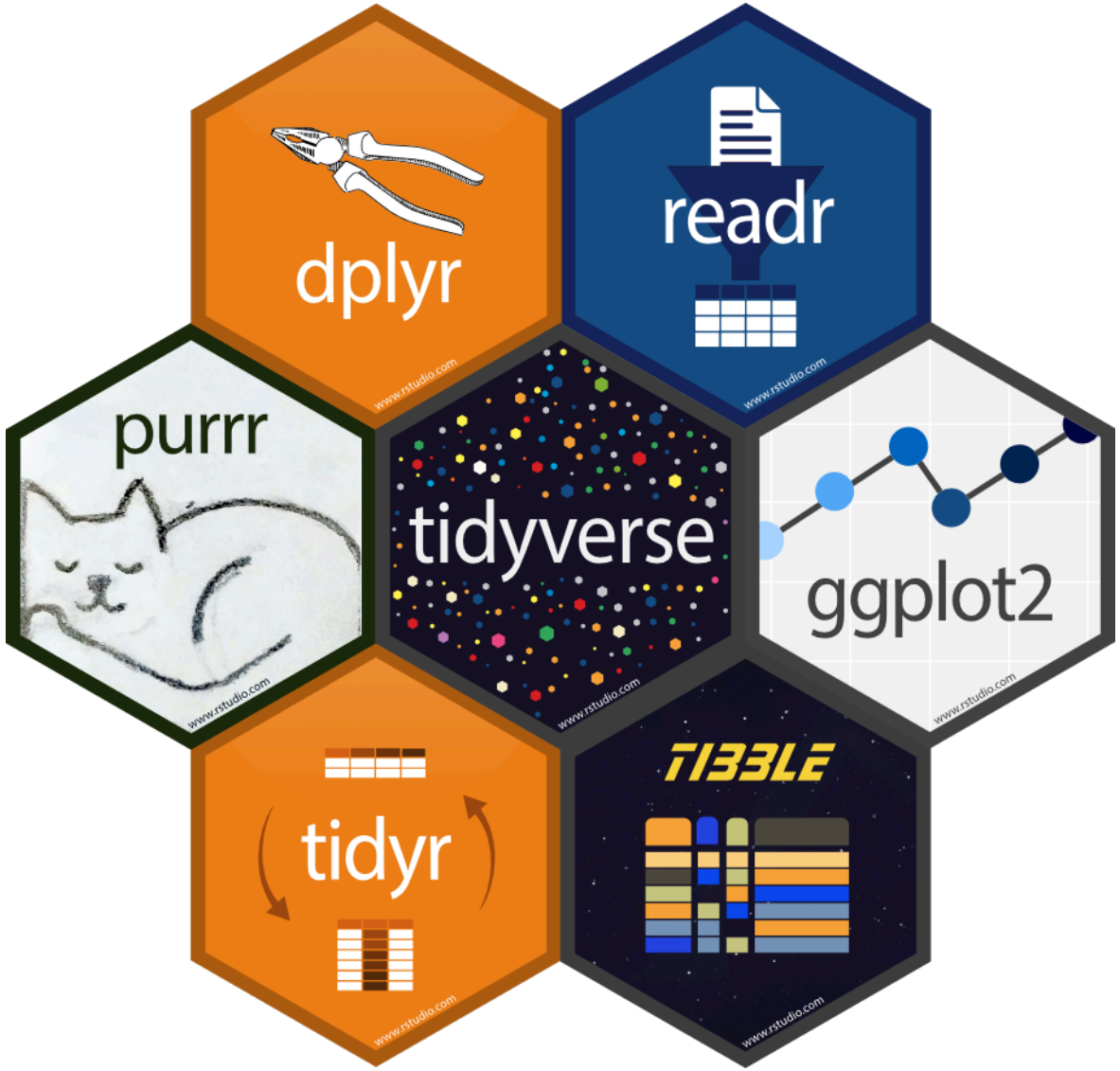
Tidyverse

- Originally 'Hadleyverse'

- Collection of R packages with shared:
  - Philosophy
  - Structure
  - Syntax

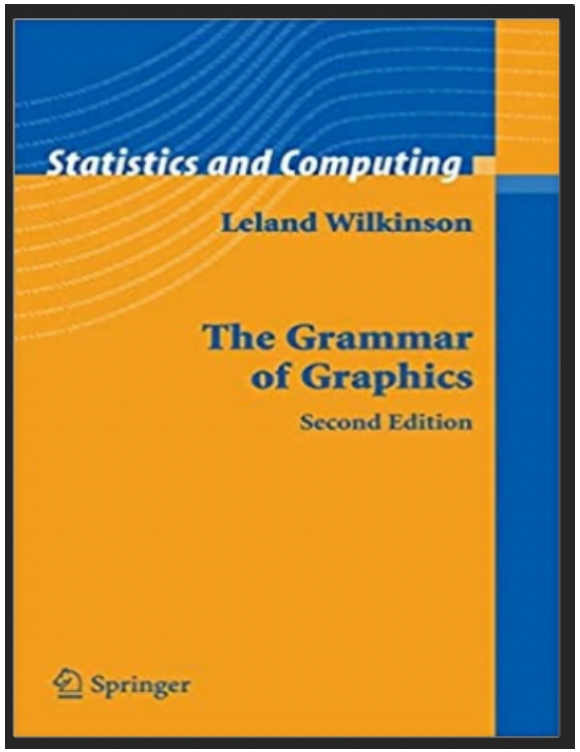
- Package 'piping' (%>%)
  - Functions act as verbs

- Over 27 packages including:
  - Readr – importing data
  - Dplyr – data cleaning
  - Ggplot2 – data visualization
  - Lubridate – working with dates
  - Stringr – working with strings



Ggplot2

- Developed by Hadley Wickham
- 10+ years old
- Readability > base R plot
- Over 84 citations
  - ggtext
  - Ggthemes
  - Gganimate
  - esquisse
- 'Grammer of Graphics'
- 8 main class of functions
  - Align with key elements and attributes of graphic communication



Structure

1. Data
   - Structure
     - Continuous
     - Discrete
   - Format
     - Wide
     - Long
2. Graph Attributes
   - Dimension
   - Axes Scale
   - Symbols, color schemes, etc.
3. Key elements
   - Title
   - Axes Labels
   - Embedded comments

ggplot2:: Syntax

1. Data
2. Function
3. Coordinates
4. Mapping
5. Geometrics
6. Scales
7. Facets
8. Themes

Structure

```
df %>%
```

Structure

```
df %>%
```

- Considerations
  - Discrete or Continuous data for coordinates
  - Discrete or Continuous metadata for mapping
  - Missing Data etc
  - Structure check
    - skimr::skim()
  - Tidyverse can be really useful here
    - Data transformations
    - Data reshaping
- Arguments
  - Data can be piped to function or called within
    - skimr::skim(df)
    - df %>% dplyr::filter() %>% dplyr::pivot()

Structure

```
df %>% +
ggplot(...) +
```

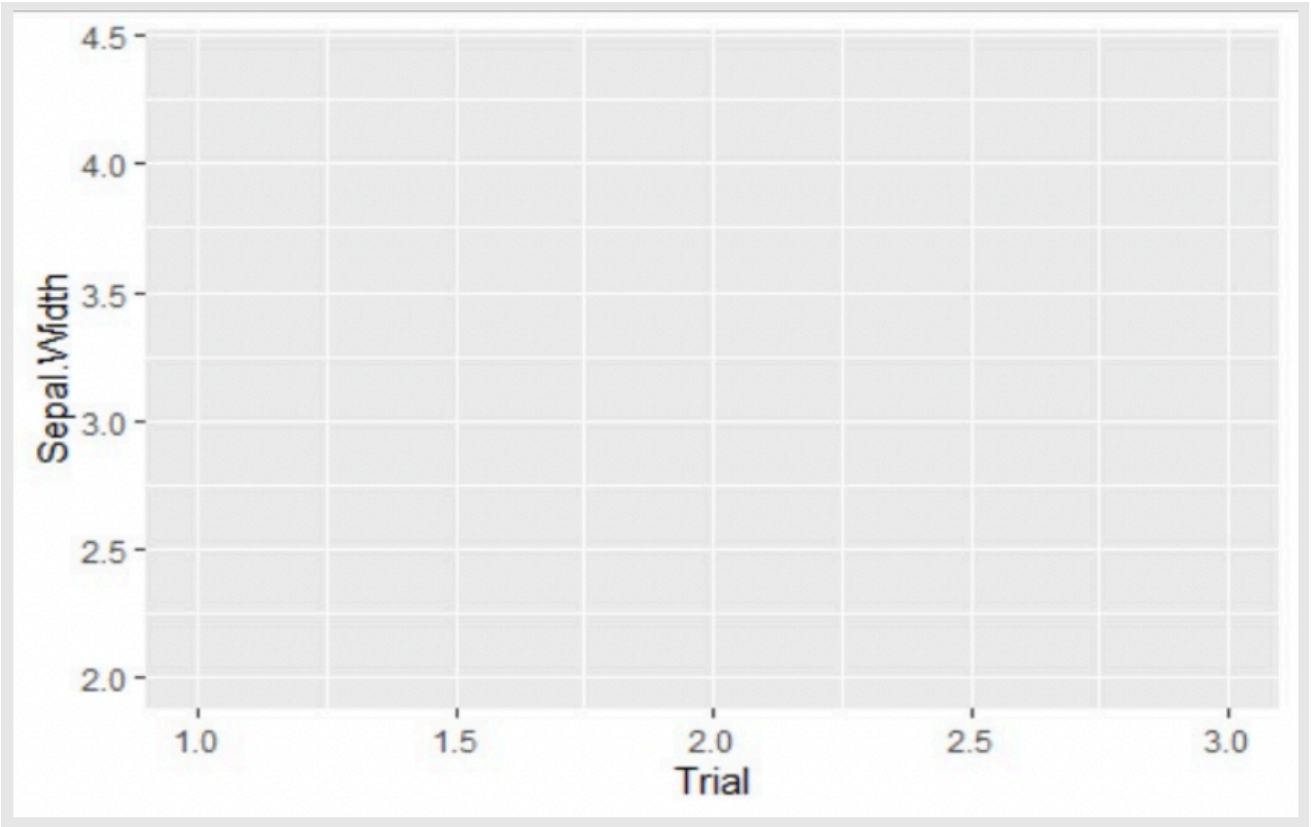
Structure

ggplot(…)

- Considerations
  - Calls the plot object
  - Best placement of df for plot development
- Arguments
  - Data
  - Coordinates
  - Mapping
    - ggplot(df, aes()) + …
    - ggplot(df) +
    - ggplot() +
    - df %>% ggplot(., aes()) +

Structure

```
df %>% +
ggplot(., eas (x = x, y = y, ...)) +
```



Structure

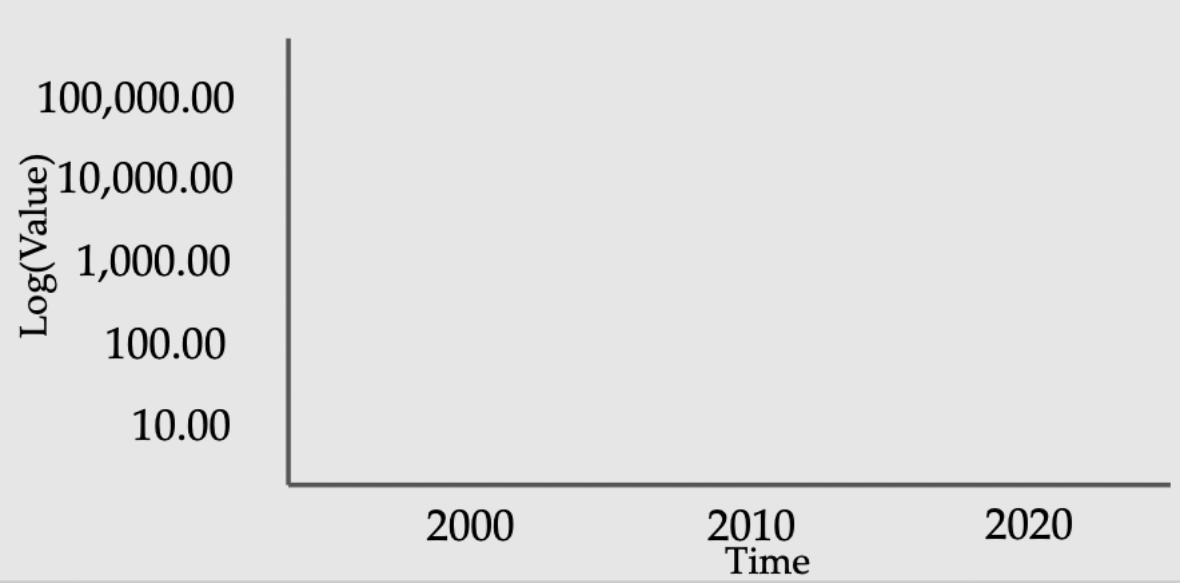
```
ggplot(., eas (x = x, y = y, …)) +
```

- Considerations
  - Data positions for plot
  - Not necessary to specify here, but must be supplied in plot
  - Continuous Data
    - Coordinate according to the data
  - Discrete Data
    - At 1,2,3 etc. on axis
    - Alphabetical for character class
    - Level for factor class

Structure

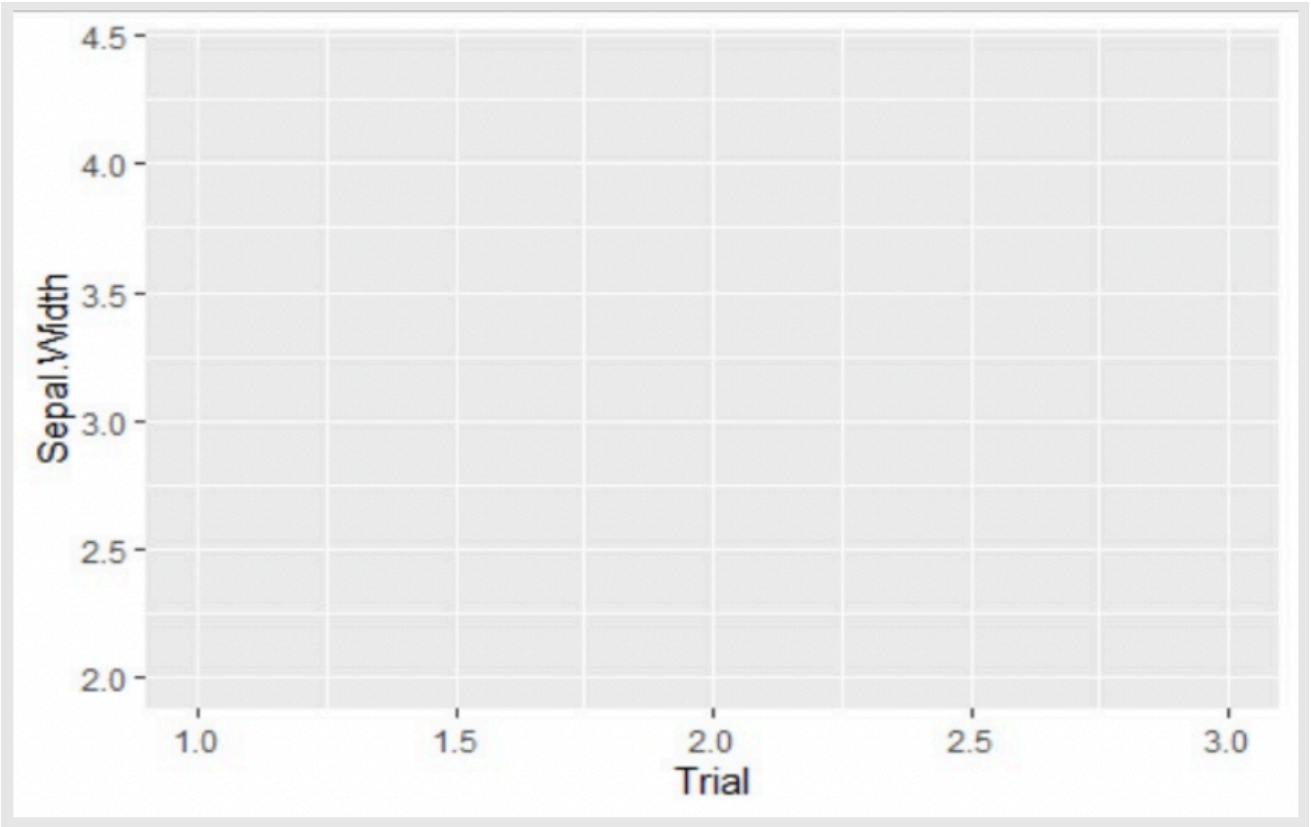
```
ggplot(., eas (x = x, y = y, …)) +  +
```

- Arguments
  - x =
  - y =
  - Positional, not necessary to specify
  - For geometries that use count, y is not accepted



Structure

```
df %>% +
ggplot(., eas (x = x, y = y, *=var1)) +
```



Structure

```
ggplot(., eas (x = x, y = y, *=var1)) +
```

- Considerations
  - Variables are mapped to visual properties (aesthetics)
  - Choosing the aesthetic (*)
    - Color
    - Fill
    - Size
    - Shape
    - Linetype
    - Transparency
  - Is it Continuous or Discrete?
  - What are you mapping to?

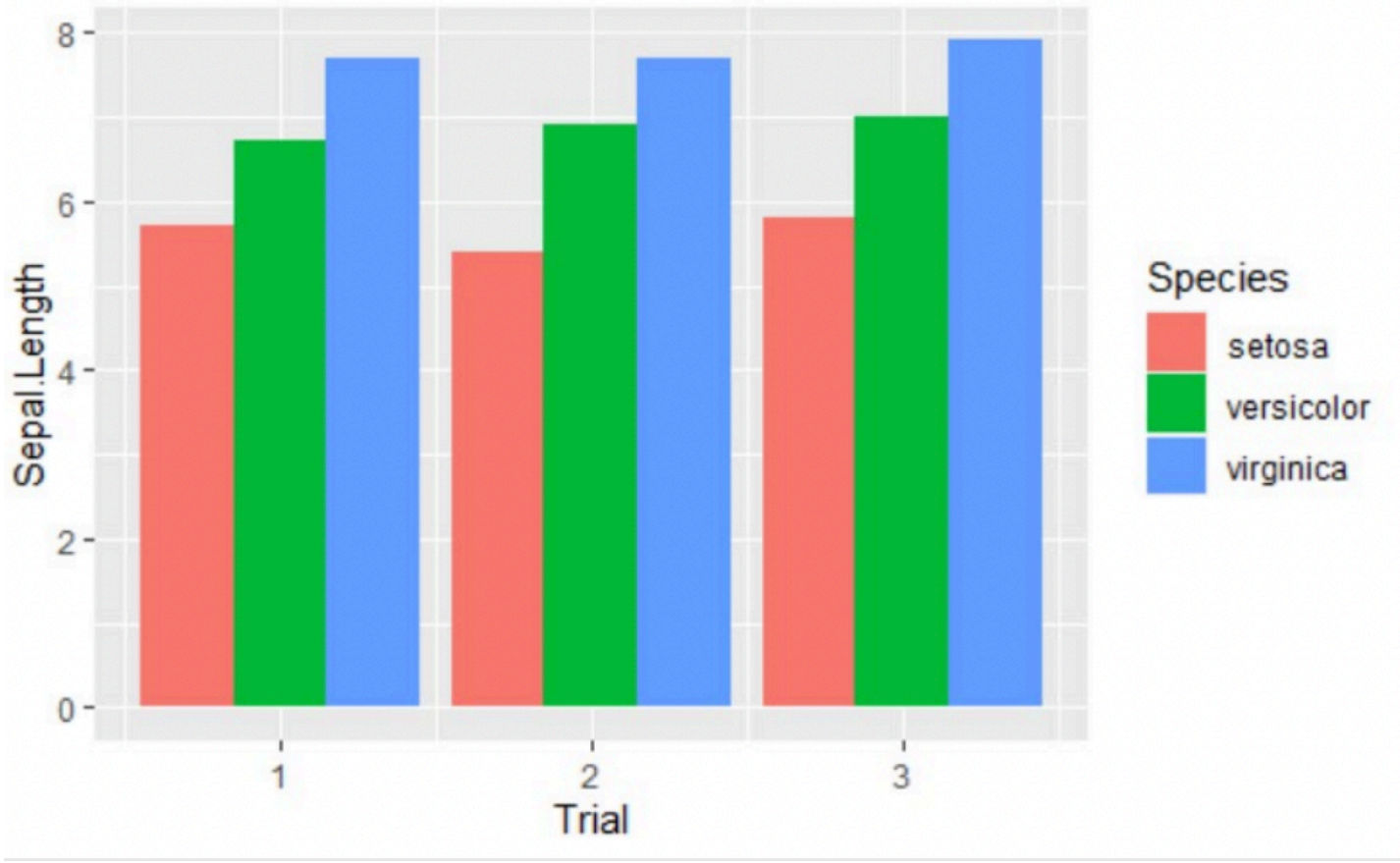
Structure

```
ggplot(., eas (x = x, y = y, *=var1)) +
```

- Arguments
  - Mappings can be set in ggplot()
    - ggplot(., aes(color=var1))
  - Mappings can be set in individual layers
    - geom_point (., aes(color=var1))
  - What is outside of the mapping will be interpreted literally
    - geom_point (., aes(color='var1'))
  - Levels of the mapping are set in scales, else:
    - (., aes(color='black'))

Structure

```
df %>% +
ggplot(., eas (x = x, y = y, ...)) +
geom_*(...) +
```



Structure

```
geom_*(...) +
```

- Considerations
  - What kind of visual representations (*) are the best approach?
  - Types of Geometry
    - Reference Lines
    - Bar Charts
    - Dots and points
    - Boxplots
    - Heatmaps
    - Density
    - Polygons
    - Jitters
    - Error Bars
    - Text and Labels

