Data Visualization with ggplot2

DATA VISUALIZATION AND DASHBOARDS

ibes non-data ink. Design elements!

The plotting space you are using

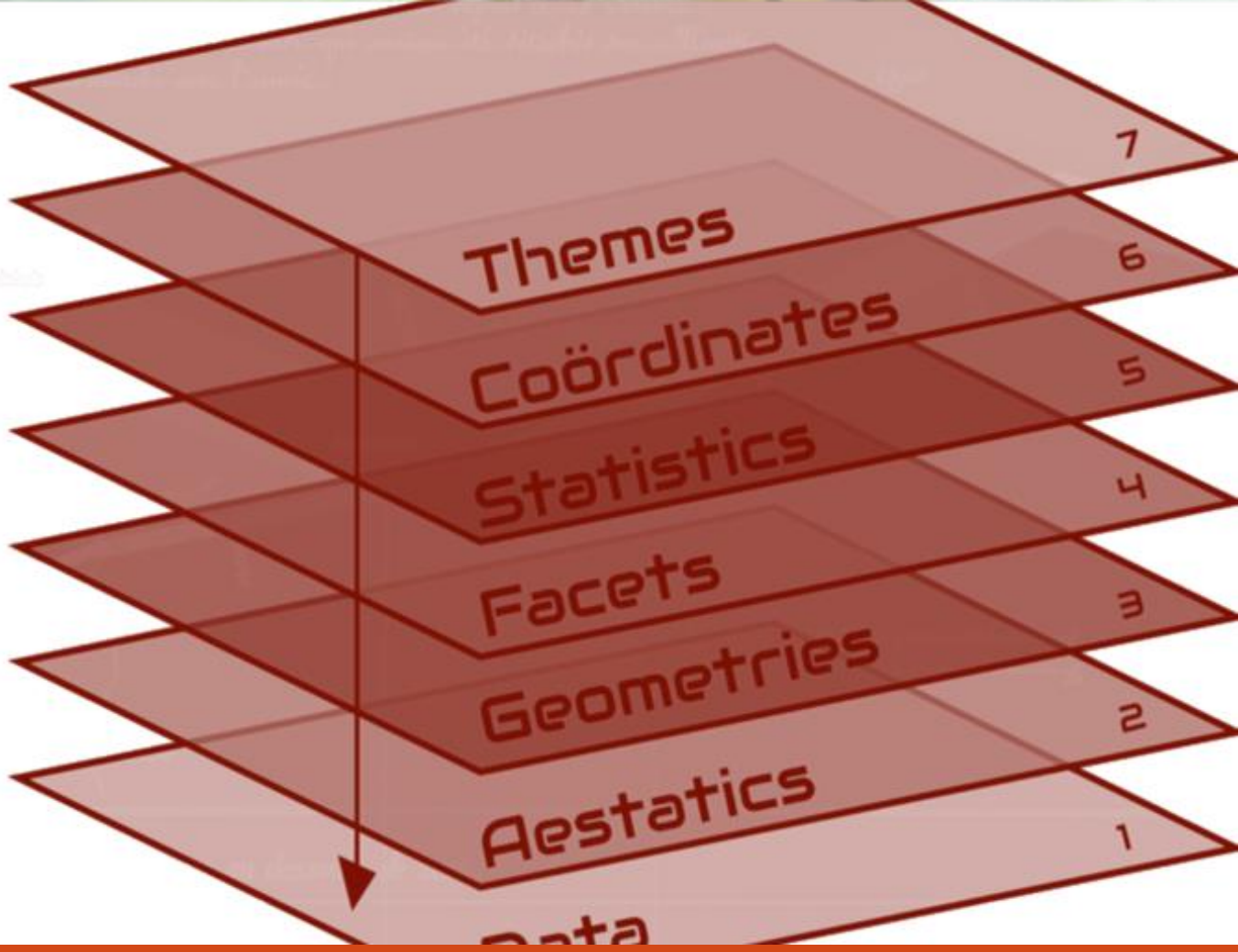
Statistical models & summaries

Rows and columns of sub-plots

hapes used to represent your data

scales on which the data is mapped

The actual variables to be plotted



10. The Grammar of Graphics

Grammar of Graphics

It is one thing to recognize when charts are **effective** and their aesthetics make them **easy to read**, and when they are laid out in a dashboard which tells a **compelling** visual story (and when they are not).