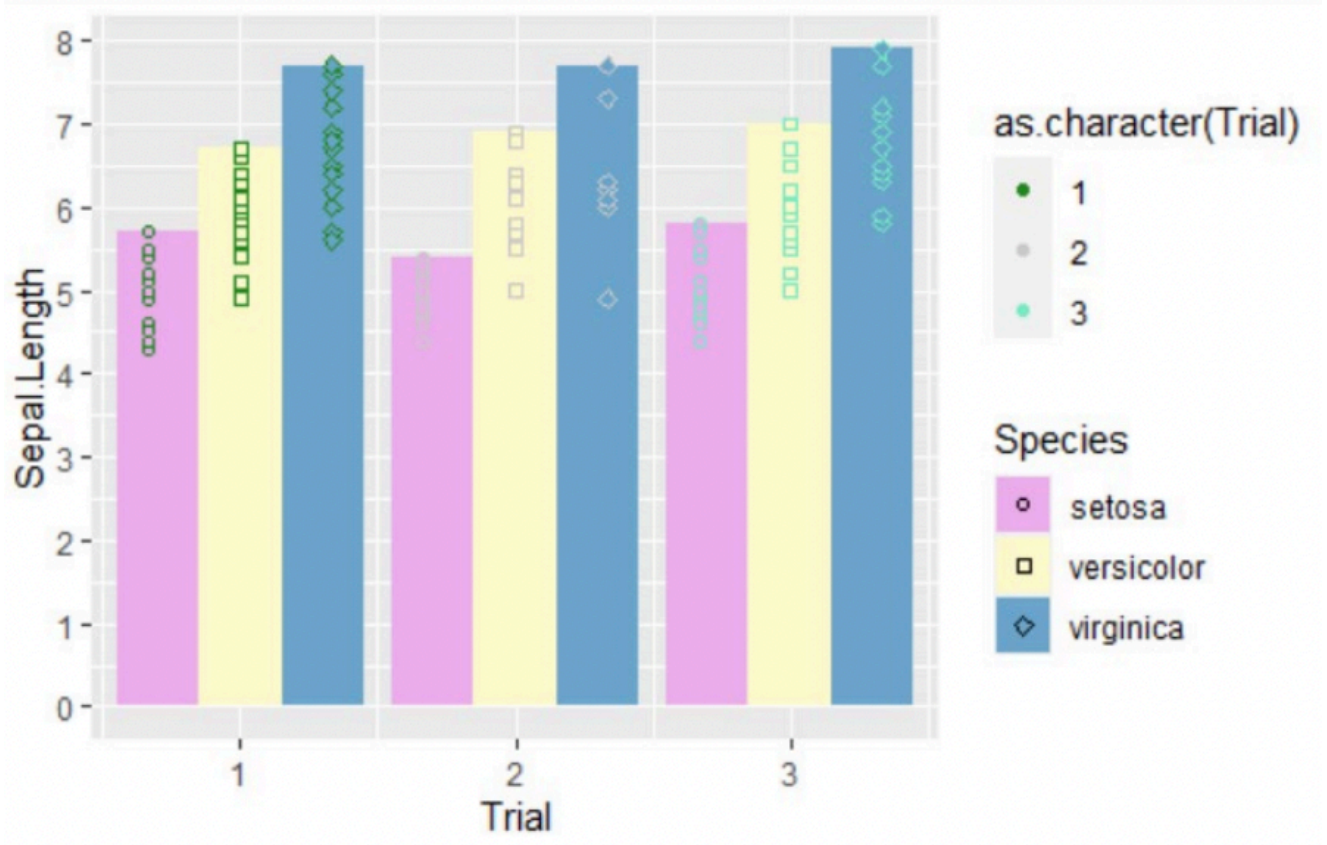
Structure

```
geom_*(...) +
```

- Arguments
  - Over 40 individual geoms
  - Can build individual mappings within geoms_*()
  - Can add multiple geometries as annotation layers
  - Know the defaults
    - geom_*(mapping = , data = , stat = , position =)
  - geom_*()
    - geom_point()
    - geom_hist()
    - geom_bar()
    - geom_col()
    - geom_tile()

Structure

```
df %>% +
ggplot(., eas (x = x, y = y, ...)) +
geom_*(...) +
scale_*_*(...) +
```



Structure

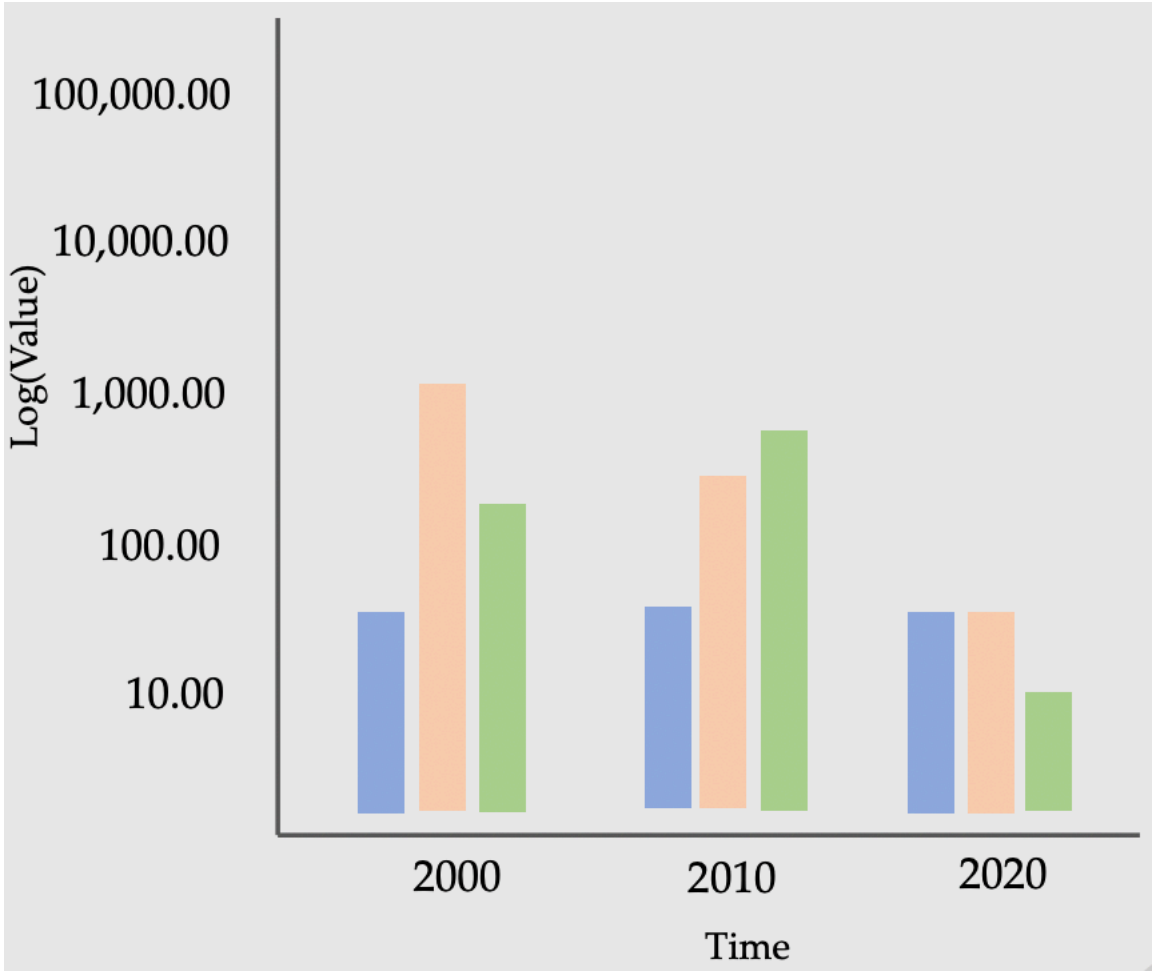
scale_*_*(...) +

- Concept
  - Specify Data and Mappings
  - Set arguments for coordinates (*)
    - X
    - y
  - Set arguments for mappings (*)
    - Color
    - Fill
    - Alpha
    - Linetype
    - ,… and many more
  - Different calls for discrete and continuous data types
  - Commonly used defaults are prebuilt

Structure

scale_*_*(…) +

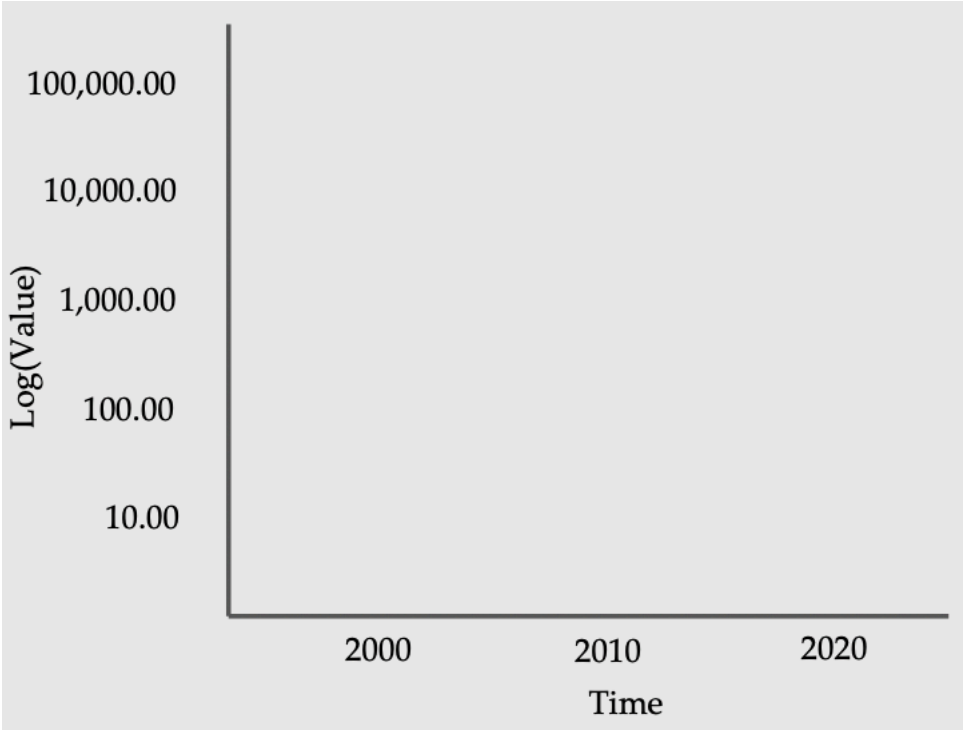
- Arguments
  - Scale_*_*(
    name = ,
    breaks = ,
    values = ,
    labels = ,
    limits = ,
    trans = ,
    guide = ,
    position = ,
    …)



Structure

scale_*_*(...) +

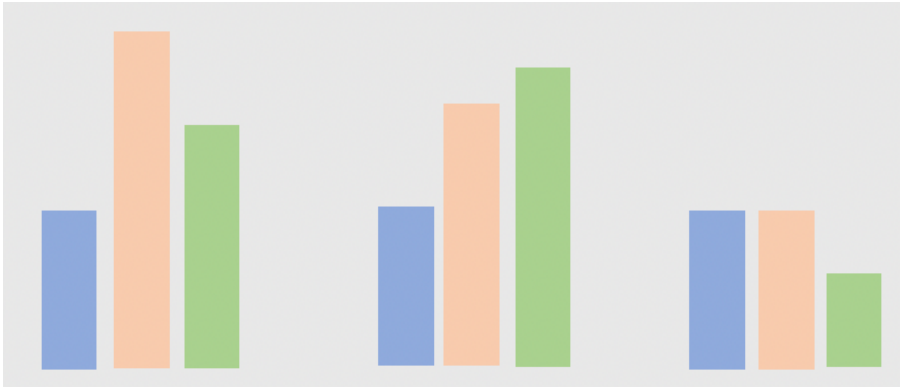
- Standard Axis Scales
  - scale_x_continuous(...)
  - scale_y_continuous(...)
  - scale_x_discrete(...)
  - scale_y_discrete(...)
- Pre-built Custom Axis Scales
  - scale_*_log10(...)
  - scale_*_reverse(...)
  - scale_*_sqrt(...)
  - scale_*_datetime(...)
  - … and many more



Structure

scale_*_*(...) +

- Standard Alpha Scales
  - scale_alpha_continuous(...)
  - scale_alpha_discrete(...)
- Standard Linetype Scales
  - scale_linetype_continuous(...)
  - scale_linetype_discrete(...)
- Standard Shape Scales
  - scale_shape_continuous(...)
  - scale_shape_discrete(...)

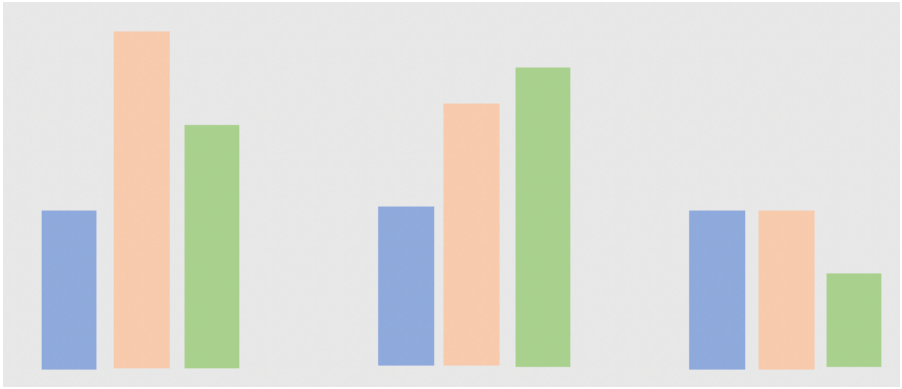


Structure

```
scale_*_*(...) +
```

- Standard Color Scales
  - scale_color_continuous(…)
  - scale_fill_continuous(…)
  - scale_color_manual(…)
  - scale_fill_manual(…)

- Pre-Built Custom Color Scales
  - scale_*_brewer(...)
  - scale_*_gradient(...)
  - scale_*_gradientn(...)
  - scale_*_viridis(…)
  - … and many more

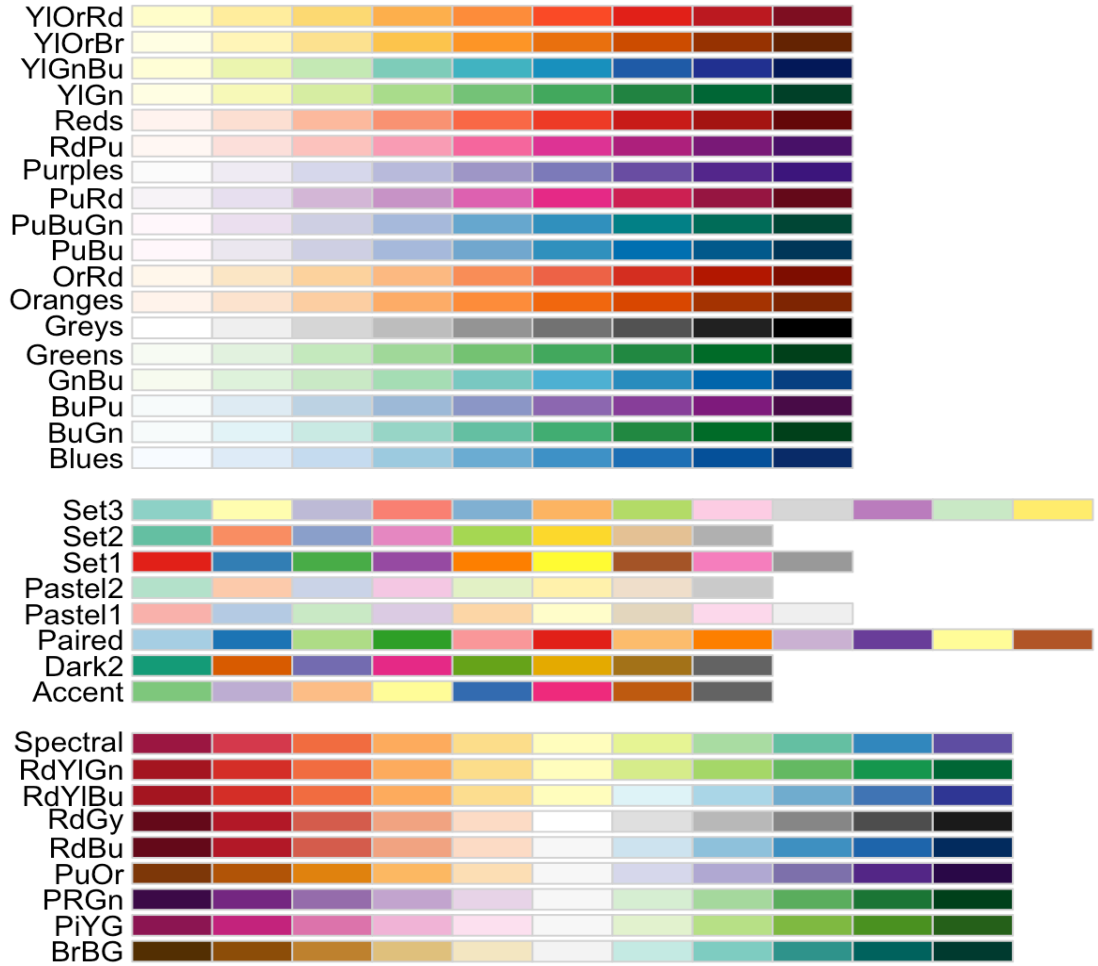


Structure

scale_*_*(...) +

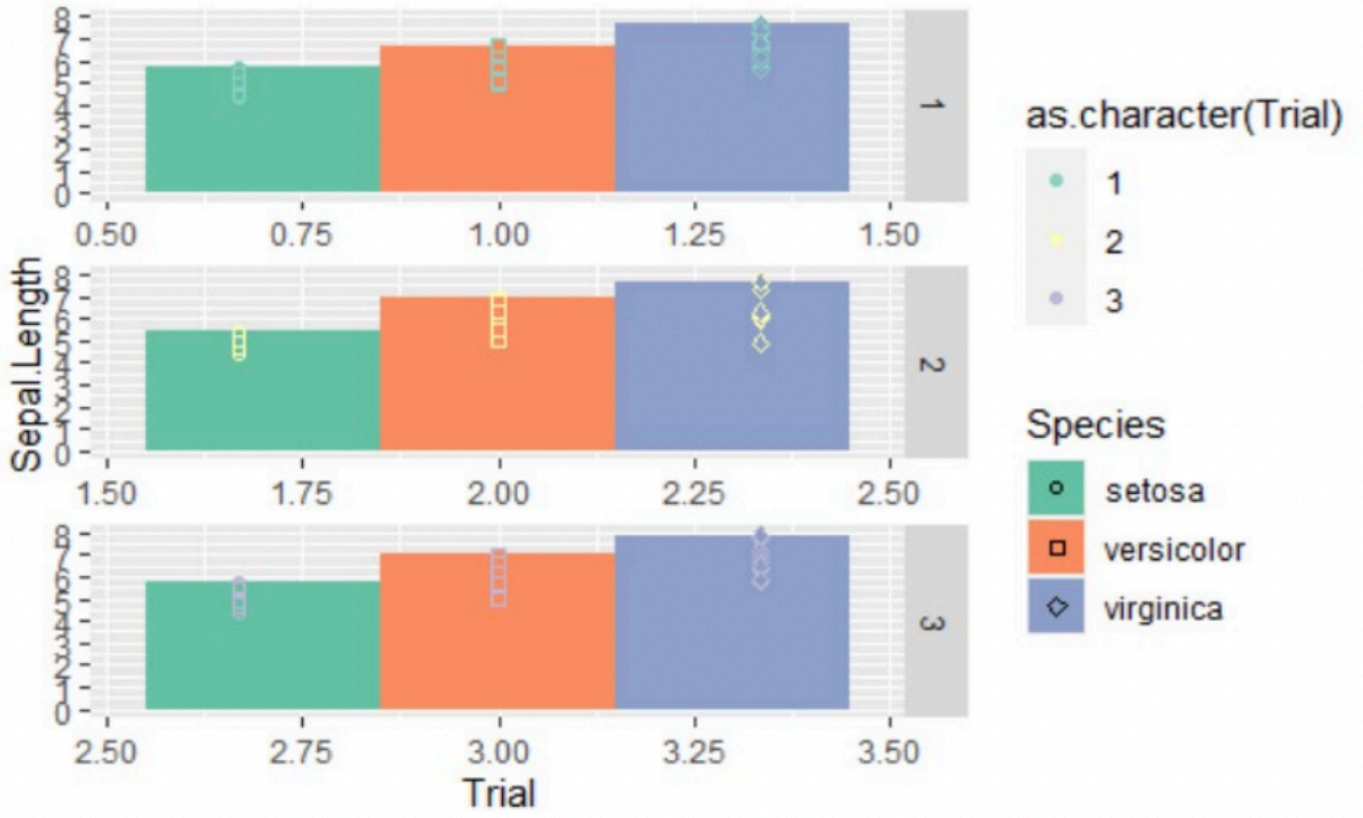
- Standard Scale – specify colors
  - scale_color_manual(
    values = c('red', 'green', 'blue')

- Pre-Built Custom Color Scales
  - scale_*_brewer
    (type =,
    palette = ,
    direction = ,
    aesthetics =)
  - scale_color_brewer(palete = 'set2')



Structure

```
df %>% +
ggplot(., eas (x = x, y = y, ...)) +
geom_*(...) +
scale_*_*(...) +
facet_*(...) +
```



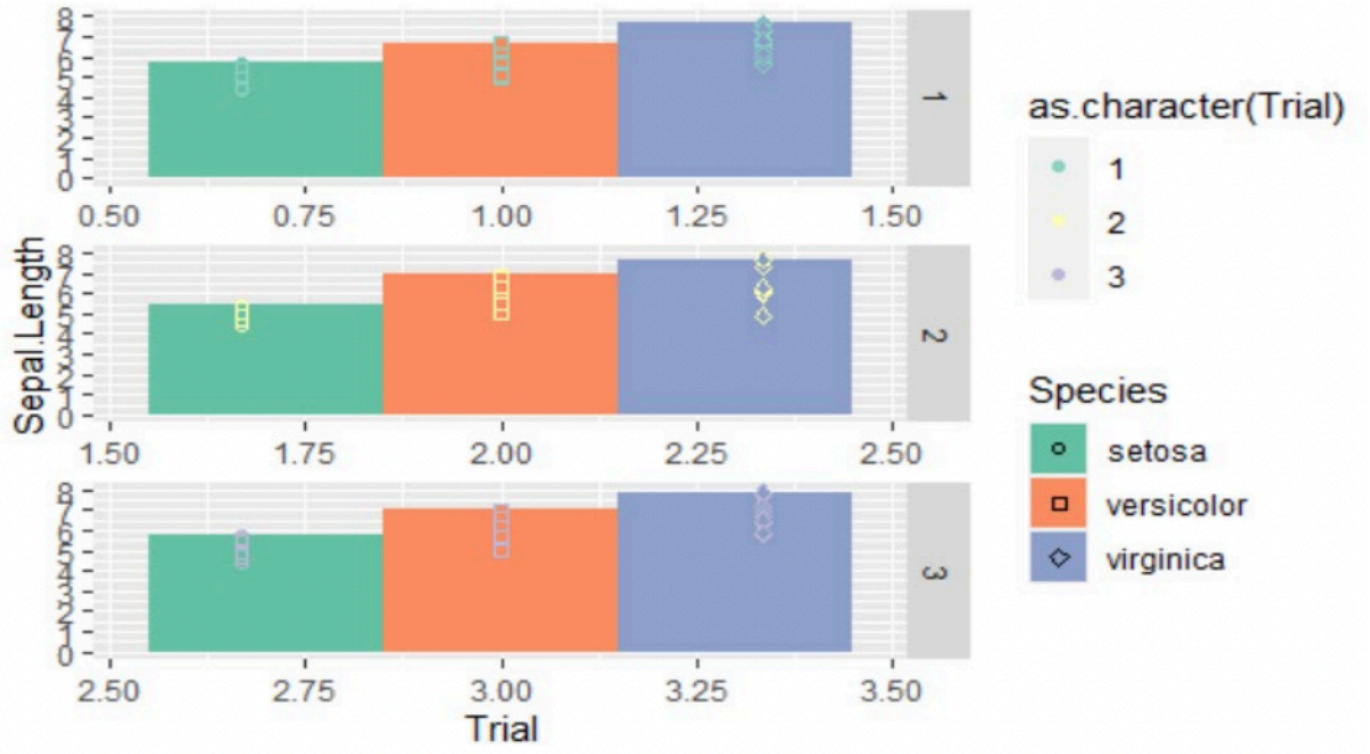
Structure

```
facet_*(...) +
```

- Concept
  - Highlight levels in data through multiple panels
  - facet_wrap(…)
    - creates a ribbon of levels
  - facet_grid(…)
    - creates a matrix of rows and columns of variable combinations
- Arguments
  - nrow =,
  - ncol =,
  - scales = ,
  - shrink =,
  - labeller = ,
  - strip.position = ,
  - …)