It is another thing altogether to learn how to **build** such charts.

The **grammar of graphics** [Wilkinson, 1999; Wickham, 2009] provides a reliable path to do so.

[D.J. Sarkar, A Comprehensive Guide to the Grammar of Graphics for Effective Visualization of Multi-dimensional Data]

Grammar of Graphics

"A **grammar** is defined as a set of structural rules which helps define and establish the **components of a language**.

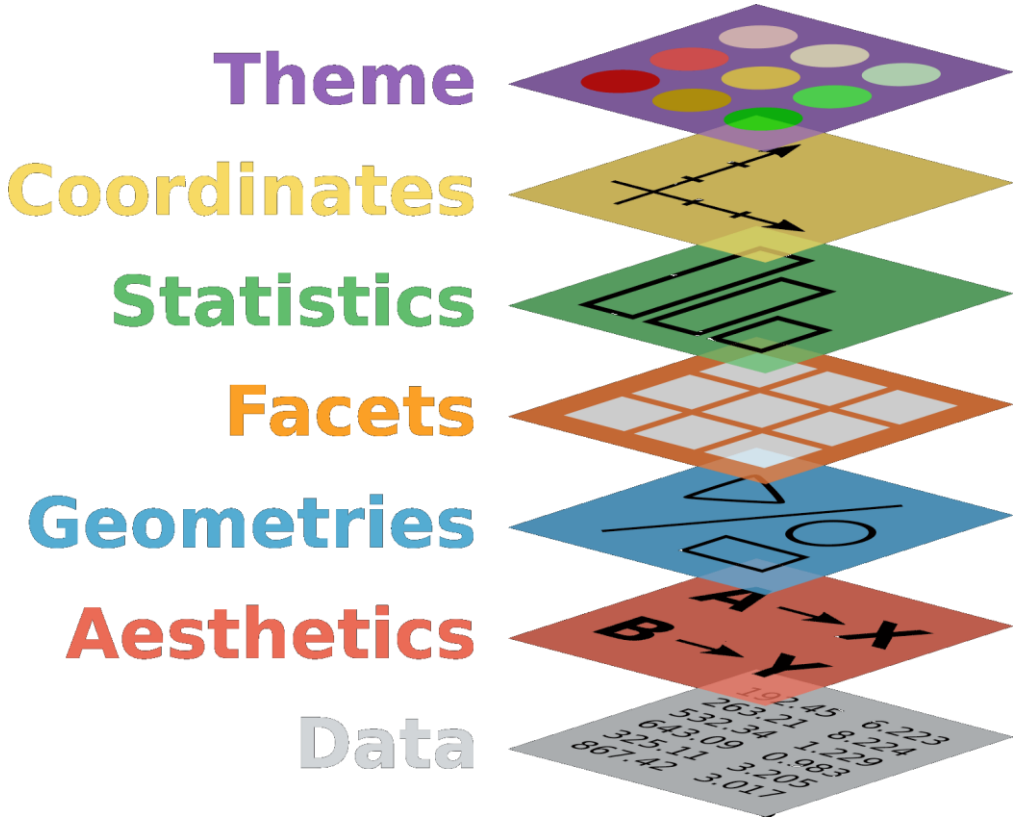
A language's system/structure usually consists of **syntax** and **semantics**.

A grammar of graphics is a framework which follows a **layered** approach to describe and construct visualizations or graphics in a **structured manner**.

The layered grammar of graphics uses pre-defined components to build charts (instead of random trials and errors)."

[Introduction to data visualisation with **ggplot2**, *QCBS R Workshop Series;* T. De Beus, Think About the Grammar of Graphics When Improving Your Graphs]

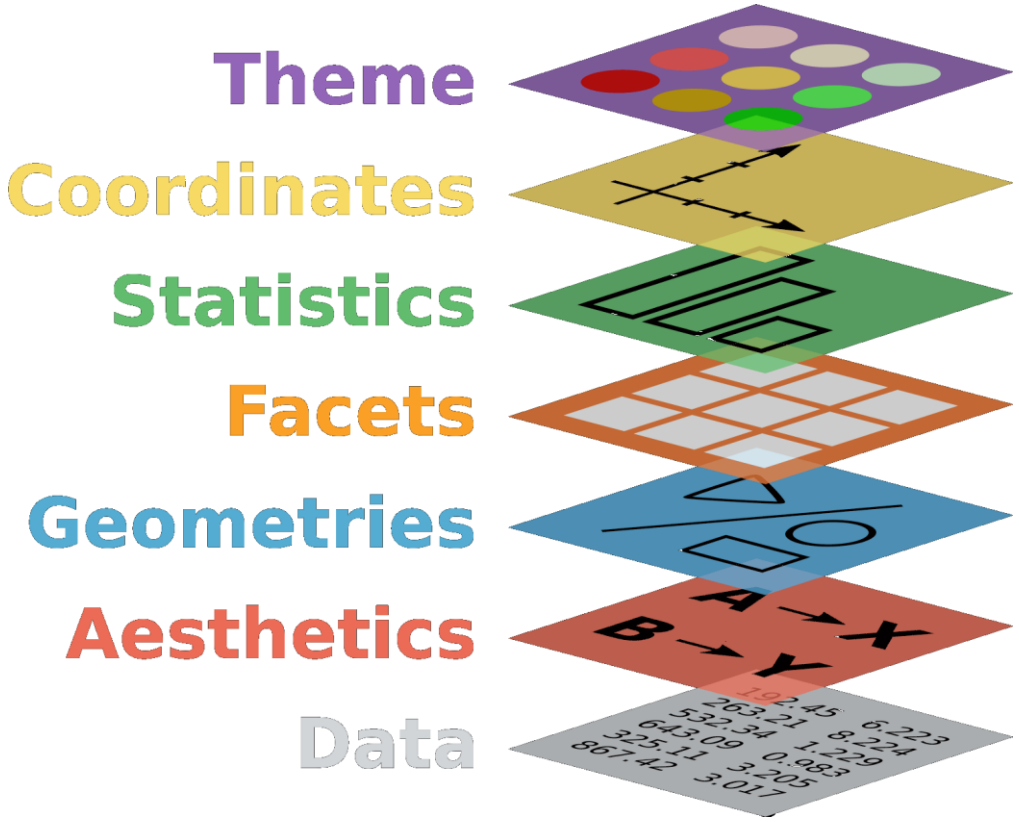
Grammar of Graphics Layers



1. **Data** (required): the plotting observations are found in rows, the variables in columns
2. **Aesthetics** (required): the mapping of the dataset's variables to the chart's scales (position, shape, size, colour, etc.)
3. **Geometry** (required): the type of chart on which the data is represented (bars, lines, points, etc.)
4. **Facets** (optional): the subsets of the data represented on the chart (levels)

[Introduction to data visualisation with **ggplot2**, *QCBS R Workshop Series;* T. De Beus, Think About the Grammar of Graphics When Improving Your Graphs]

Grammar of Graphics Layers



5. **Statistics** (optional): the measures that could provide context to the chart (centrality, dispersion, trend, etc.)

6. **Coordinates** (required): the chart plotting space (axes, scale, etc.)

7. **Themes** (required): the design choices that are used to create a visual identity (fonts, colours, etc.)

Examples – Gapminder Dataset

The Gapminder dataset (https://gapminder.org) contains socio-demographic information (upwards of 500 variables) for the Earth's nations, for years ranging from 1800 to 2020.

We deconstruct 8 charts built from this dataset using the layered grammar of graphics framework.