Structure

```
df %>% +
ggplot(., eas (x = x, y = y, ...)) +
geom_*(...) +
scale_*_*(...) +
facet_*(...) +
theme(...)
```



Structure

theme(…)

- Concept
  - Encompasses ALL the options of ggplot2:: plot ‘Elements’
  - 4 main modifiers
    - line: all line elements
    - rect: all rectangular elements
    - text: all text elements
    - title: all title elements (including: plot, axes, legends..)
  - Pre-built themes available
    - theme_bw(…)
    - theme_grey(…)
  - ggthemes::
    - a package with a wider selection of Pre-Built themes

Structure

theme(…)

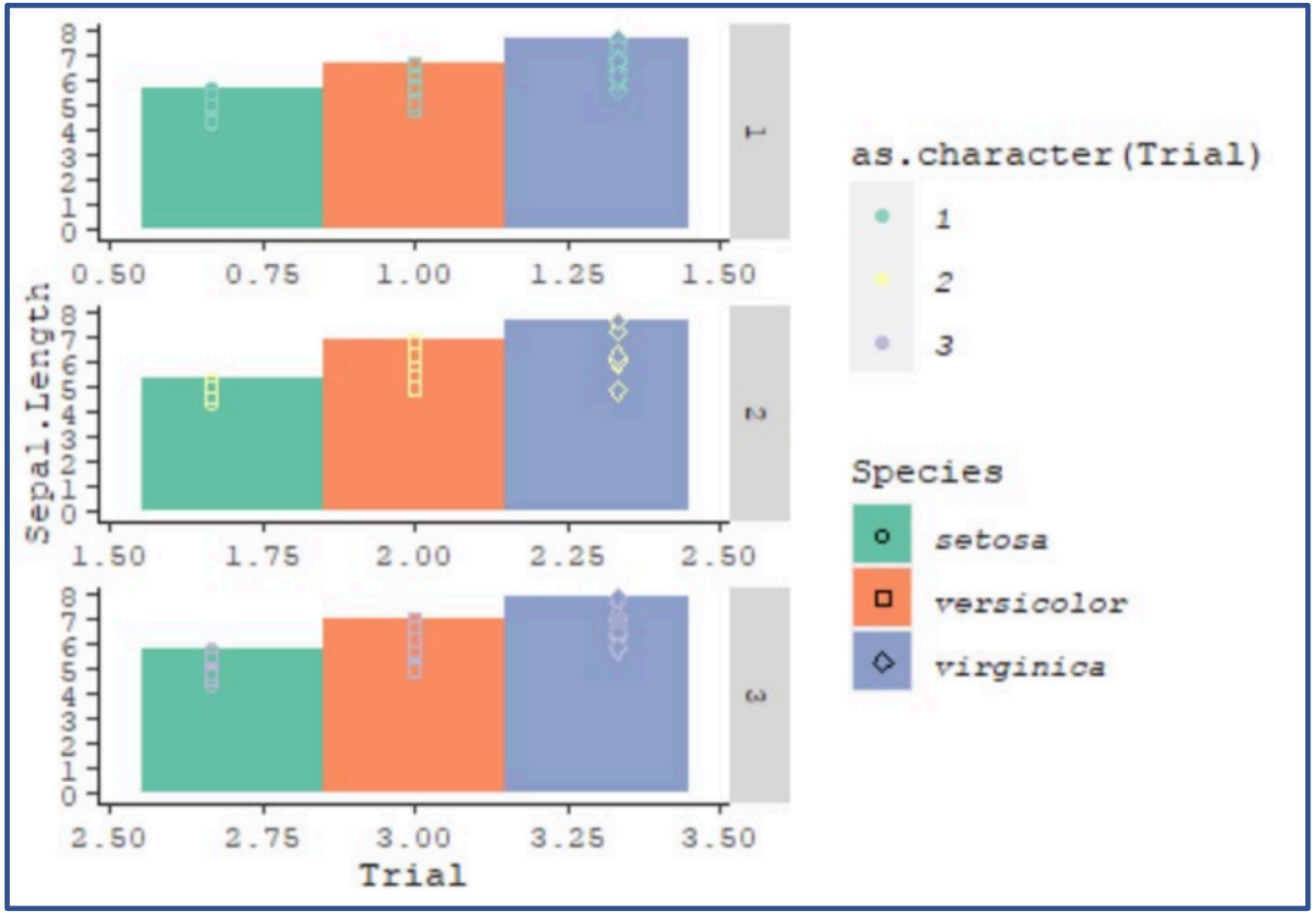
- Argument
  - The main modifiers:
    - theme(
    - text* = element_text(),
    - panel* = element_rect(),
    - axis* = element_line(),
    - title* = element_text())
  - where * = …

Structure

theme(…)

```
theme(
aspect.ratio,
  axis.title,
  axis.title.x,
  axis.title.x.top,
  axis.title.x.bottom,
  axis.title.y,
  axis.title.y.left,
  axis.title.y.right,
  axis.text,
  axis.text.x,
  axis.text.x.top,
  axis.text.x.bottom,
  axis.text.y,
  axis.text.y.left,
  axis.text.y.right,
  axis.ticks,
  axis.ticks.x,
  axis.ticks.x.top,
  axis.ticks.x.bottom,
  axis.ticks.y,
  axis.ticks.y.left,
  axis.ticks.y.right,
  axis.ticks.length,
  axis.ticks.length.x,
  axis.ticks.length.x.top,
  axis.ticks.length.x.bottom,
  axis.ticks.length.y,
  axis.ticks.length.y.left,
  axis.ticks.length.y.right,
  axis.line,
  axis.line.x,
  axis.line.x.top,
  axis.line.x.bottom,
  axis.line.y,
  axis.line.y.left,
  axis.line.y.right,
  legend.background,
  legend.margin,
  legend.spacing,
  legend.spacing.x,
  legend.spacing.y,
  legend.key,
  legend.key.size,
  legend.key.height,
  legend.key.width,
  legend.text,
  legend.text.align,
  legend.title,
  legend.title.align,
  legend.position,
  legend.direction,
  legend.justification,
  legend.box,
  legend.box.just,
  legend.box.margin,
  legend.box.background,
  legend.box.spacing,
  panel.background,
  panel.border,
  panel.spacing,
  panel.spacing.x,
  panel.spacing.y,
  panel.grid,
  panel.grid.major,
  panel.grid.minor,
  panel.grid.major.x,
  panel.grid.major.y,
  panel.grid.minor.x,
  panel.grid.minor.y,
  panel.ontop,
  plot.background,
  plot.title,
  plot.title.position,
  plot.subtitle,
  plot.caption,
  plot.caption.position,
  plot.tag,
  plot.tag.position,
  plot.margin,
  strip.background,
  strip.background.x,
  axis.ticks.length.x.bottom,
  strip.background.x,
strip.background.y,
  strip.placement,
  strip.text,
  strip.text.x,
  strip.text.y,
  strip.switch.pad.grid,
  strip.switch.pad.wrap,
  ...,)
```

Application

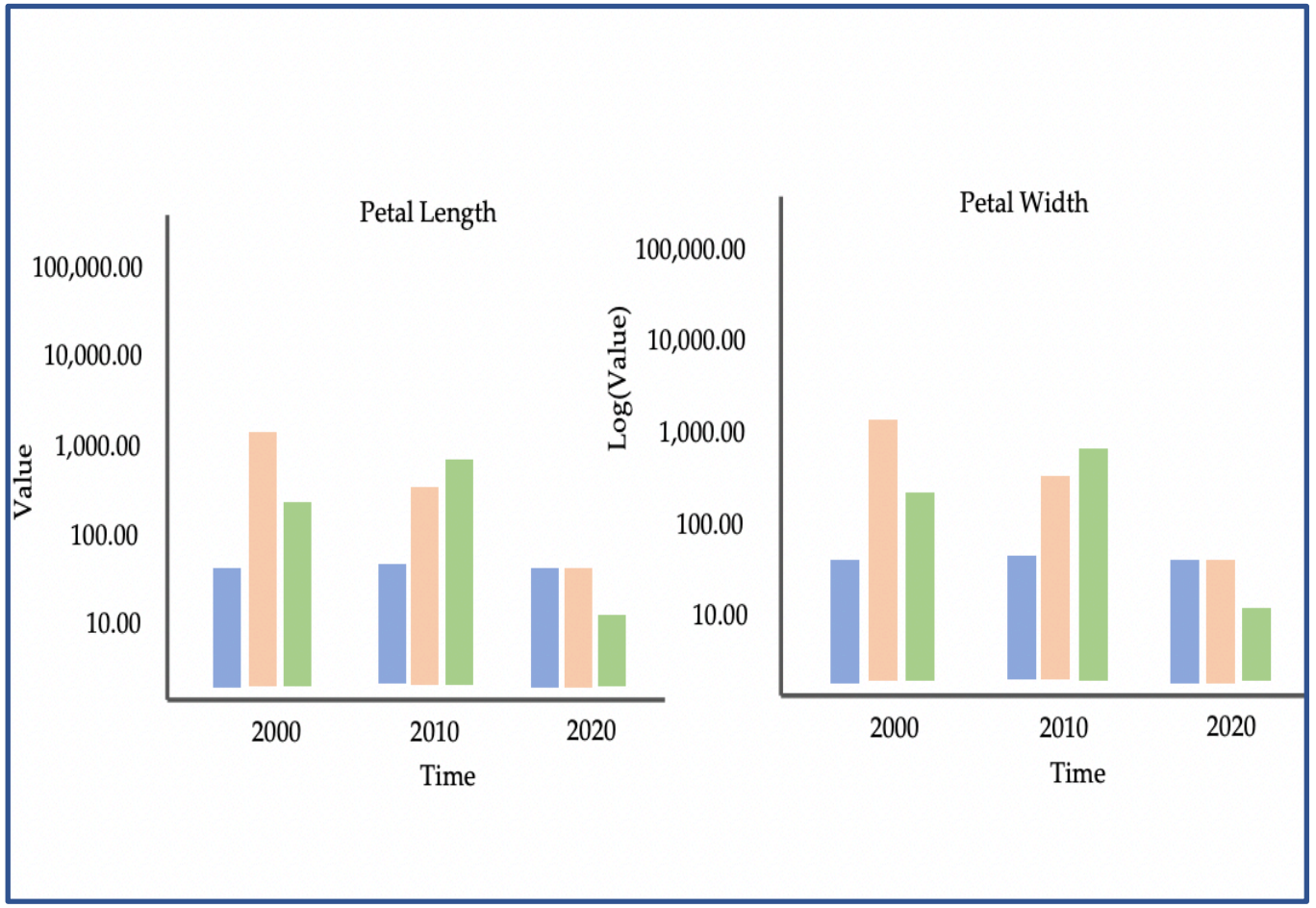


- Facetted Width and Length
- Bar Graph
- Trial axis breaks as '2010', 2015', '2020'
- Y axis at 2, 4, 6, 8
- Choose a different palette for species
- Ditch the points, and accompanying scale
- Dodge position
- Species in italics
- No grey in facet
- New font

SKETCH → STORY

Application

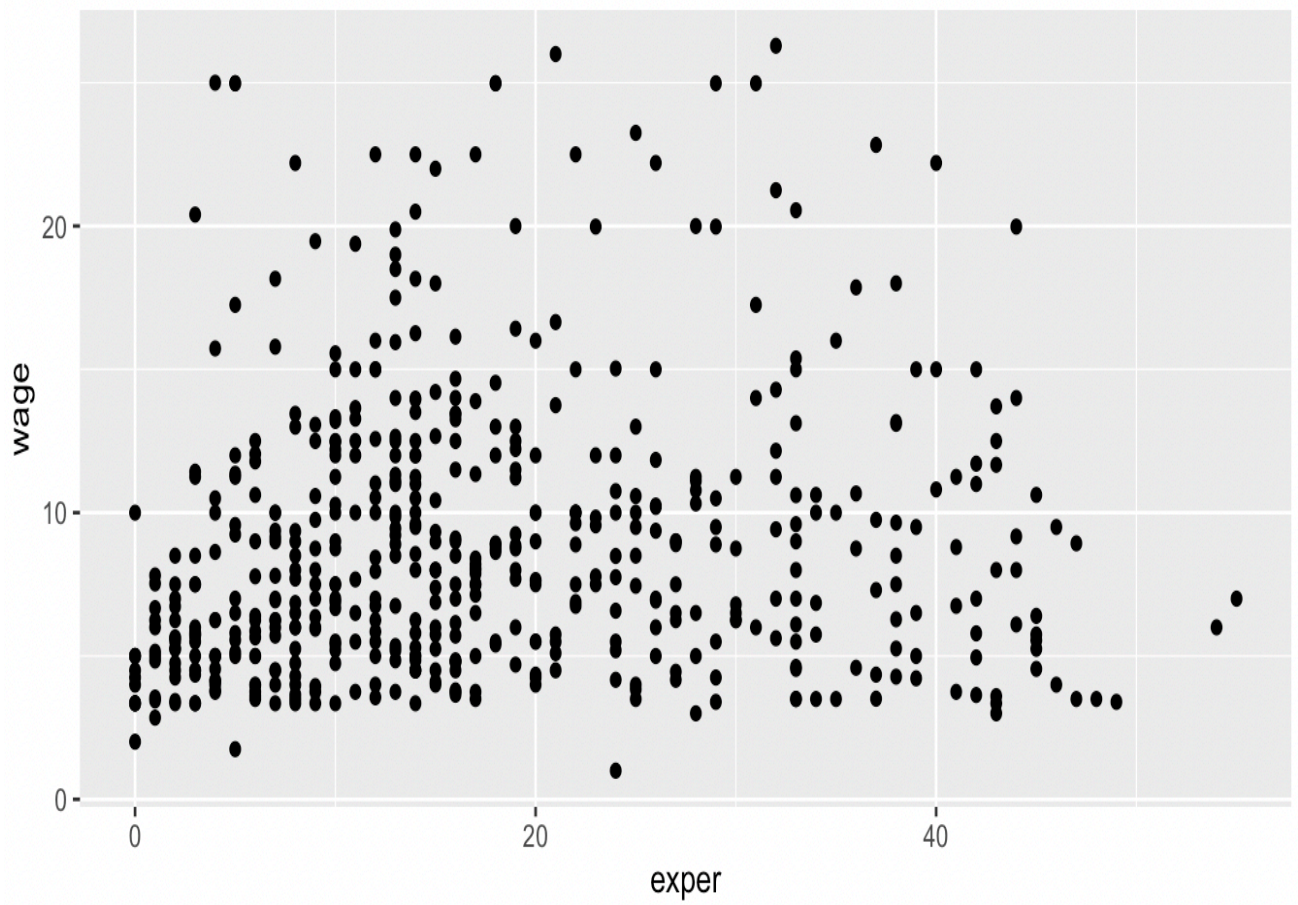
- Facetted Width and Length
- Bar Graph
- Trial axis breaks as ‘2010’, 2015’, ‘2020’
- Y axis at 2, 4, 6, 8
- Choose a different palette for species
- Ditch the points, and accompanying scale
- Dodge position
- Species in italics
- No grey in facet
- New font



SKETCH → STORY

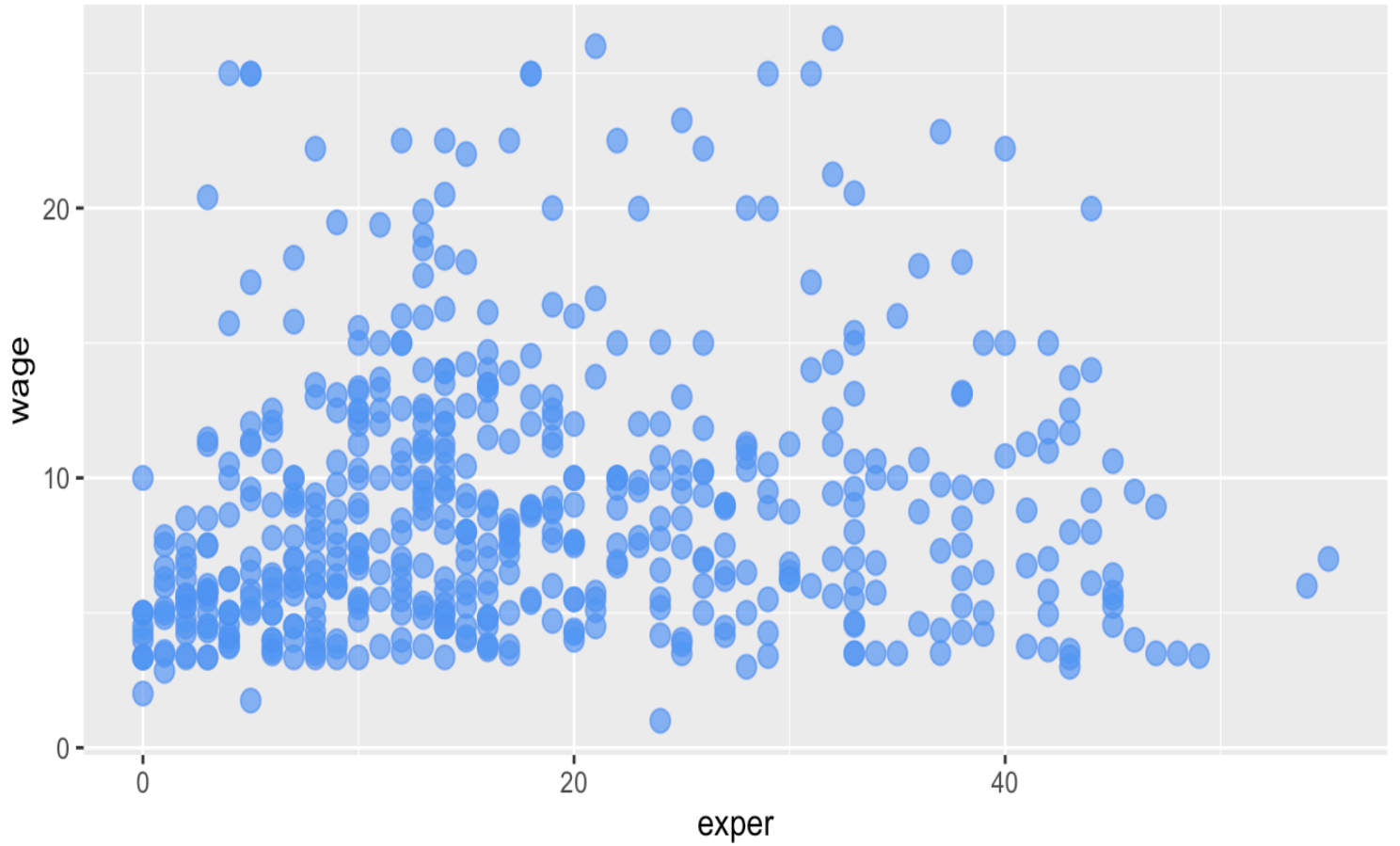
Examples

```r
# load data
library(mosaicData)
data(CPS85 , package = "mosaicData")

# delete outlier
library(dplyr)

plotdata <- filter(CPS85, wage < 40)

# redraw scatterplot
library(ggplot2)
ggplot(data = plotdata, mapping = aes(x = exper, y = wage)) + geom_point()
```



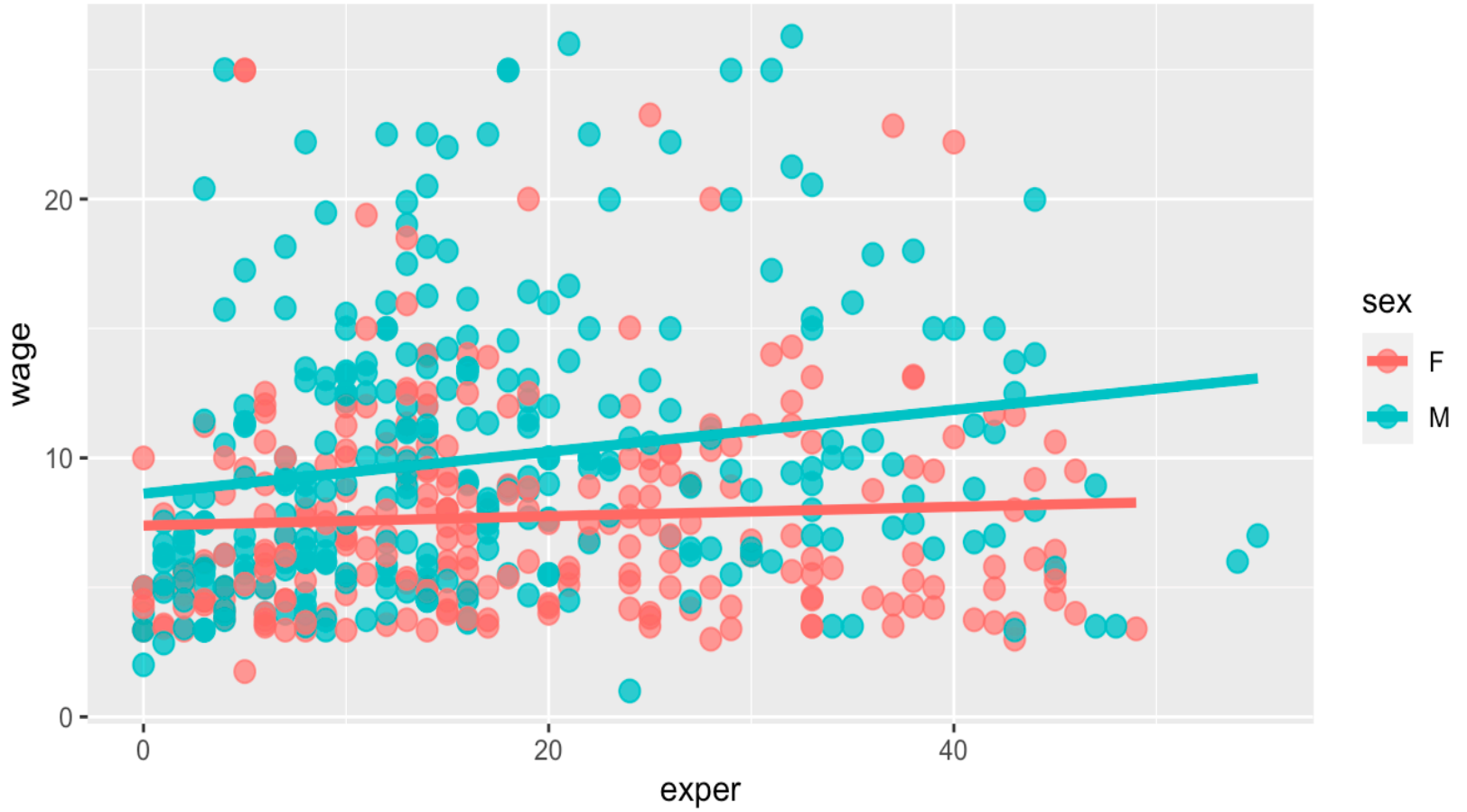
Examples

```
# make points blue, larger, and semi-transparent
ggplot(data = plotdata,
       mapping = aes(x = exper, y = wage)) +
  geom_point(color = "cornflowerblue",
             alpha = .7,
             size = 3)
```



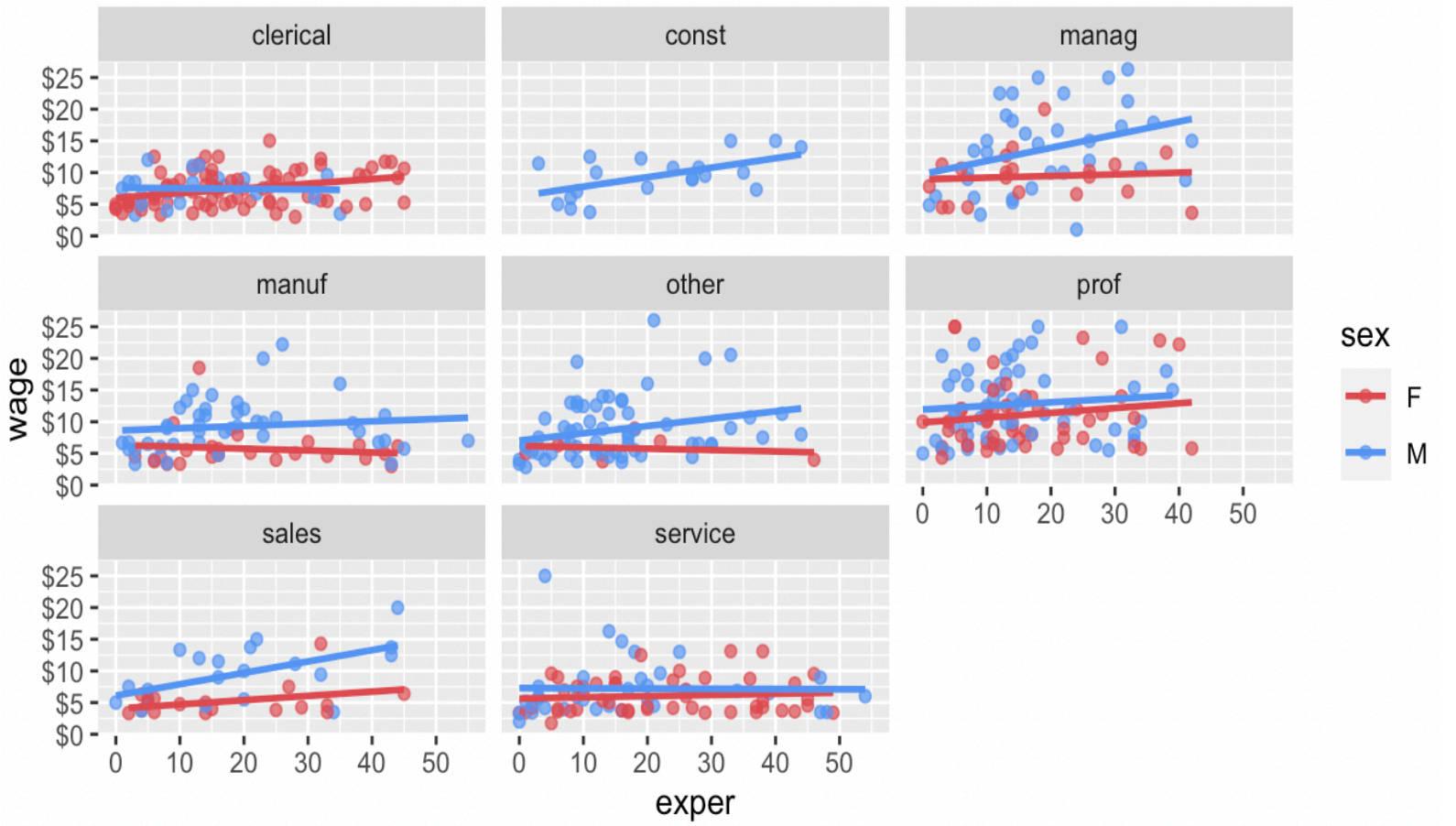
Example

```
# indicate sex using color
ggplot(data = plotdata,
       mapping = aes(x = exper,
                     y = wage,
                     color = sex)) +
  geom_point(alpha = .7,
             size = 3) +
  geom_smooth(method = "lm",
              se = FALSE,
              size = 1.5)
```



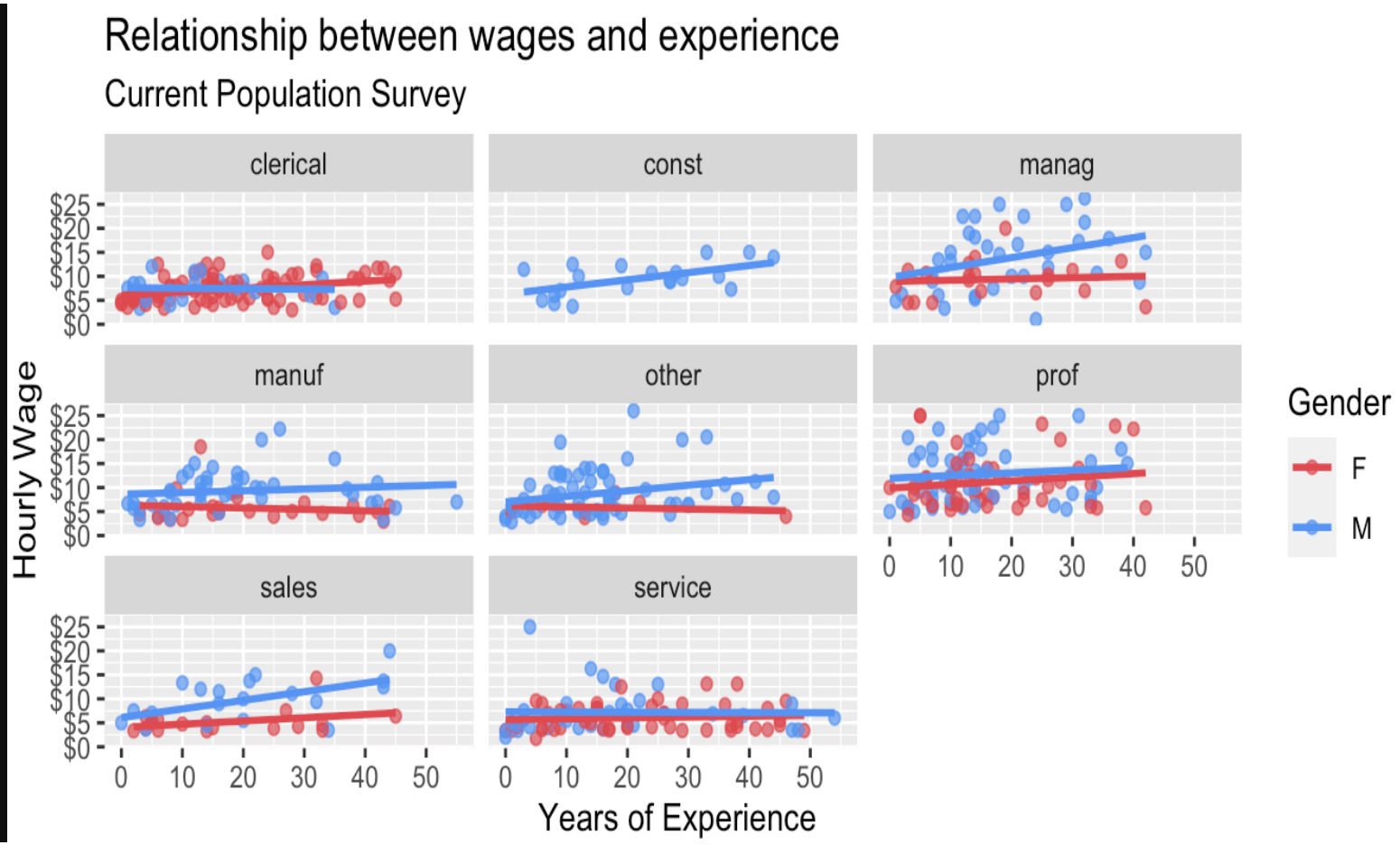
Example

```
# reproduce plot for each level of job sector
ggplot(data = plotdata,
       mapping = aes(x = exper,
                     y = wage,
                     color = sex)) +
  geom_point(alpha = .7) +
  geom_smooth(method = "lm",
              se = FALSE) +
  scale_x_continuous(breaks = seq(0, 60, 10)) +
  scale_y_continuous(breaks = seq(0, 30, 5),
                     label = scales::dollar) +
  scale_color_manual(values = c("indianred3",
                                "cornflowerblue")) +
  facet_wrap(~sector)
```



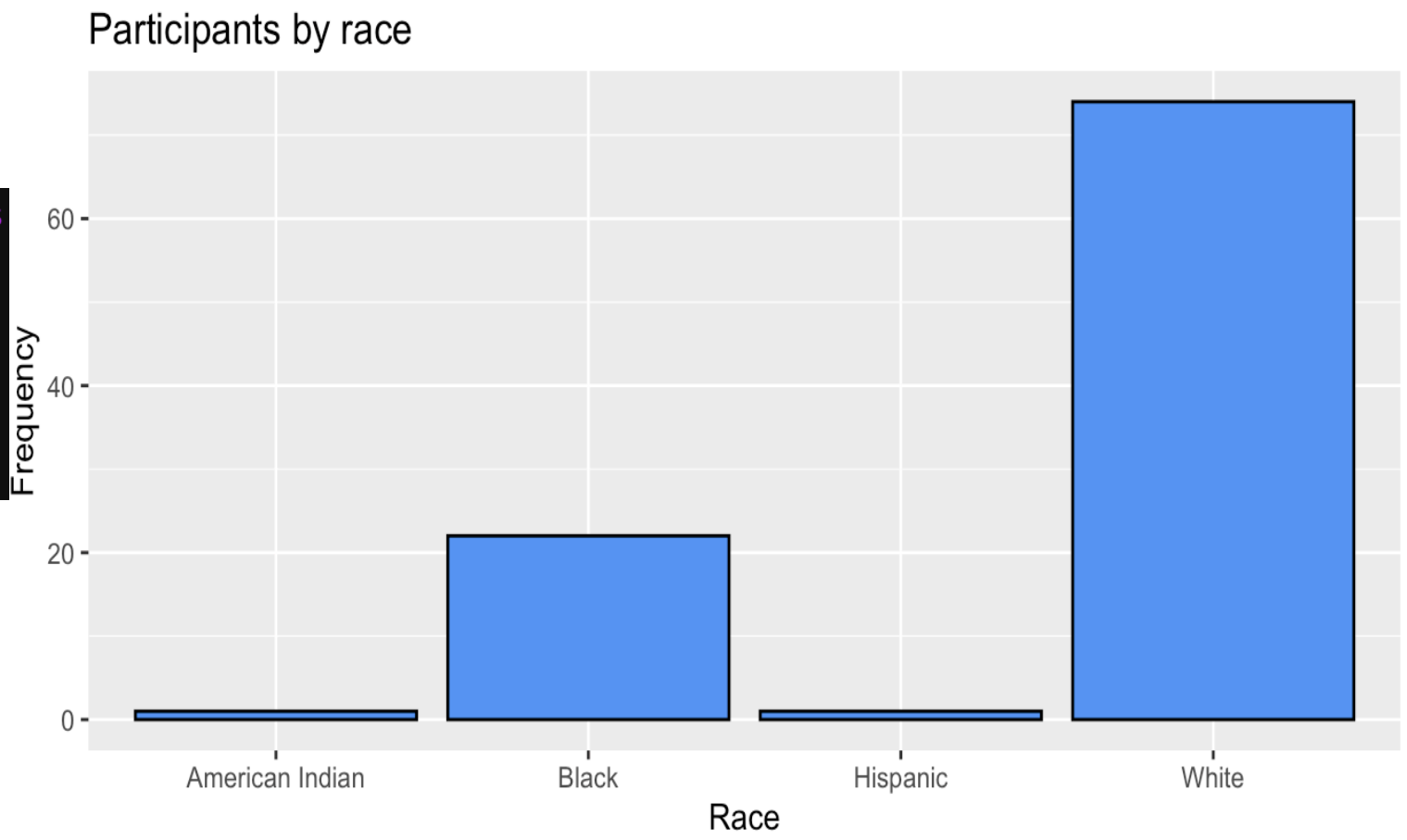
Examples

```
# add informative labels
ggplot(data = plotdata,
       mapping = aes(x = exper,
                     y = wage,
                     color = sex)) +
  geom_point(alpha = .7) +
  geom_smooth(method = "lm",
              se = FALSE) +
  scale_x_continuous(breaks = seq(0, 60, 10)) +
  scale_y_continuous(breaks = seq(0, 30, 5),
                     label = scales::dollar) +
  scale_color_manual(values = c("indianred3",
                                "cornflowerblue")) +
  facet_wrap(~sector) +
  labs(title = "Relationship between wages and experience",
       subtitle = "Current Population Survey",
       caption = "source: http://mosaic-web.org/",
       x = " Years of Experience",
       y = "Hourly Wage",
       color = "Gender")
```



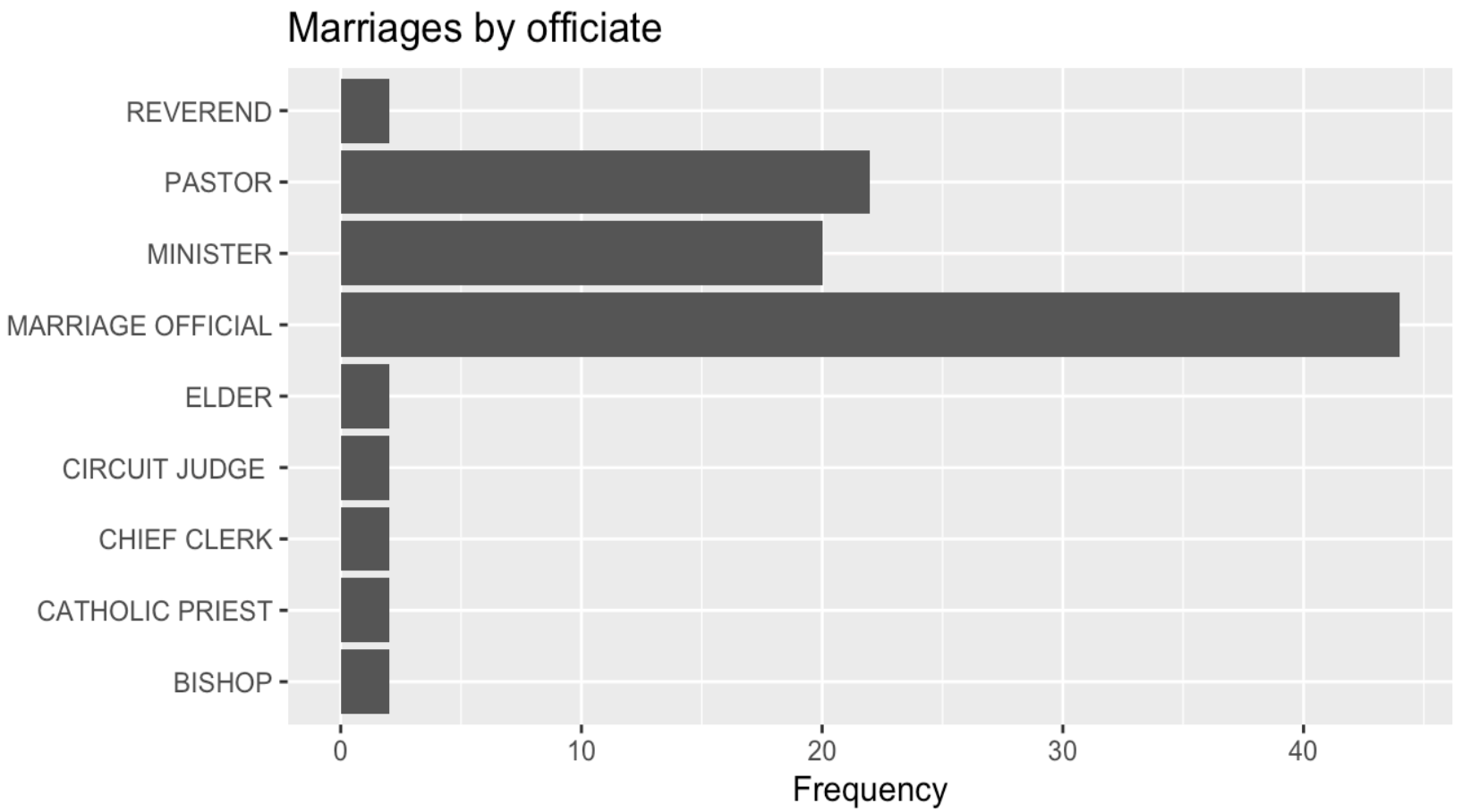
Example

```r
# plot the distribution of race with modified colors and labels
ggplot(Marriage, aes(x = race)) +
  geom_bar(fill = "cornflowerblue",
           color="black") +
  labs(x = "Race",
       y = "Frequency",
       title = "Participants by race")
```



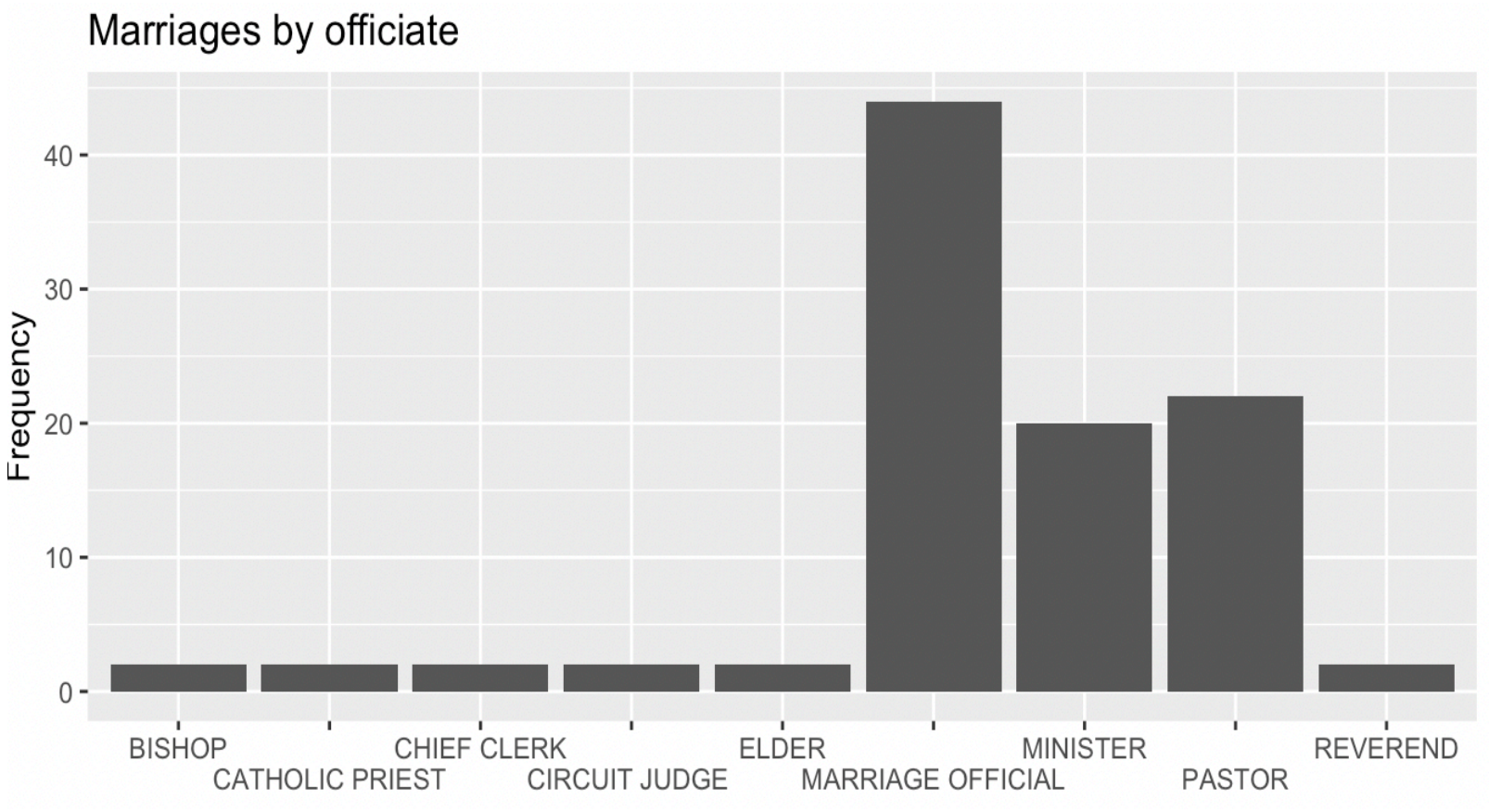
Example

```
# horizontal bar chart
ggplot(Marriage, aes(x = officialTitle)) +
  geom_bar() +
  labs(x = "",
       y = "Frequency",
       title = "Marriages by officiate") +
  coord_flip()
```



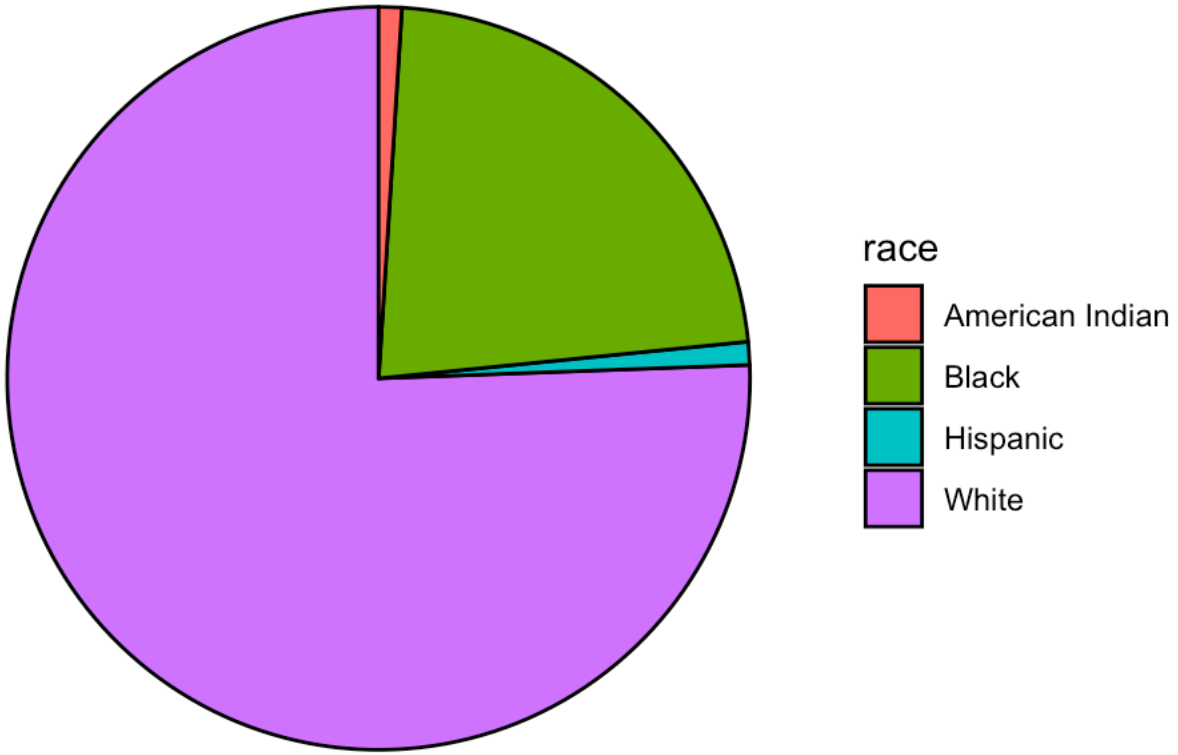
Example

```
# bar chart with staggered labels
lbls <- paste0(c("", "\n"),
               levels(Marriage$officialTitle))
ggplot(Marriage,
       aes(x=factor(officialTitle,
                    labels = lbls))) +
  geom_bar() +
  labs(x = "",
       y = "Frequency",
       title = "Marriages by officiate")
```



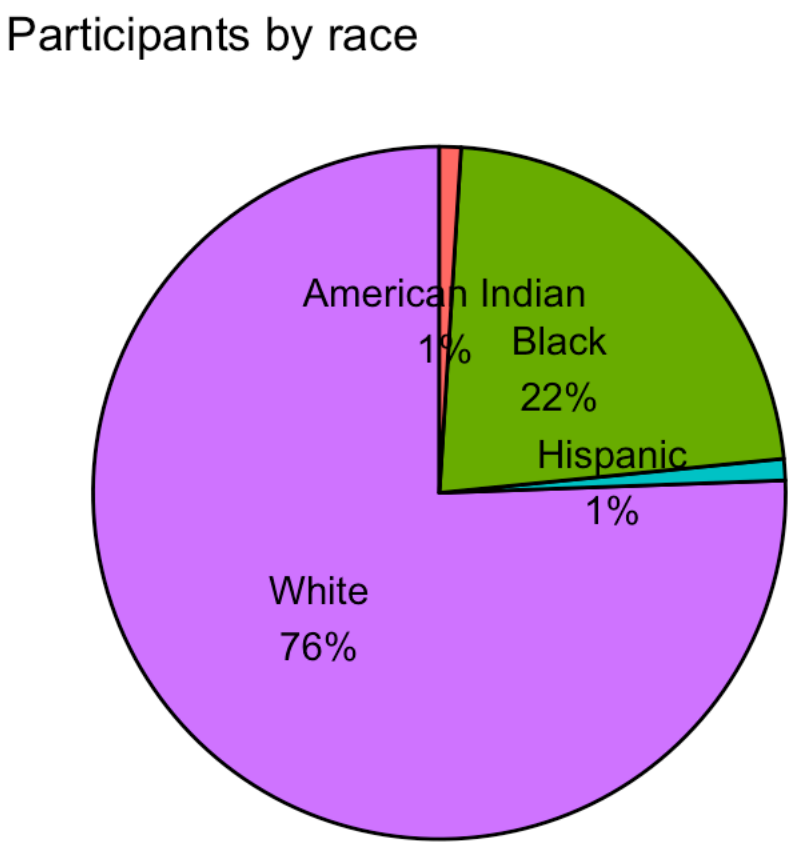
Example

```
# create a basic ggplot2 pie chart
plotdata <- Marriage %>%
  count(race) %>%
  arrange(desc(race)) %>%
  mutate(prop = round(n * 100 / sum(n), 1),
         lab.ypos = cumsum(prop) - 0.5  *prop)

ggplot(plotdata,
       aes(x = "",
           y = prop,
           fill = race)) +
  geom_bar(width = 1,
           stat = "identity",
           color = "black") +
  coord_polar("y",
              start = 0,
              direction = -1) +
  theme_void()
```



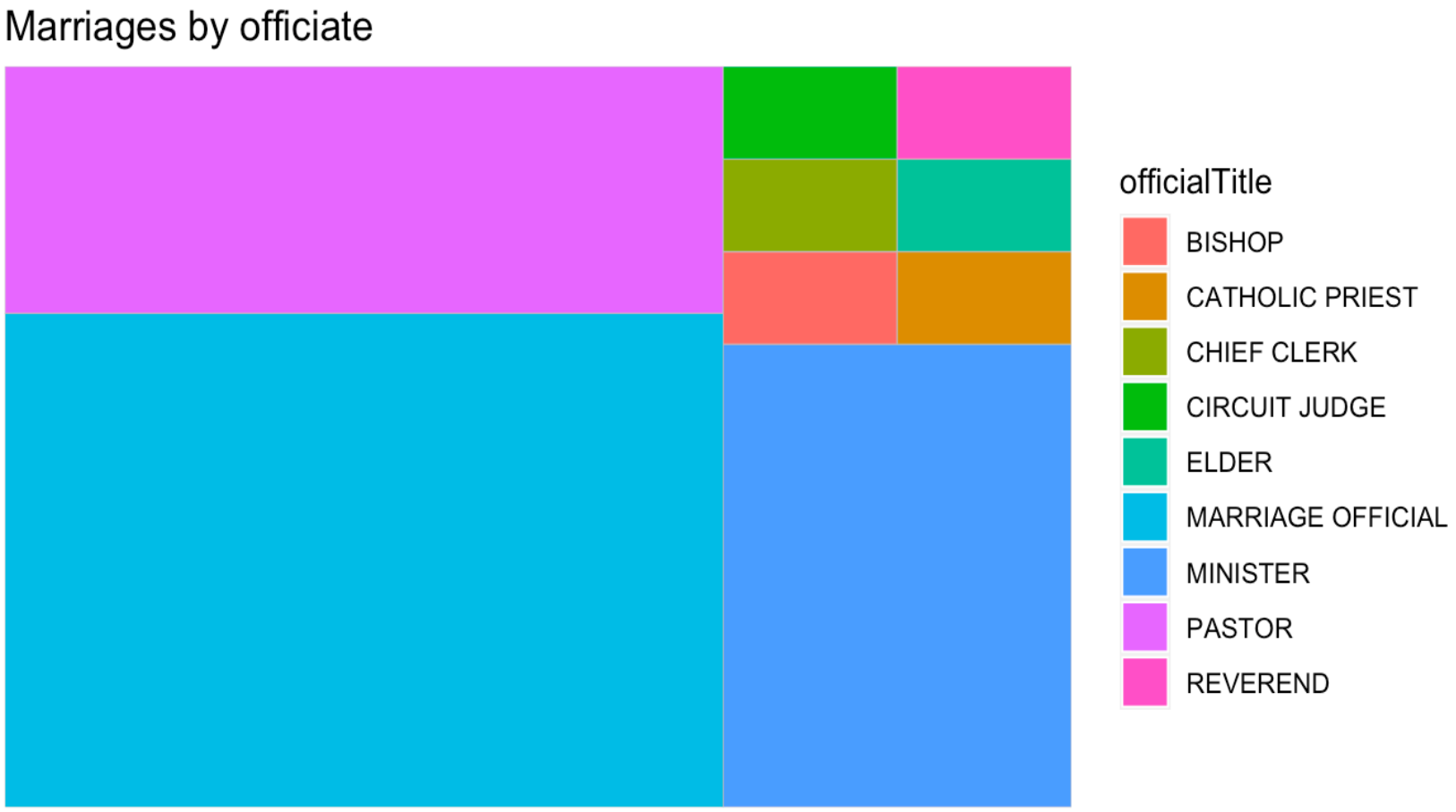
Example

```
# create a pie chart with slice labels
plotdata <- Marriage %>%
  count(race) %>%
  arrange(desc(race)) %>%
  mutate(prop = round(n*100/sum(n), 1),
         lab.ypos = cumsum(prop) - 0.5*prop)

plotdata$label <- paste0(plotdata$race, "\n",
                         round(plotdata$prop), "%")

ggplot(plotdata,
       aes(x = "",
           y = prop,
           fill = race)) +
  geom_bar(width = 1,
           stat = "identity",
           color = "black") +
  geom_text(aes(y = lab.ypos, label = label),
            color = "black") +
  coord_polar("y",
              start = 0,
              direction = -1) +
  theme_void() +
  theme(legend.position = "FALSE") +
  labs(title = "Participants by race")
```



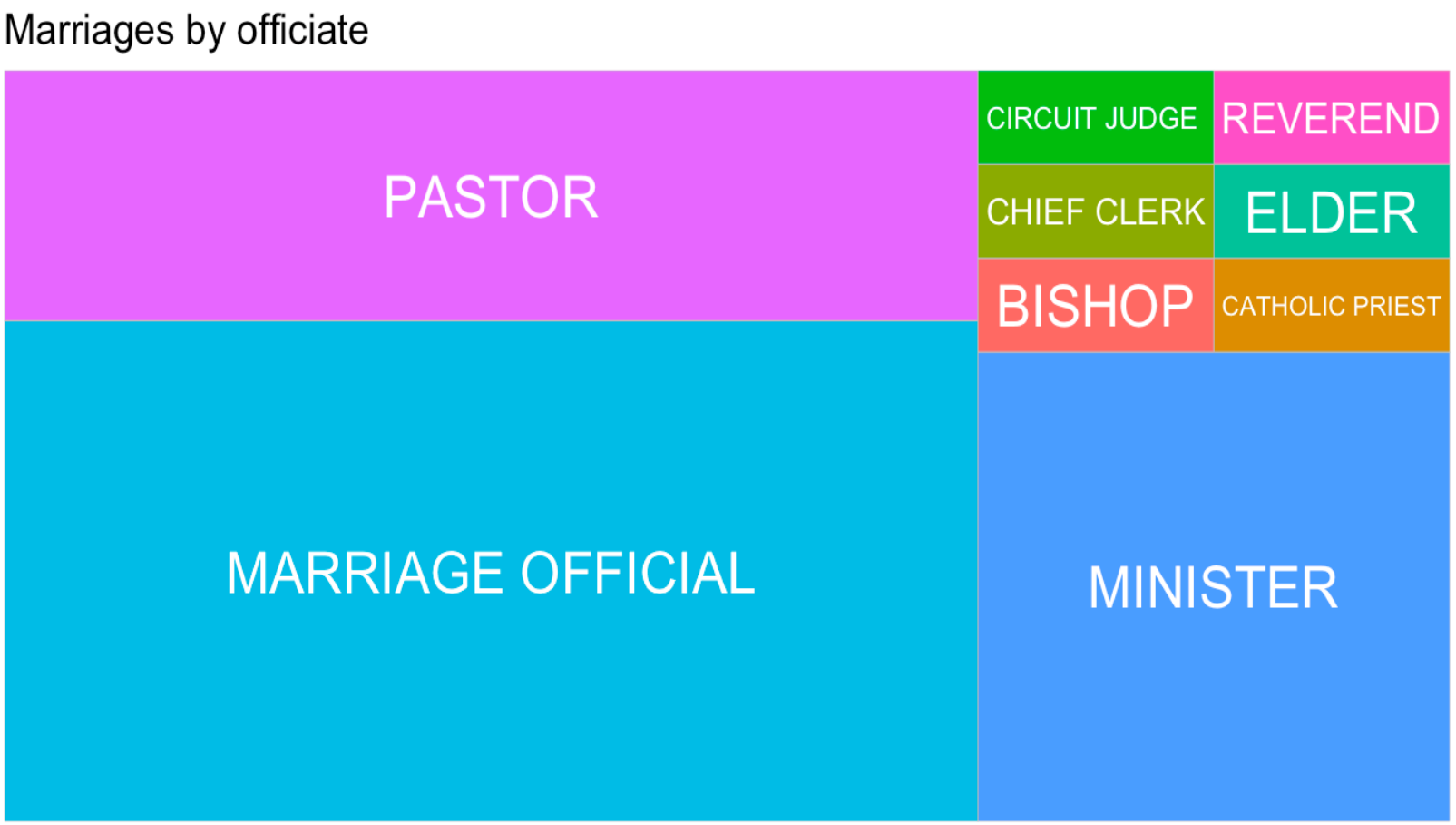
Example

```
library(treemapify)

# create a treemap of marriage officials
plotdata <- Marriage %>%
  count(officialTitle)

ggplot(plotdata,
       aes(fill = officialTitle,
           area = n)) +
  geom_treemap() +
  labs(title = "Marriages by officiate")
```



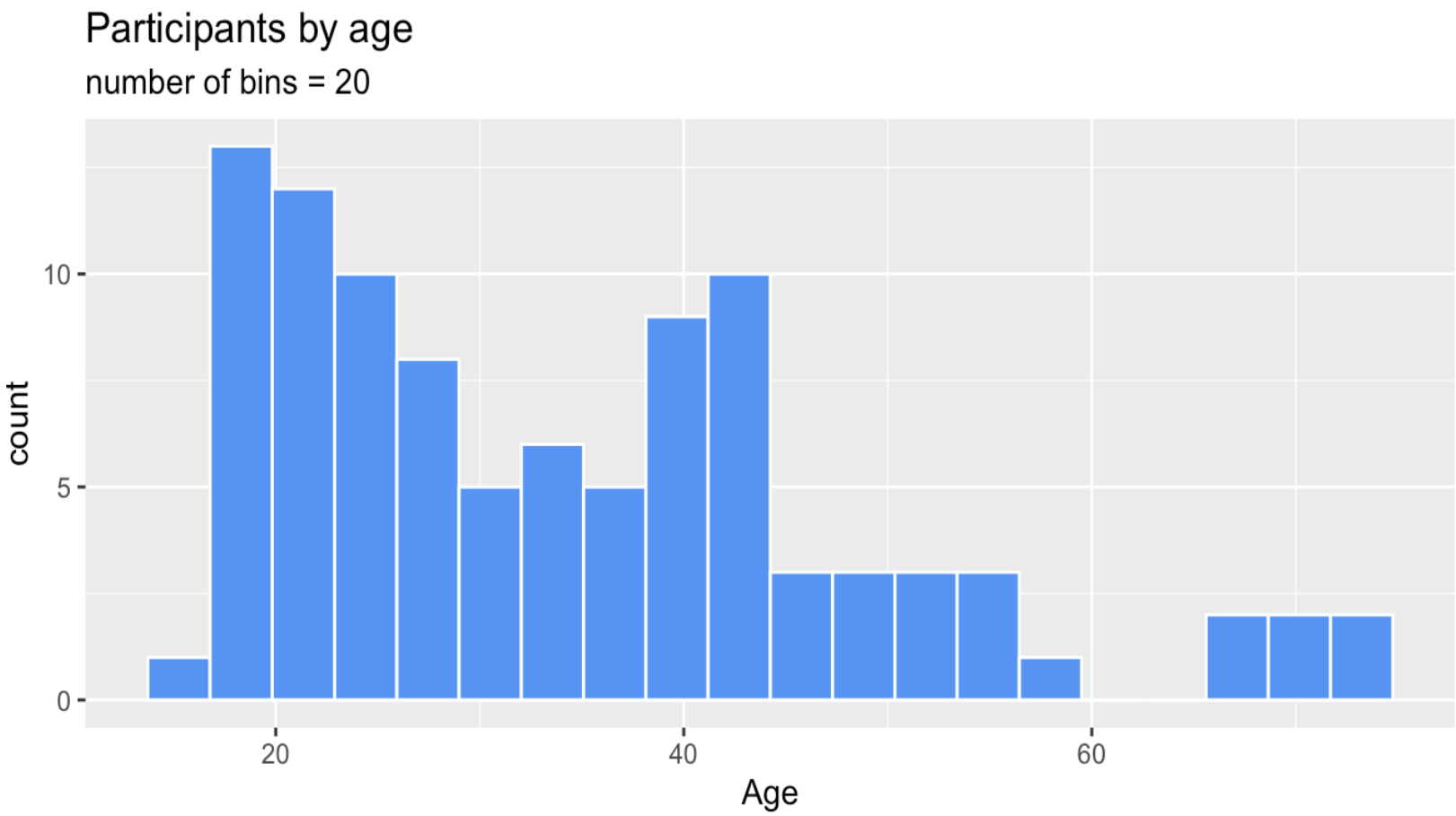
Example

```
# create a treemap with tile labels
ggplot(plotdata,
       aes(fill = officialTitle,
           area = n,
           label = officialTitle)) +
  geom_treemap() +
  geom_treemap_text(colour = "white",
                    place = "centre") +
  labs(title = "Marriages by officiate") +
  theme(legend.position = "none")
```



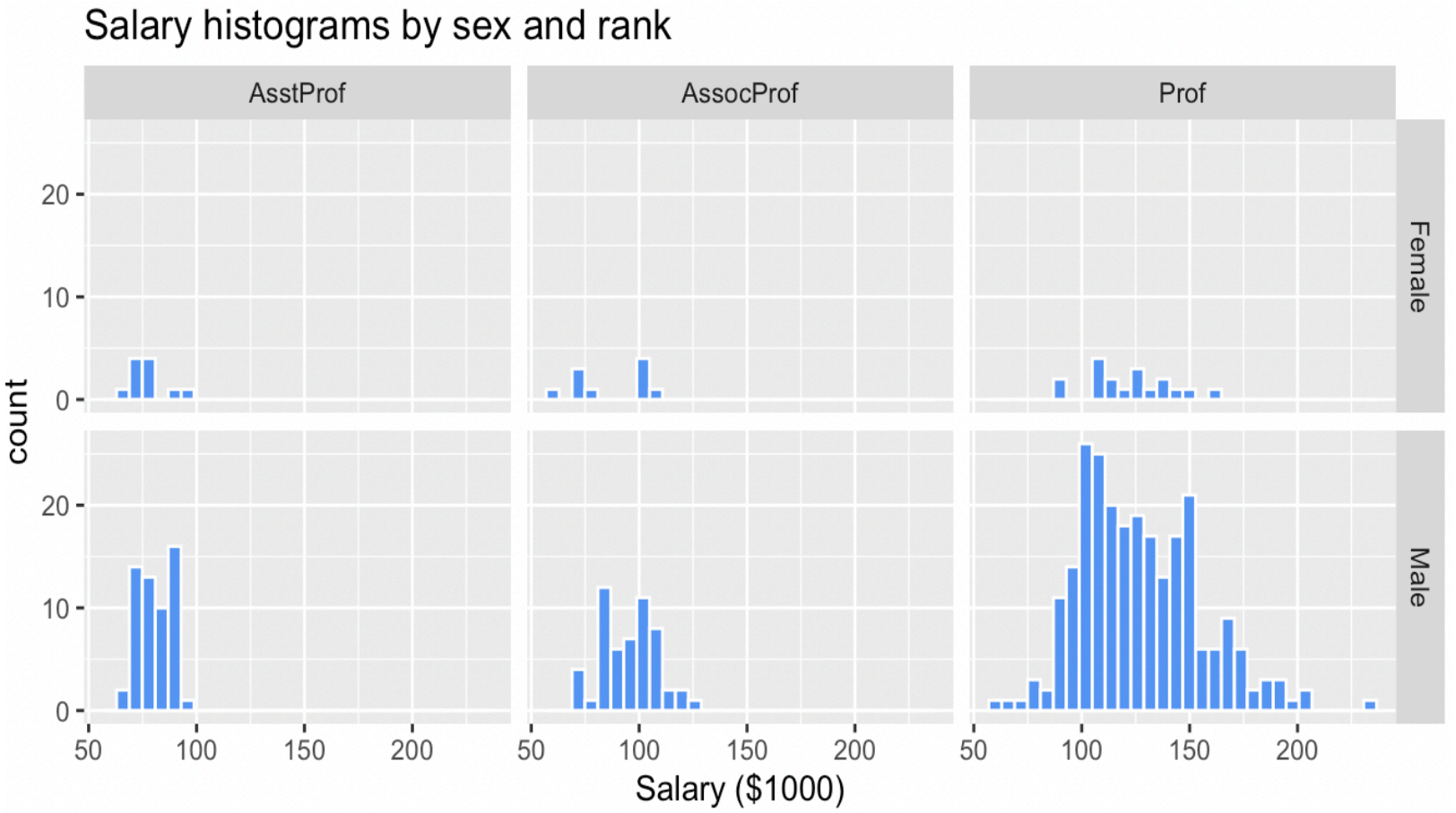
Example

```
# plot the histogram with 20 bins
ggplot(Marriage, aes(x = age)) +
  geom_histogram(fill = "cornflowerblue",
                 color = "white",
                 bins = 20) +
  labs(title="Participants by age",
       subtitle = "number of bins = 20",
       x = "Age")
```



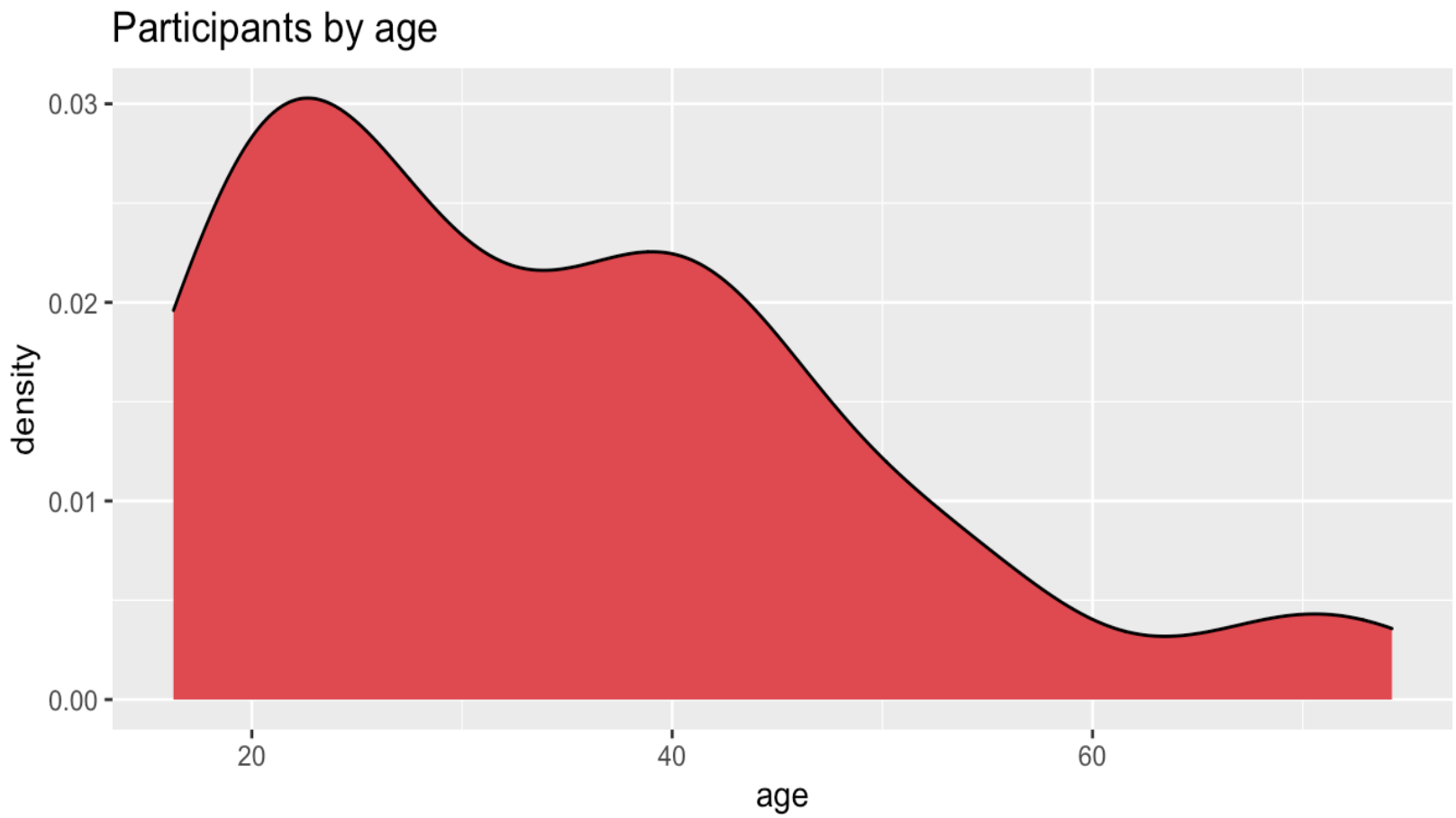
Example

```
# plot salary histograms by rank and sex
library(dplyr)
data(Salaries, package="carData")
ggplot(Salaries, aes(x = salary / 1000)) +
  geom_histogram(color = "white",
                 fill = "cornflowerblue") +
  facet_grid(sex ~ rank) +
  labs(title = "Salary histograms by sex and rank",
       x = "Salary ($1000)")
```



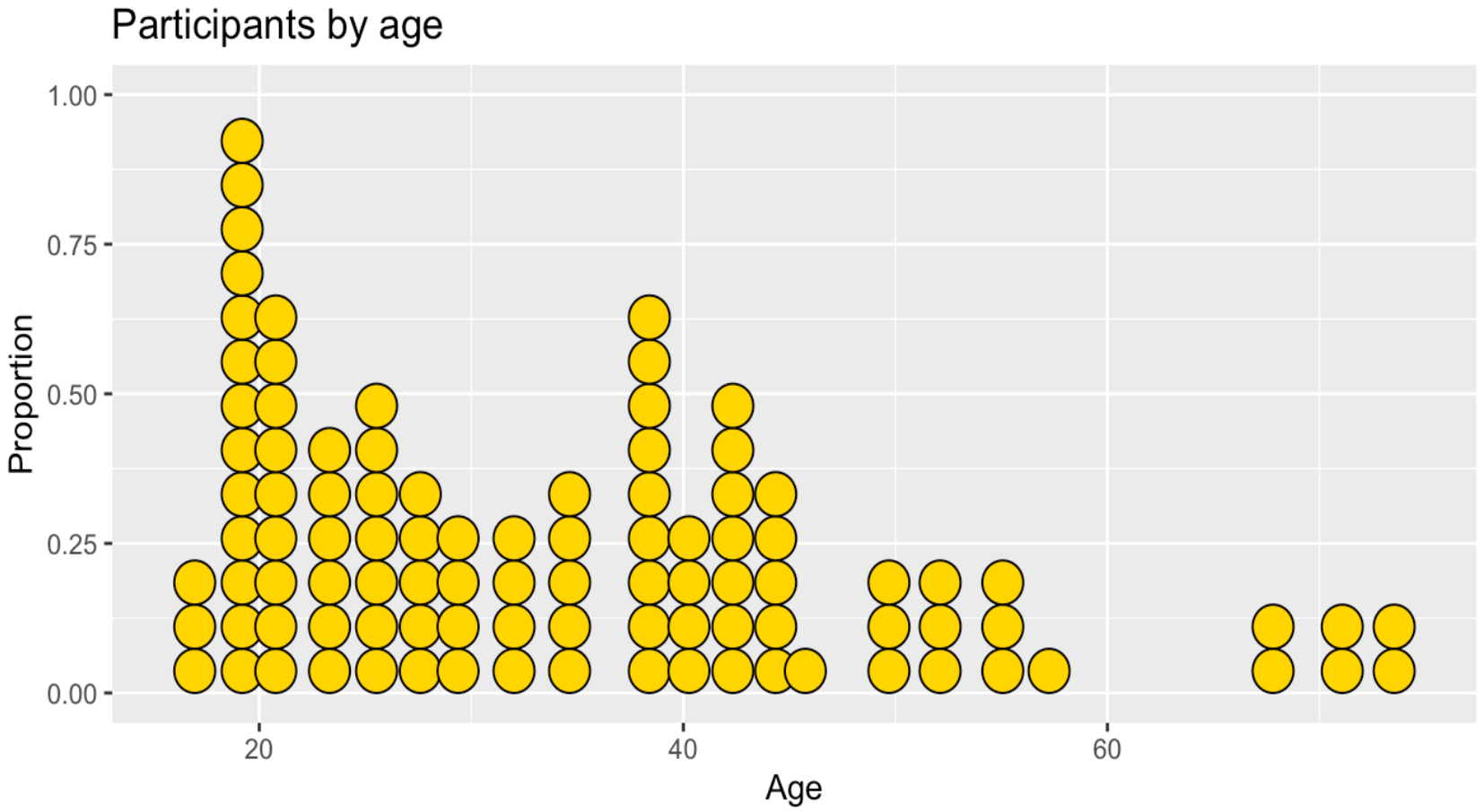
Example

```
# Create a kernel density plot of age
ggplot(Marriage, aes(x = age)) +
  geom_density(fill = "indianred3") +
  labs(title = "Participants by age")
```



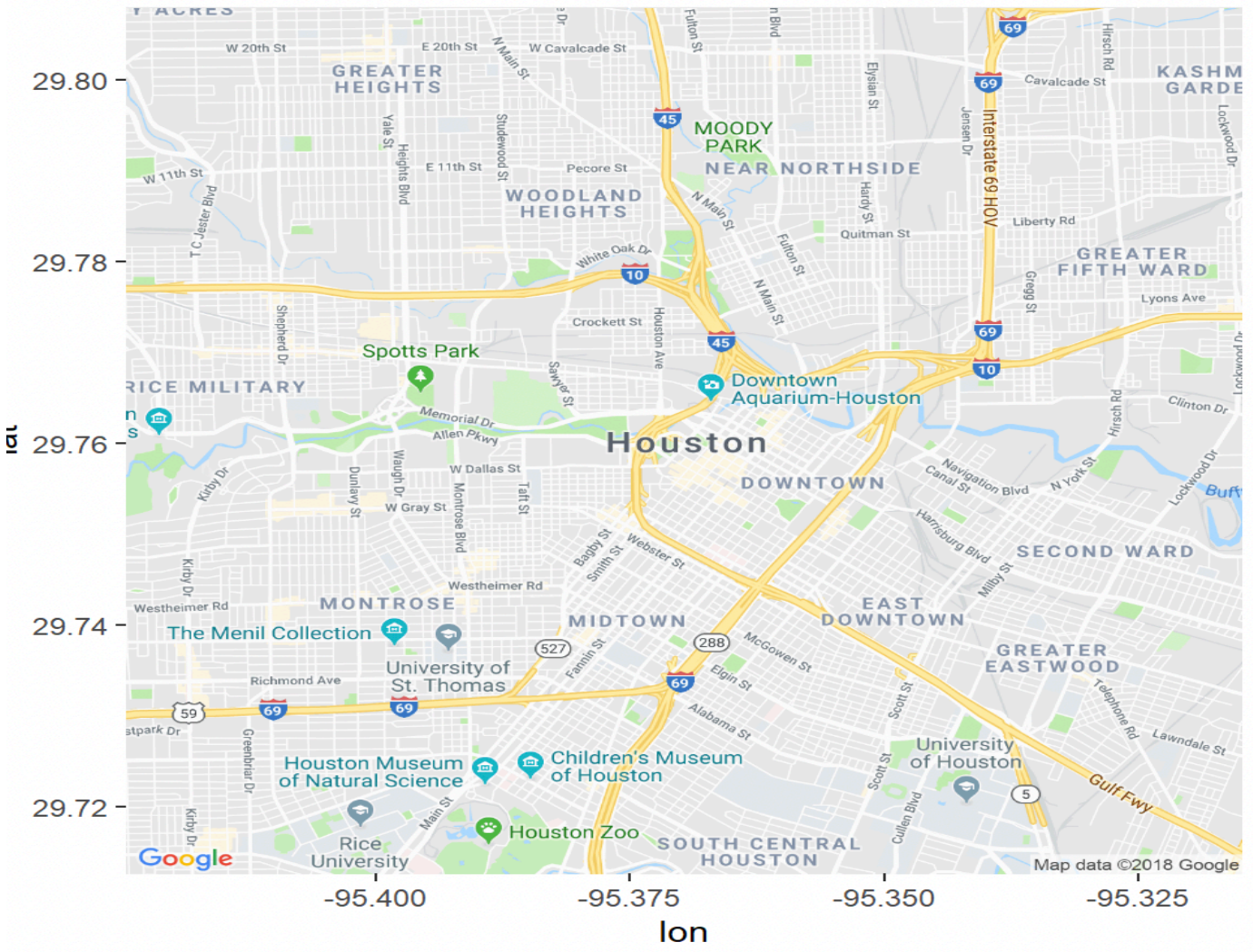
Example

```
# Plot ages as a dot plot using
# gold dots with black borders
ggplot(Marriage, aes(x = age)) +
  geom_dotplot(fill = "gold",
               color = "black") +
  labs(title = "Participants by age",
       y = "Proportion",
       x = "Age")
```



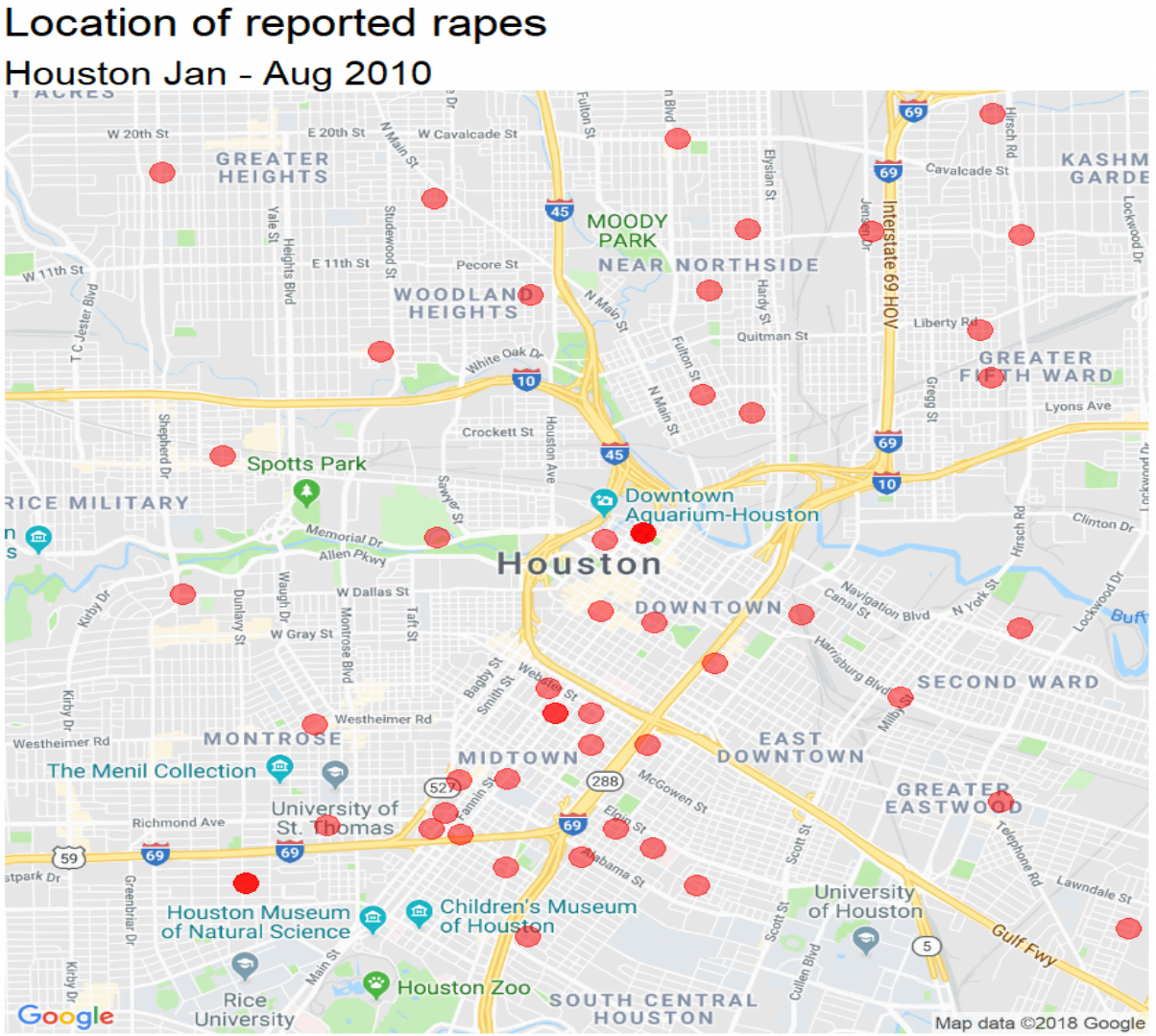
Example

```
library(ggmap)
library(dplyr)
data(crime, package="ggmap")
rapes <- filter(crime, offense == "rape") %>%
  select(date, offense, address, lon, lat)

# view data
head(rapes)

# using geocode function returns
# lon=-95.3698, lat=29.76043
houston_center <- geocode("Houston, TX")

houston_map <- get_map(houston_center,
                       zoom = 13,
                       maptype = "roadmap")
ggmap(houston_map)
```



Example

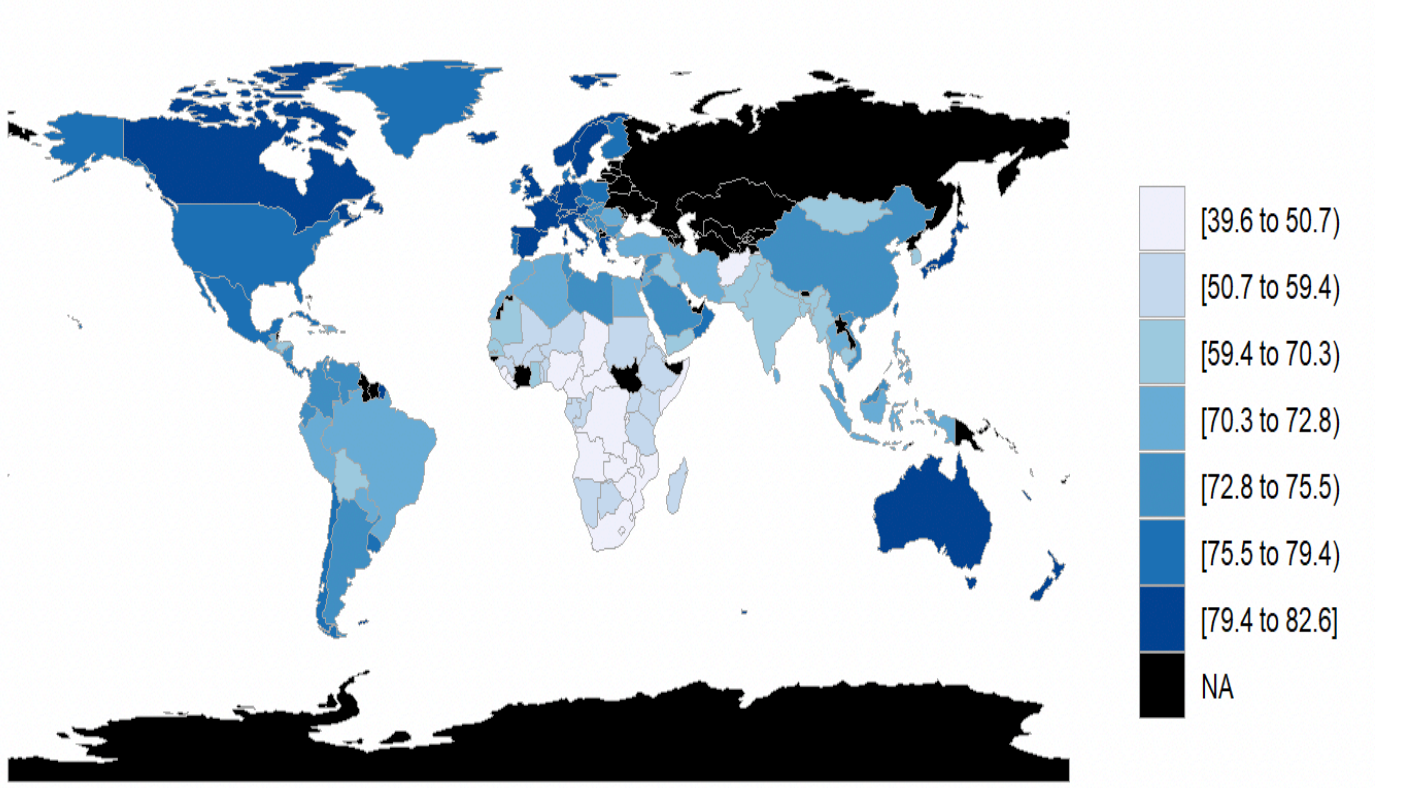
```
# remove long and lat numbers and add titles
ggmap(houston_map,
      base_layer = ggplot(aes(x=lon, y = lat),
                          data = rapes)) +
  geom_point(color = "red",
             size = 3,
             alpha = 0.5) +
  theme_void() +
  labs(title = "Location of reported rapes",
       subtitle = "Houston Jan - Aug 2010",
       caption = "source: http://www.houstontx.gov/police/cs/")
```



source: http://www.houstontx.gov/police/cs/

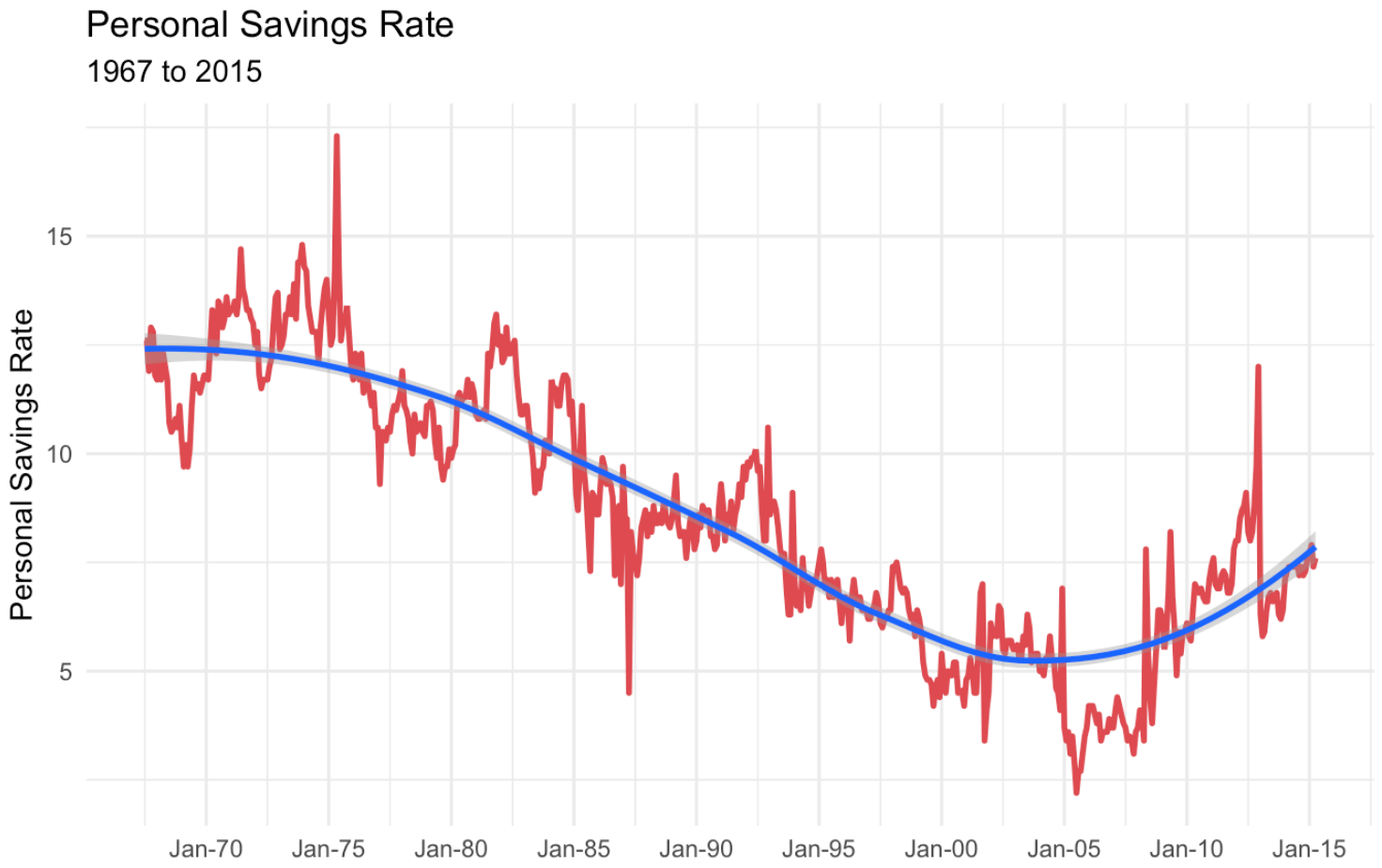
Example

```r
# view the first 12 region names in country.map
library(choroplethr)
data(country.map, package = "choroplethrMaps")
head(unique(country.map$region), 12)

# prepare dataset
data(gapminder, package = "gapminder")
plotdata <- gapminder %>%
  filter(year == 2007) %>%
  rename(region = country,
         value = lifeExp) %>%
  mutate(region = tolower(region)) %>%
  mutate(region = recode(region,
                         "united states"    = "united states of america",
                         "congo, dem. rep." = "democratic republic of the congo",
                         "congo, rep."      = "republic of congo",
                         "korea, dem. rep." = "south korea",
                         "korea. rep."      = "north korea",
                         "tanzania"         = "united republic of tanzania",
                         "serbia"           = "republic of serbia",
                         "slovak republic"  = "slovakia",
                         "yemen, rep."      = "yemen"))
country_choropleth(plotdata)
```



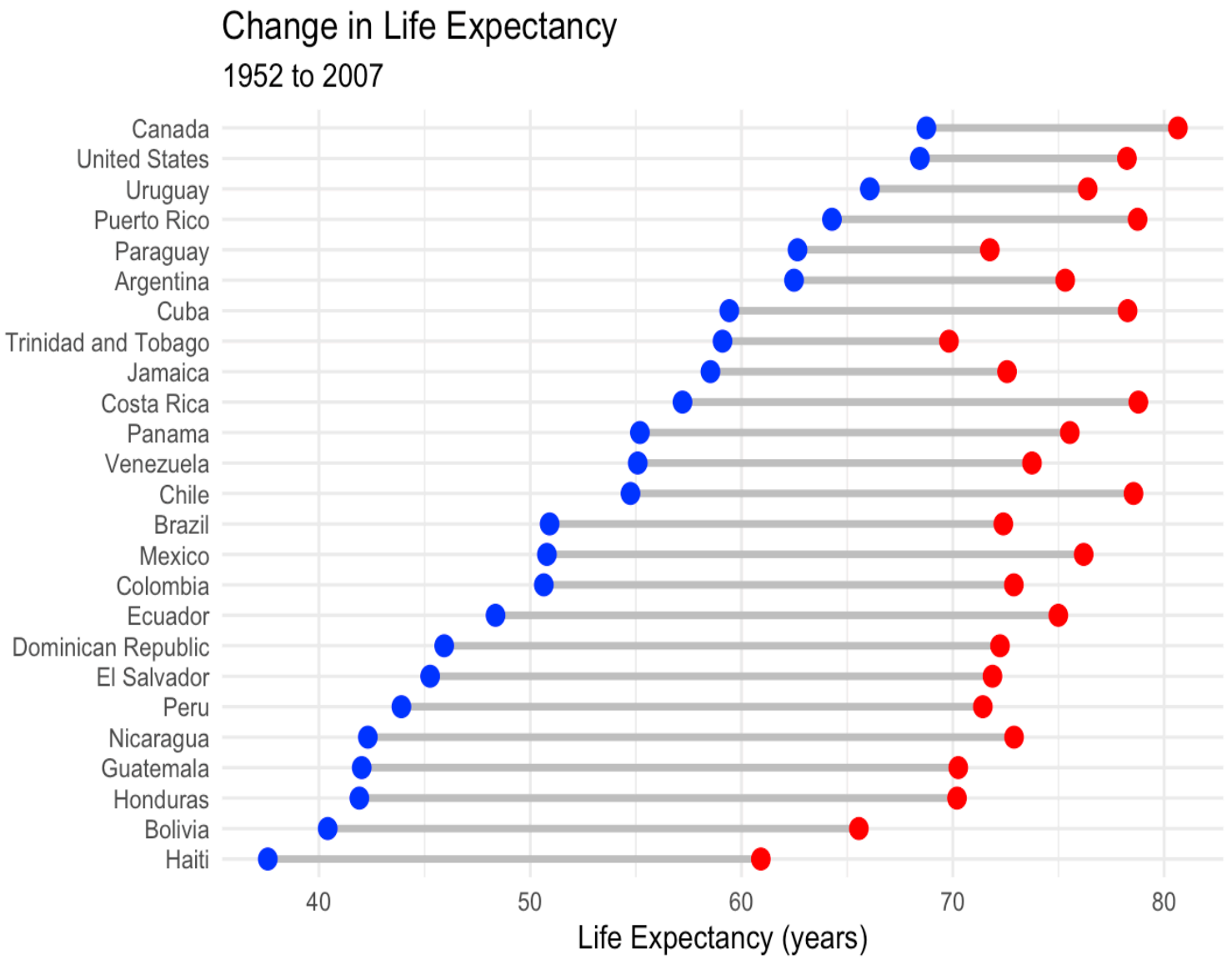
Example

```
library(scales)
ggplot(economics, aes(x = date, y = psavert)) +
  geom_line(color = "indianred3",
            size=1 ) +
  geom_smooth() +
  scale_x_date(date_breaks = '5 years',
               labels = date_format("%b-%y")) +
  labs(title = "Personal Savings Rate",
       subtitle = "1967 to 2015",
       x = "",
       y = "Personal Savings Rate") +
  theme_minimal()
```



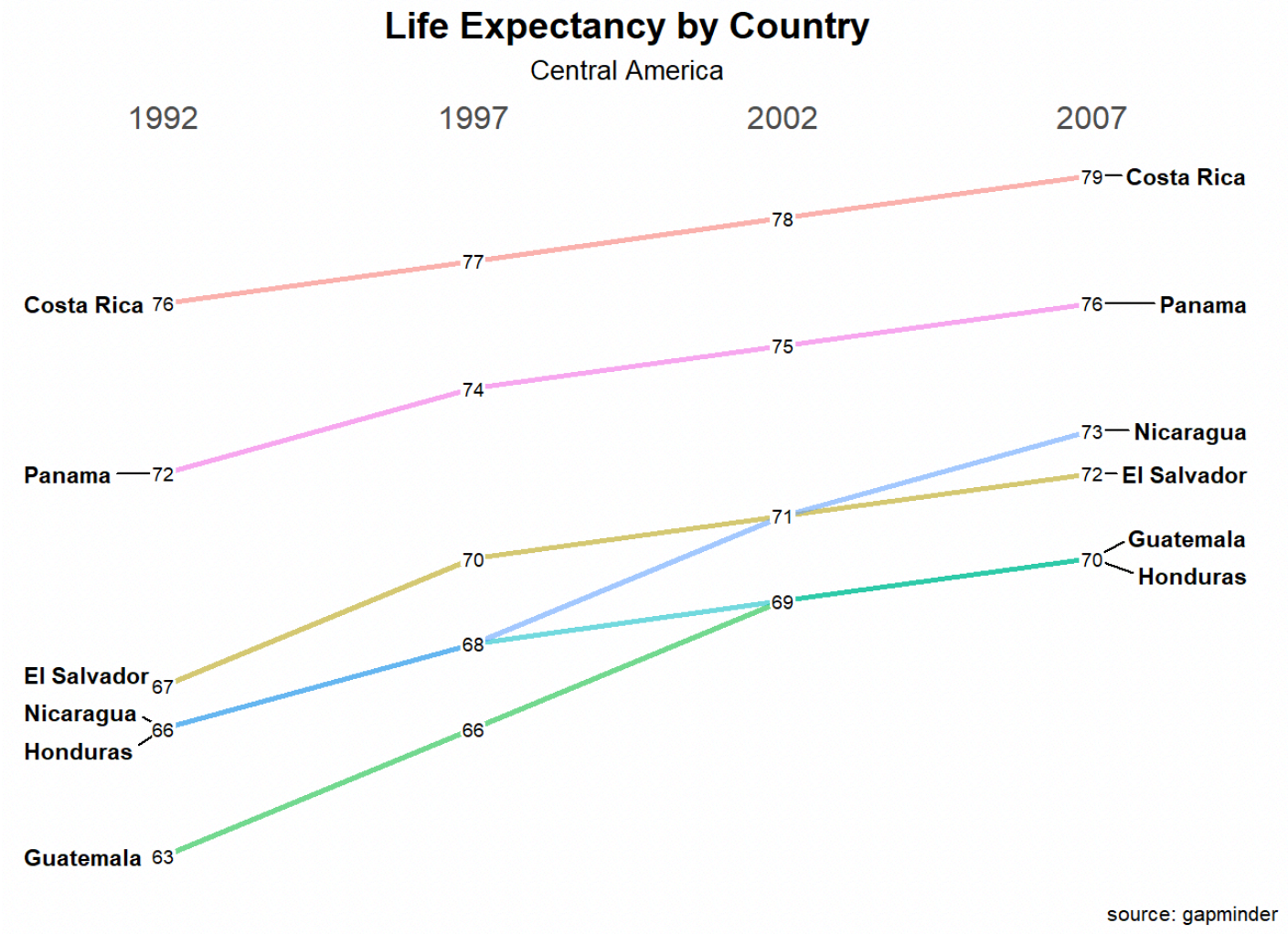
Example

```r
# create dumbbell plot
library(ggalt)
library(tidyr)
library(dplyr)
library(gapminder)

# load data
data(gapminder, package = "gapminder")

# subset data
plotdata_long <- filter(gapminder,
                        continent == "Americas" &
                          year %in% c(1952, 2007)) %>%
  select(country, year, lifeExp)

# convert data to wide format
plotdata_wide <- spread(plotdata_long, year, lifeExp)
names(plotdata_wide) <- c("country", "y1952", "y2007")

ggplot(plotdata_wide,
       aes(y = reorder(country, y1952),
           x = y1952,
           xend = y2007)) +
  geom_dumbbell(size = 1.2,
                size_x = 3,
                size_xend = 3,
                colour = "grey",
                colour_x = "blue",
                colour_xend = "red") +
  theme_minimal() +
  labs(title = "Change in Life Expectancy",
       subtitle = "1952 to 2007",
       x = "Life Expectancy (years)",
       y = "")
```



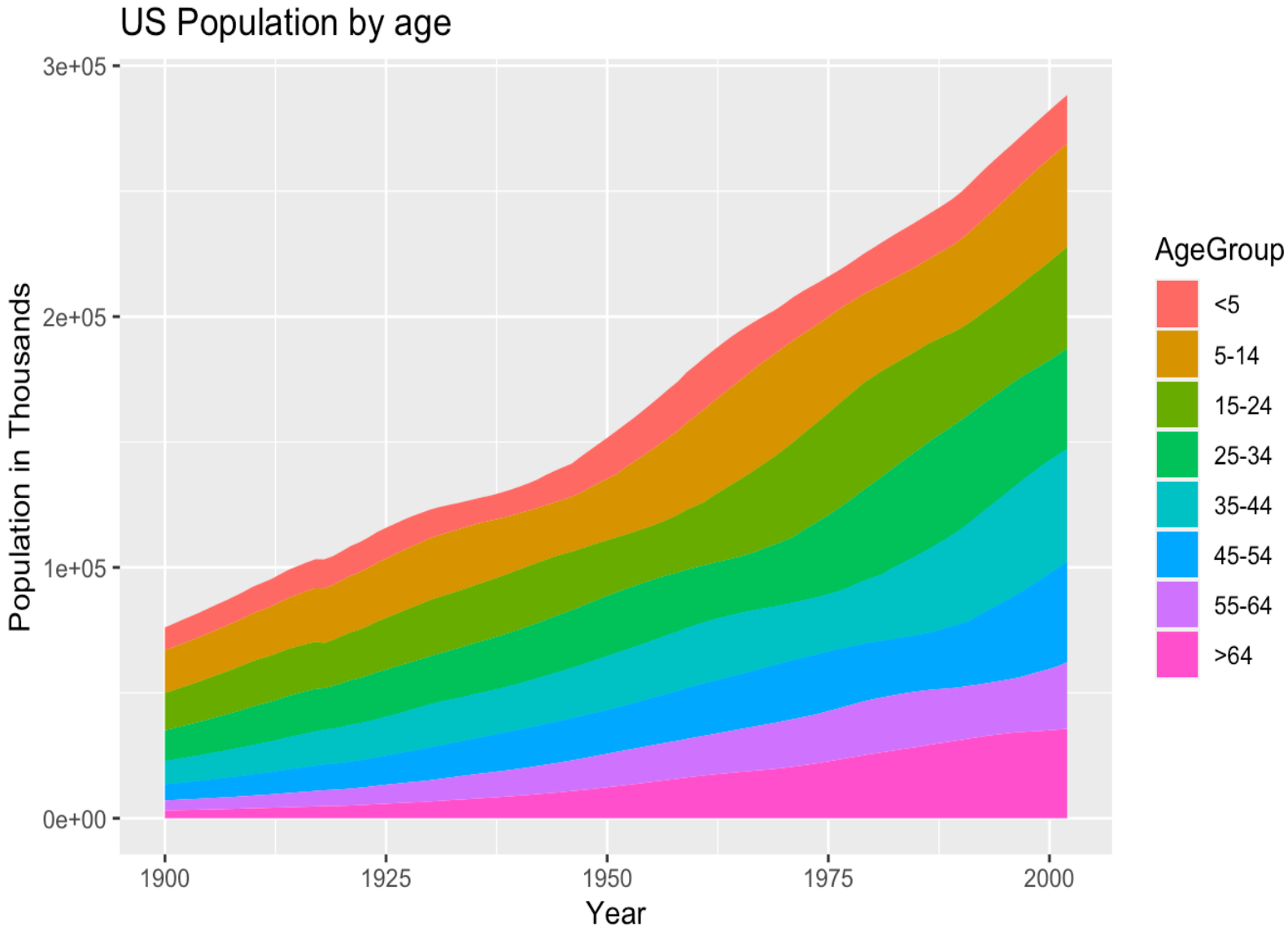
Example

```
#slope graphs
library(CGPfunctions)
# Select Central American countries data
# for 1992, 1997, 2002, and 2007

df <- gapminder %>%
  filter(year %in% c(1992, 1997, 2002, 2007) &
           country %in% c("Panama", "Costa Rica",
                          "Nicaragua", "Honduras",
                          "El Salvador", "Guatemala",
                          "Belize")) %>%
  mutate(year = factor(year),
         lifeExp = round(lifeExp))

# create slope graph
newggslopegraph(df, year, lifeExp, country) +
  labs(title="Life Expectancy by Country",
       subtitle="Central America",
       caption="source: gapminder")
```



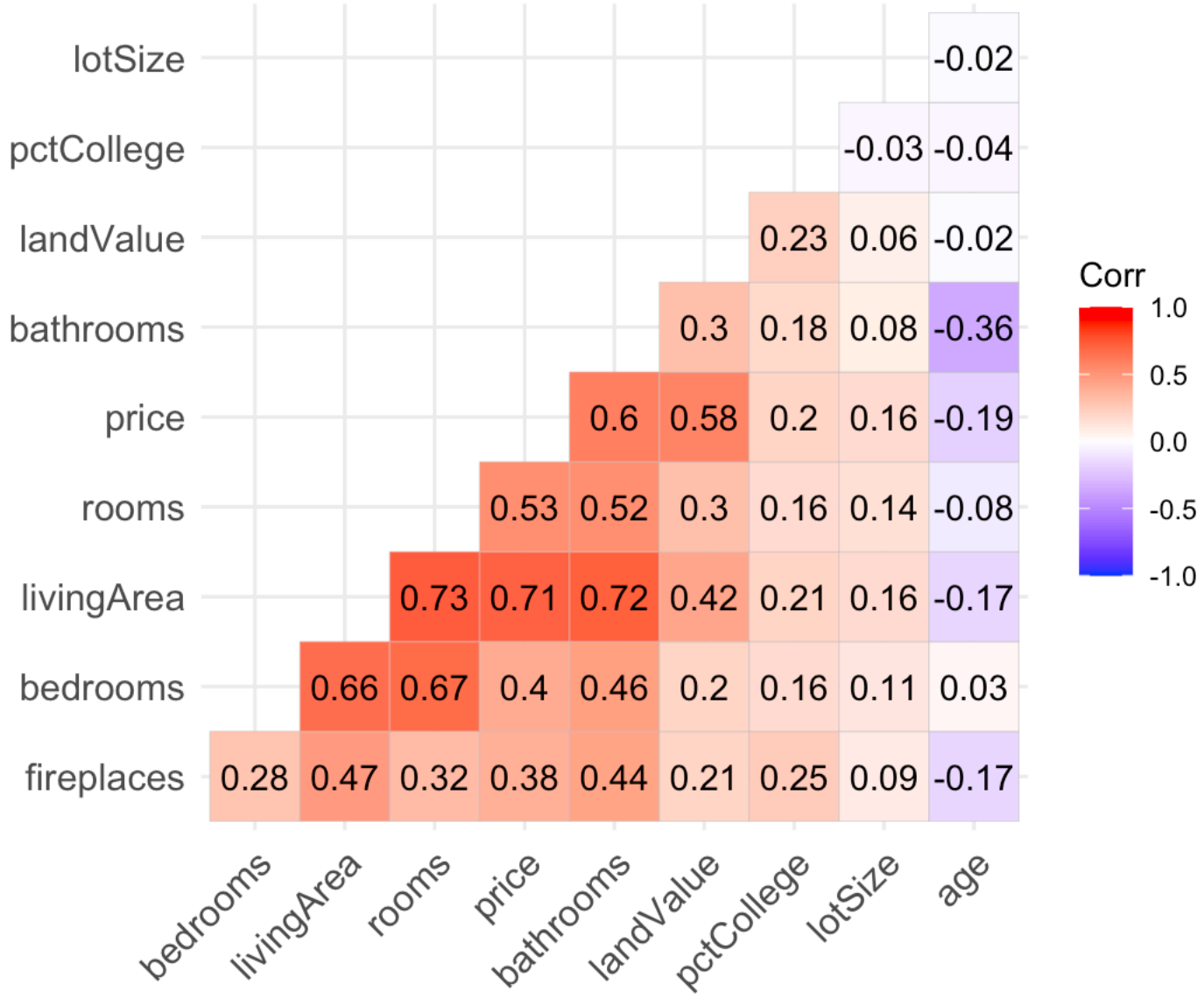
Example

```
# stacked area chart
library(gcookbook)
data(uspopage, package = "gcookbook")
ggplot(uspopage, aes(x = Year,
                     y = Thousands,
                     fill = AgeGroup)) +
  geom_area() +
  labs(title = "US Population by age",
       x = "Year",
       y = "Population in Thousands")
```



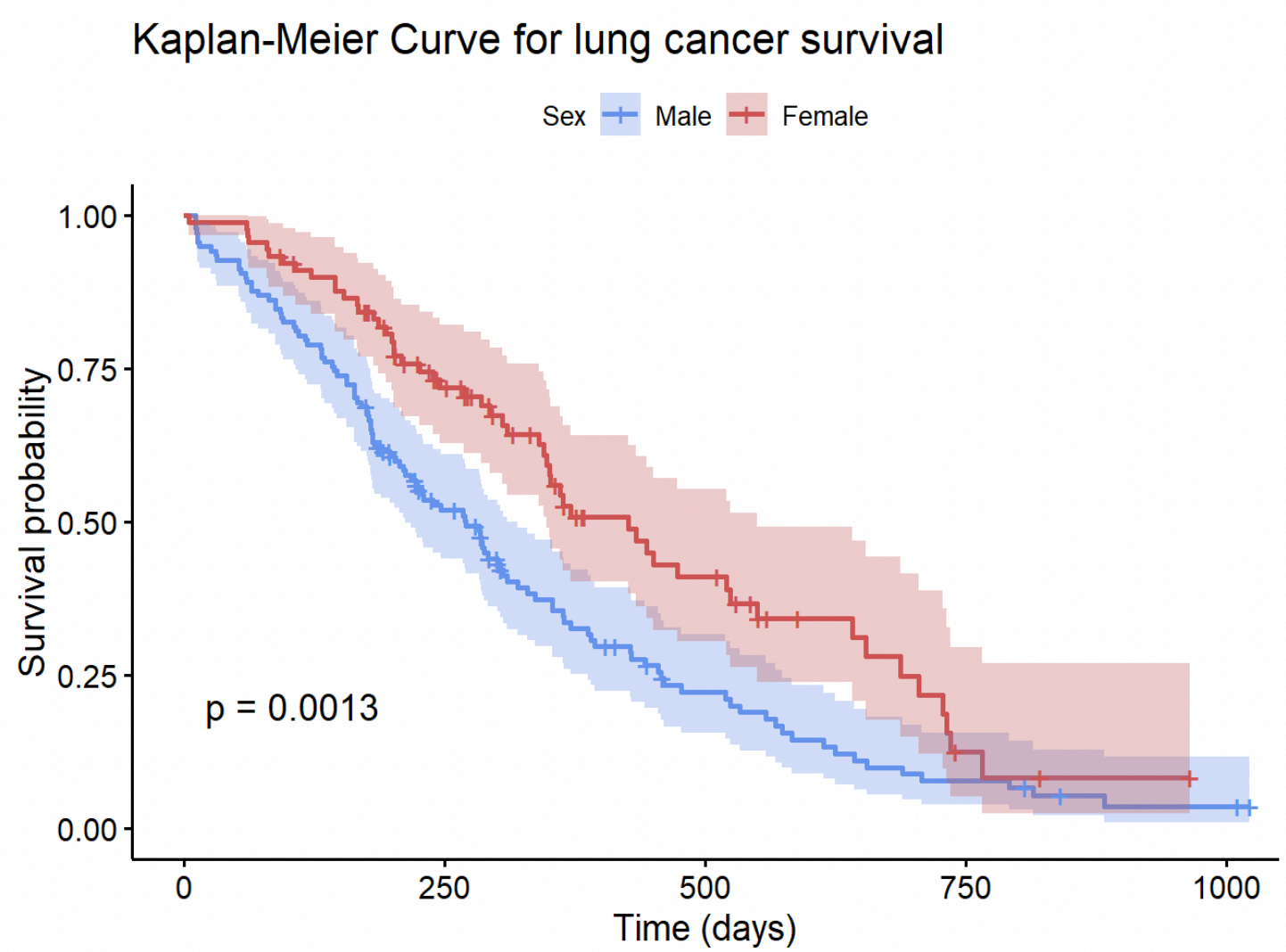
Example

```r
#coorelation plots
library(ggcorrplot)
data(SaratogaHouses, package="mosaicData")

# select numeric variables
df <- dplyr::select_if(SaratogaHouses, is.numeric)

# calulate the correlations
r <- cor(df, use="complete.obs")

ggcorrplot(r,
           hc.order = TRUE,
           type = "lower",
           lab = TRUE)
```



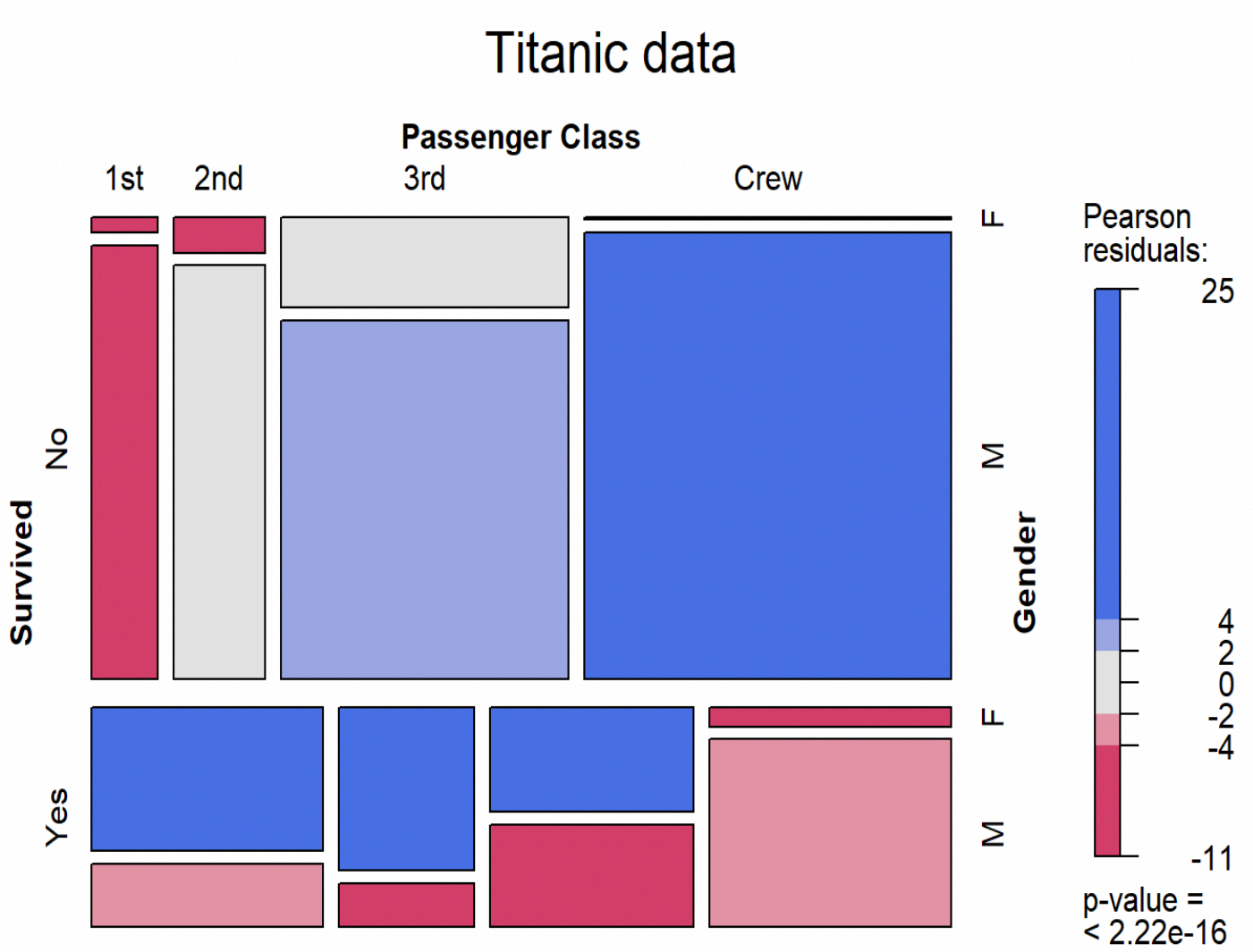
Example

```r
# plot survival curve for men and women
library(survival)
library(survminer)

data(lung)
sfit <- surv_fit(Surv(time, status) ~  sex, data=lung)
ggsurvplot(sfit,
           conf.int=TRUE,
           pval=TRUE,
           legend.labs=c("Male", "Female"),
           legend.title="Sex",
           palette=c("cornflowerblue", "indianred3"),
           title="Kaplan-Meier Curve for lung cancer survival",
           xlab = "Time (days)")
```



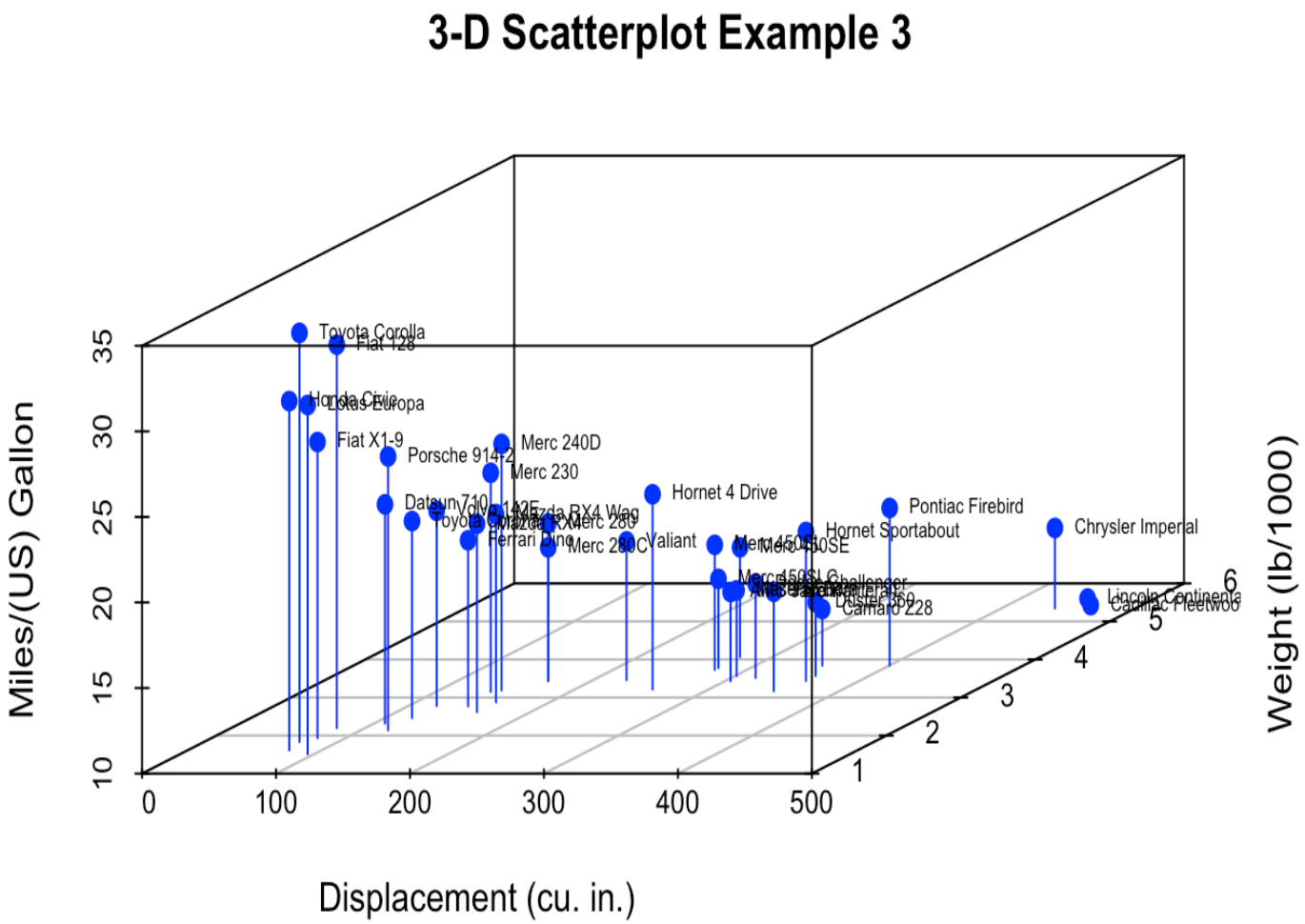
Example

```r
# mosaic graph
library(readr)
library(vcd)
data("Titanic")

# create a table
tbl <- xtabs(~Survived + Class + Sex, Titanic)
mosaic(tbl,
       shade = TRUE,
       legend = TRUE,
       labeling_args = list(set_varnames = c(Sex = "Gender",
                                             Survived = "Survived",
                                             Class = "Passenger Class")),
       set_labels = list(Survived = c("No", "Yes"),
                         Class = c("1st", "2nd", "3rd", "Crew"),
                         Sex = c("F", "M")),
       main = "Titanic data")
```



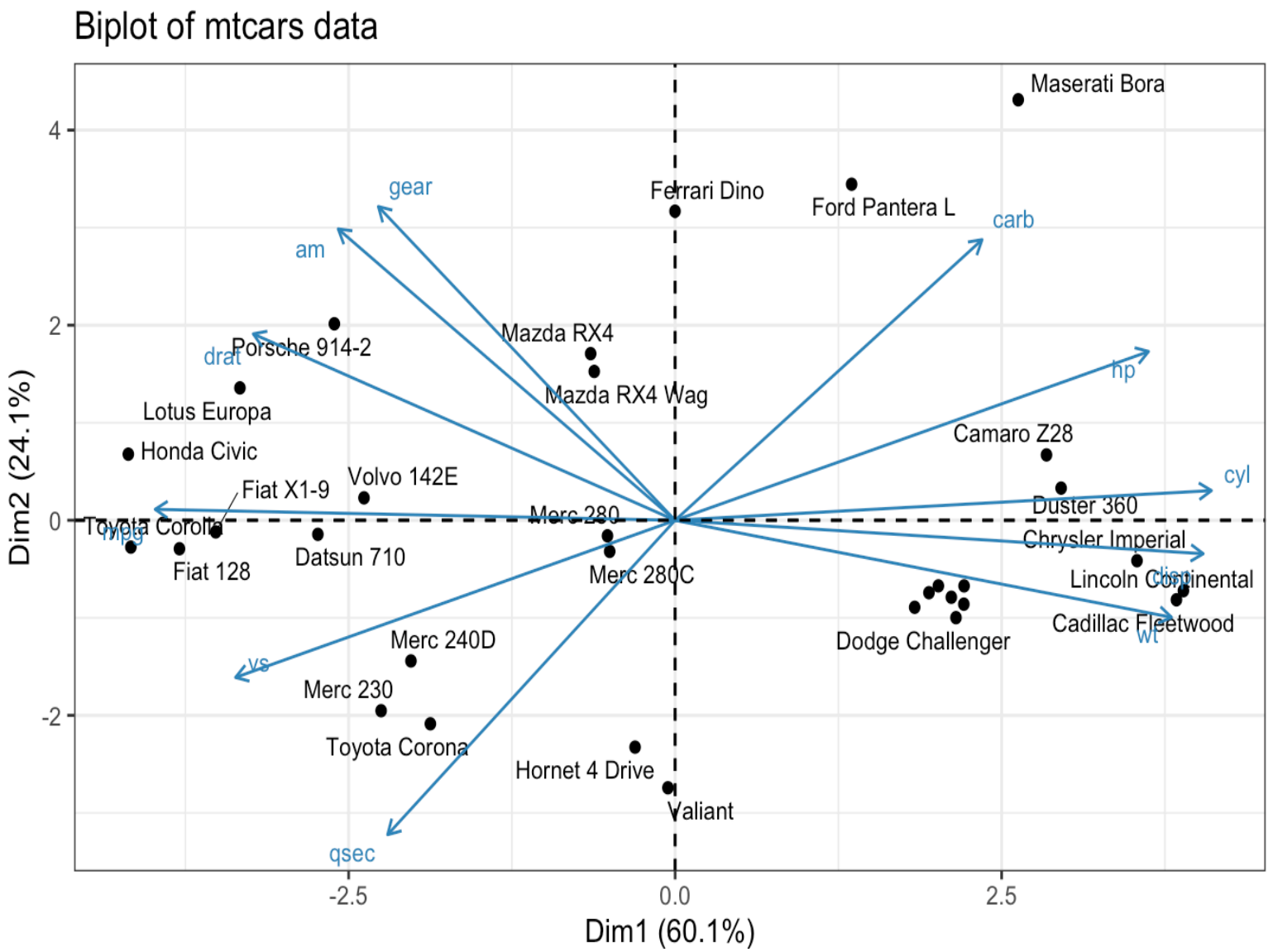
Example

```r
# basic 3-D scatterplot
library(scatterplot3d)
with(mtcars, {
  s3d <- scatterplot3d(
    x = disp,
    y = wt,
    z = mpg,
    color = "blue",
    pch = 19,
    type = "h",
    main = "3-D Scatterplot Example 3",
    xlab = "Displacement (cu. in.)",
    ylab = "Weight (lb/1000)",
    zlab = "Miles/(US) Gallon")

  # convert 3-D coords to 2D projection
  s3d.coords <- s3d$xyz.convert(disp, wt, mpg)

  # plot text with 50% shrink and place to right of points
  text(s3d.coords$x,
       s3d.coords$y,
       labels = row.names(mtcars),
       cex = .5,
       pos = 4)
})
```



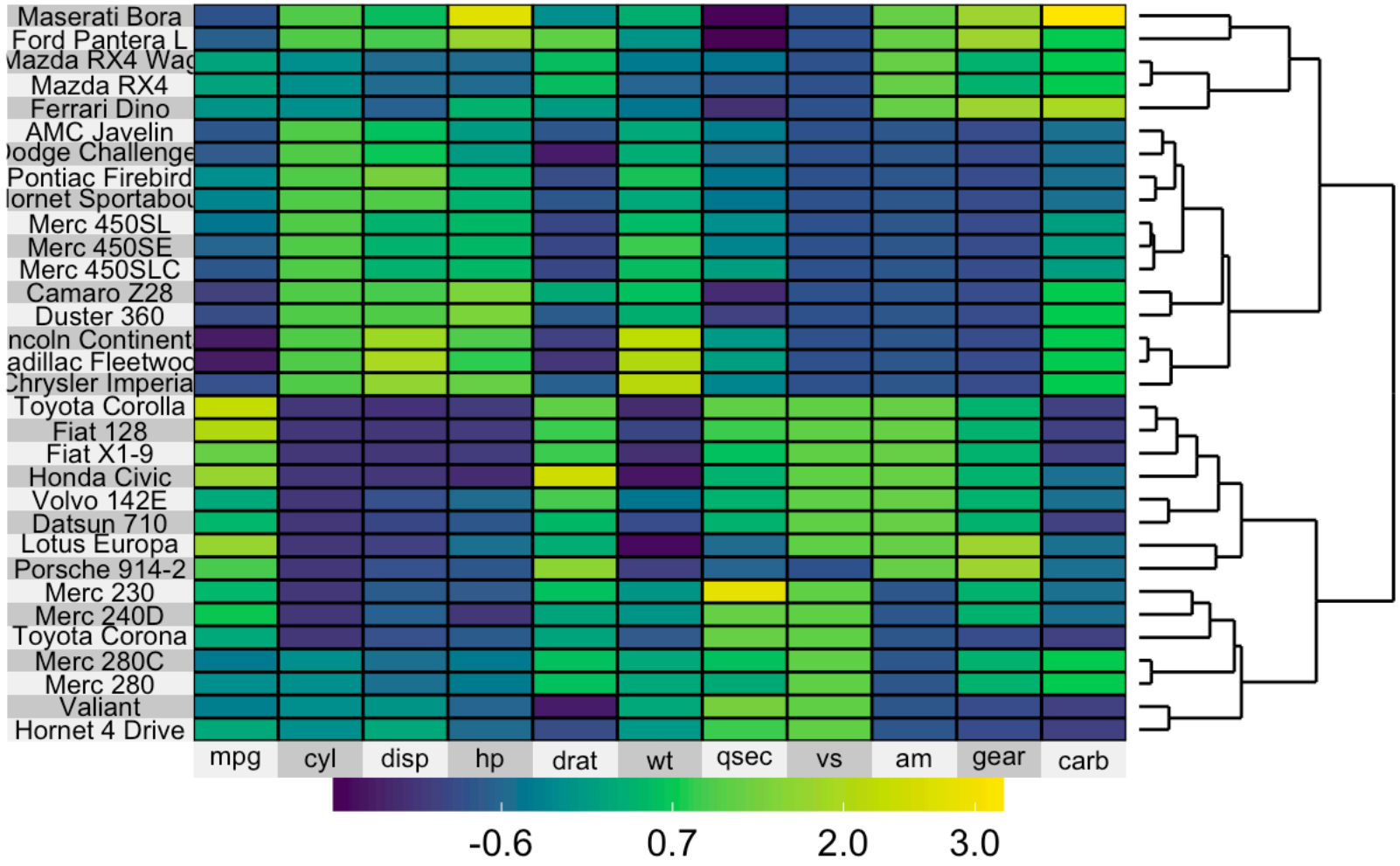
Example

```r
# create a biplot
# load data
data(mtcars)

# fit a principal components model
fit <- prcomp(x = mtcars,
              center = TRUE,
              scale = TRUE)

# plot the results
library(factoextra)
fviz_pca(fit,
         repel = TRUE,
         labelsize = 3) +
  theme_bw() +
  labs(title = "Biplot of mtcars data")
```



Example

```
# create a heatmap
data(mtcars)
library(superheat)
superheat(mtcars,
          scale = TRUE,
          left.label.text.size=3,
          bottom.label.text.size=3,
          bottom.label.size = .05,
          row.dendrogram = TRUE )
```

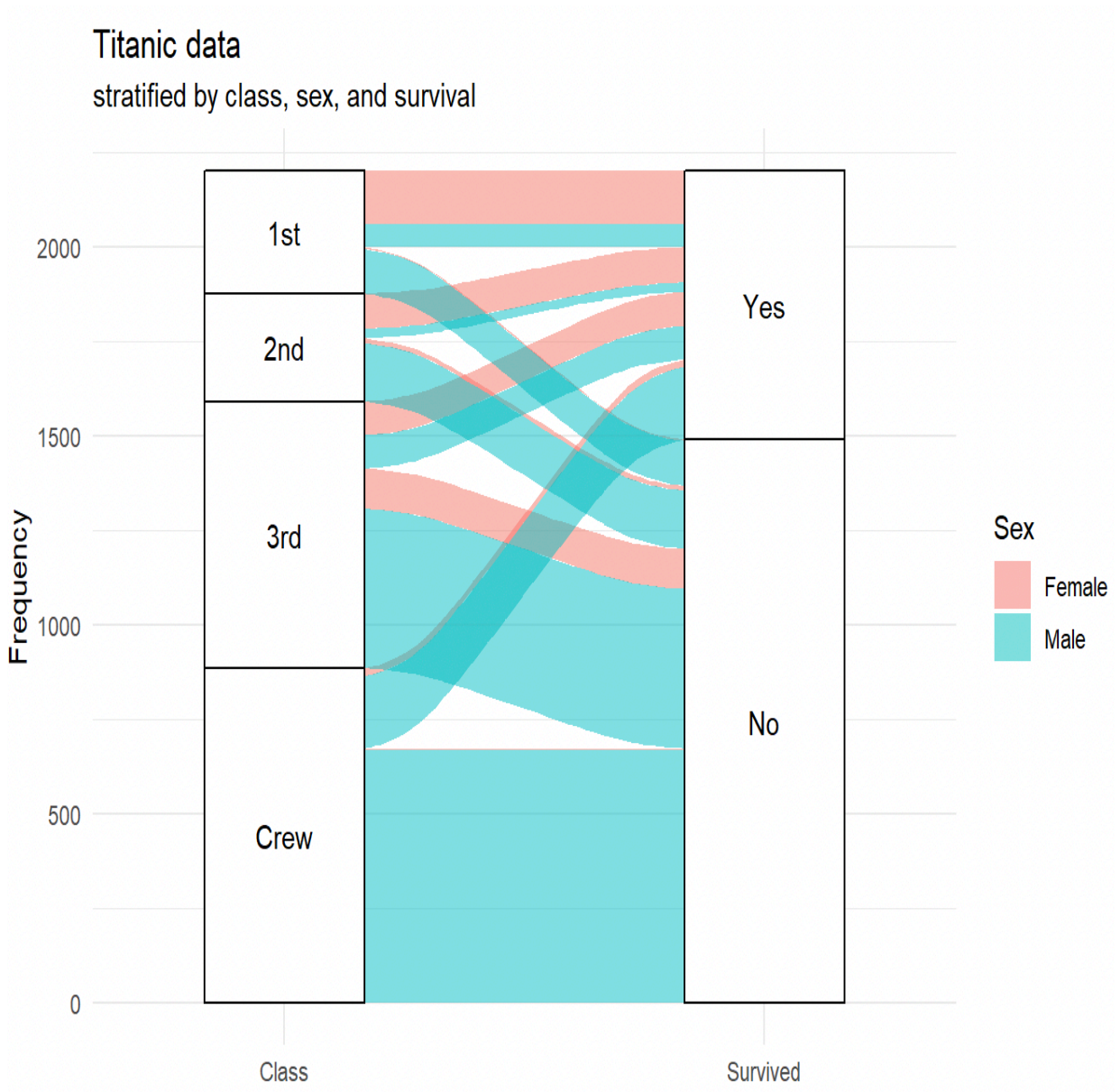


Example

```
#wordcloud
# install packages for text mining
install.packages(c("tm", "SnowballC",
                   "wordcloud", "RColorBrewer",
                   "RCurl", "XML"))
script <- "http://www.sthda.com/upload/rquery_wordcloud.r"
source(script)
res<-rquery.wordcloud("JFKspeech.txt",
                      type ="file",
                      lang = "english")
```

Example

```r
# create alluvial diagram
# input data
library(readr)
titanic <- read_csv("titanic.csv")

# summarize data
titanic_table <- titanic %>%
  group_by(Class, Sex, Survived) %>%
  count()

titanic_table$Survived <- factor(titanic_table$Survived,
                                 levels = c("Yes", "No"))

head(titanic_table)
library(ggplot2)
library(ggalluvial)

ggplot(titanic_table,
       aes(axis1 = Class,
           axis2 = Survived,
           y = n)) +
  geom_alluvium(aes(fill = Sex)) +
  geom_stratum() +
  geom_text(stat = "stratum",
            label.strata = TRUE) +
  scale_x_discrete(limits = c("Class", "Survived"),
                   expand = c(.1, .1)) +
  labs(title = "Titanic data",
       subtitle = "stratified by class, sex, and survival",
       y = "Frequency") +
  theme_minimal()
```



FIN

HANDS-ON EXERCISES

Sample Hands-on Exercise

Submit the code (name it as your LASTNAME_exercise3.R) in all 3 exercise to ronrick.da-ano@mcgill.ca with the subject EXERCISE 3.

Using the same code in slide 58, perform the following:

- It is best to avoid scientific notation in your graphs. How likely is it that the average reader will know that 3e+05 means 300,000,000? It is easy to change the scale in ggplot2. Simply divide the Thousands variable by 1000 and report it as Millions. While we are at it, let's

  - ❑ create black borders to highlight the difference between groups
  - ❑ reverse the order the groups to match increasing age
  - ❑ improve labelling
  - ❑ choose a different color scheme
  - ❑ choose a simpler theme.
  - ❑ The levels of the *AgeGroup* variable can be reversed using the fct_rev function in the forcats package i.e. forcats::fct_rev(AgeGroup)))

Sample Hands-on Exercise

Using the same code in slide 66, perform the following:

- Add Axis2 = Sex and Axis3 = Survived
- In the **geom_alluvium** change the fill as Class.
- Add **theme**(legend.position = "none") at the end.

Sample Hands-on Exercise

```
1  # create heatmap for gapminder data (Asia)
2  library(tidyr)
3  library(dplyr)
4
5  # load the gapminder data
6
7
8  # subset Asian countries
9  asia <- gapminder %>%
10   # asian continent
11   filter(__) %>%
12   #select year, country lifeExp
13   select(__)
14
15 # convert to long to wide format
16 plotdata <- spread(asia, year, lifeExp)
17
18 # save country as row names
19 plotdata <- as.data.frame(__)
20 row.names(__) <- plotdata$__
21 plotdata$country <- NULL
22
23 # row order
24 sort.order <- order(__$"2007")
25
26 # color scheme
27 library(RColorBrewer)
28 colors <- rev(brewer.pal(5, "Blues"))
29
30
31 # create the heat map
32 superheat(__,
33           scale = FALSE,
34           left.label.text.size=3,
35           bottom.label.text.size=3,
36           bottom.label.size = .05,
37           heat.pal = colors,
38           order.rows = sort.__,
39           title = "Life Expectancy in Asia")
```