Data: Gapminder countries, 2009

Geometry: stacked density chart

Aesthetics:

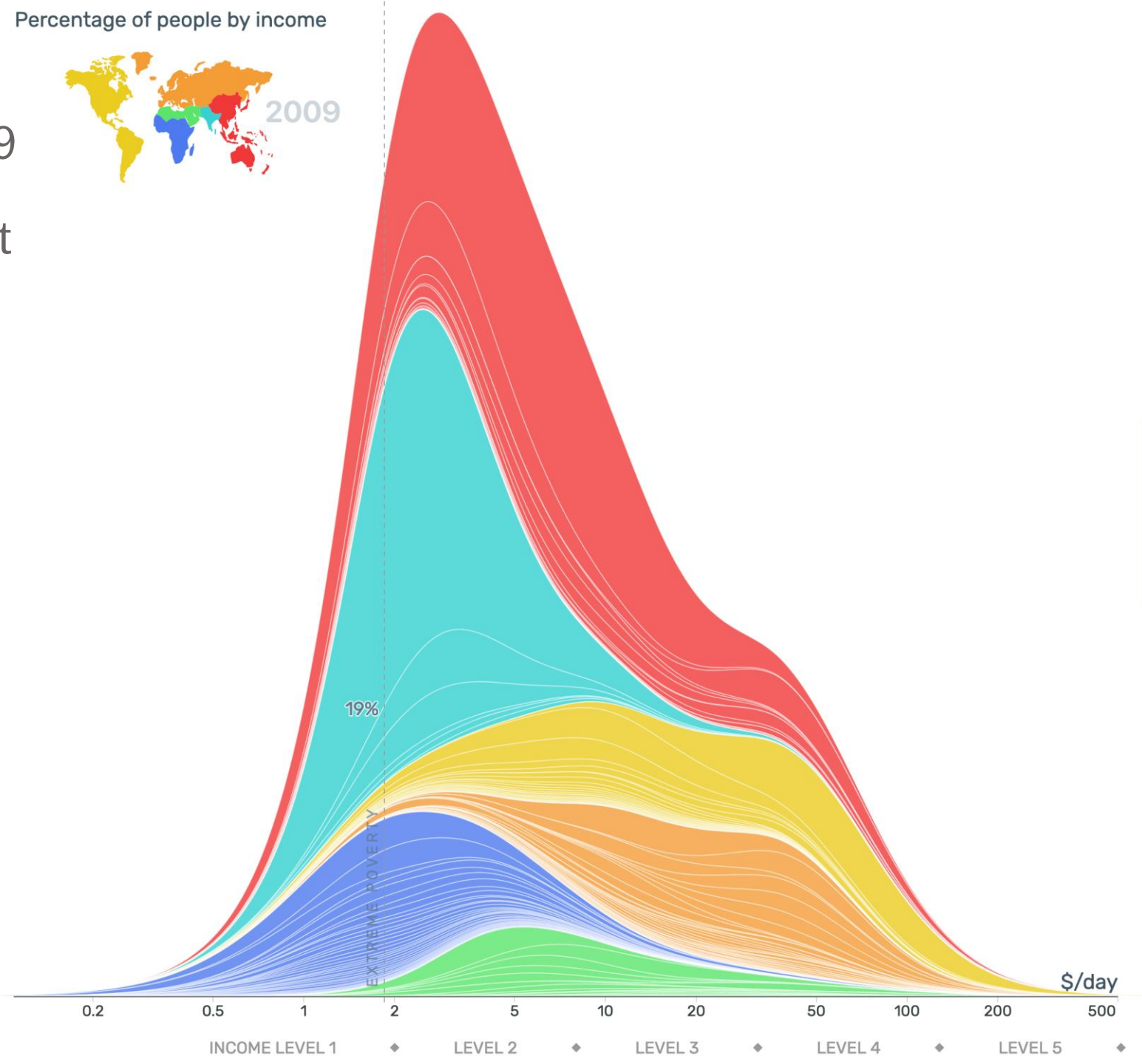
- x: daily income
- (y: percentage per country)
- fill: region

Facets: none

Statistics: extreme poverty prop

Coordinates: logarithmic (x)

Theme: Gapminder Tools; adornment (Extreme Poverty)



Data: Gapminder countries, 2009

Geometry: bubble chart

Aesthetics:

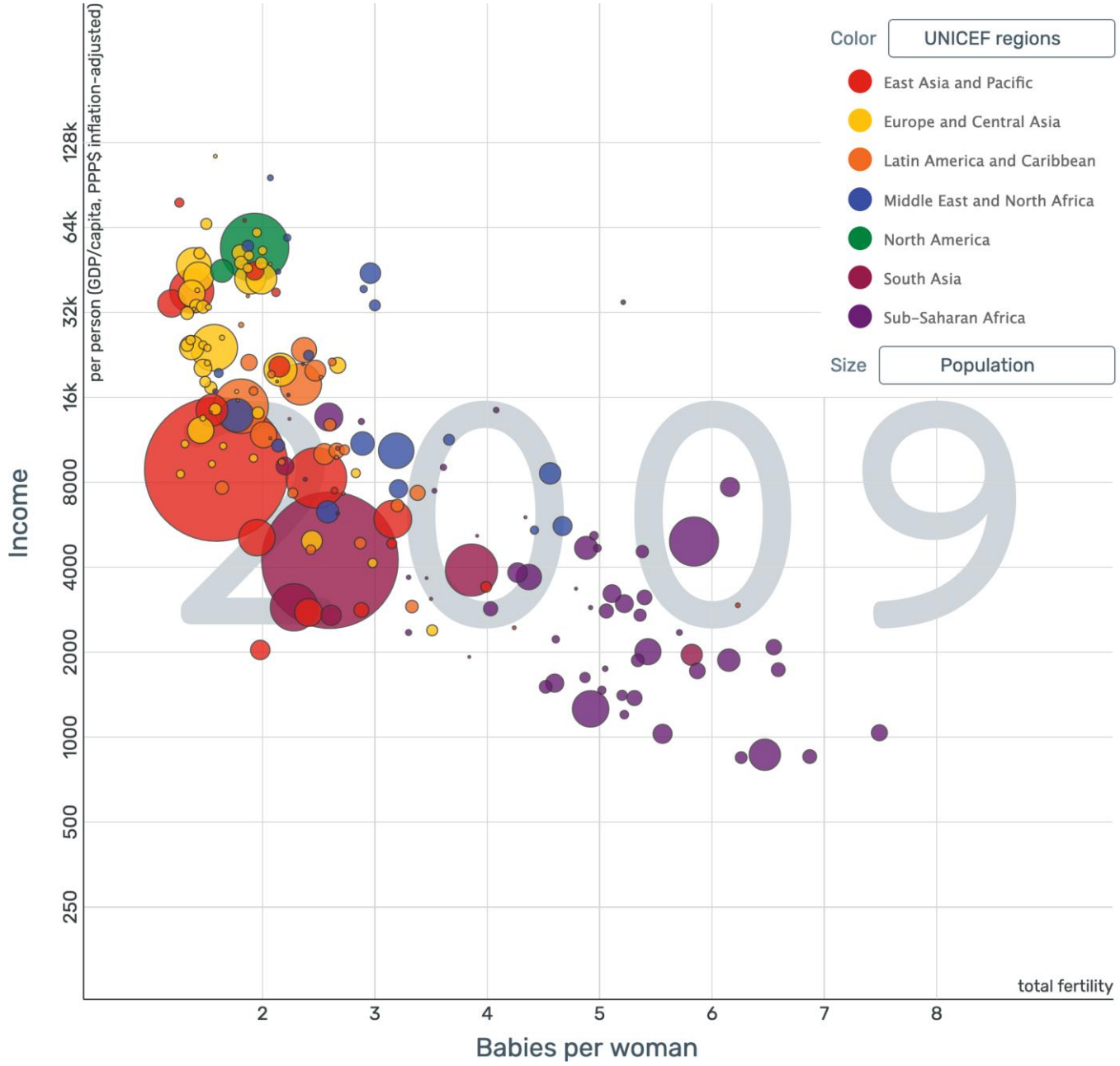
- x: total fertility
- y: income per person
- fill: UNICEF region
- size: population

Facets: none

Statistics: none

Coordinates: logarithmic (x, y, size)

Theme: Gapminder Tools



Data: Gapminder countries, 2009

Geometry: scatterplot chart

Aesthetics:

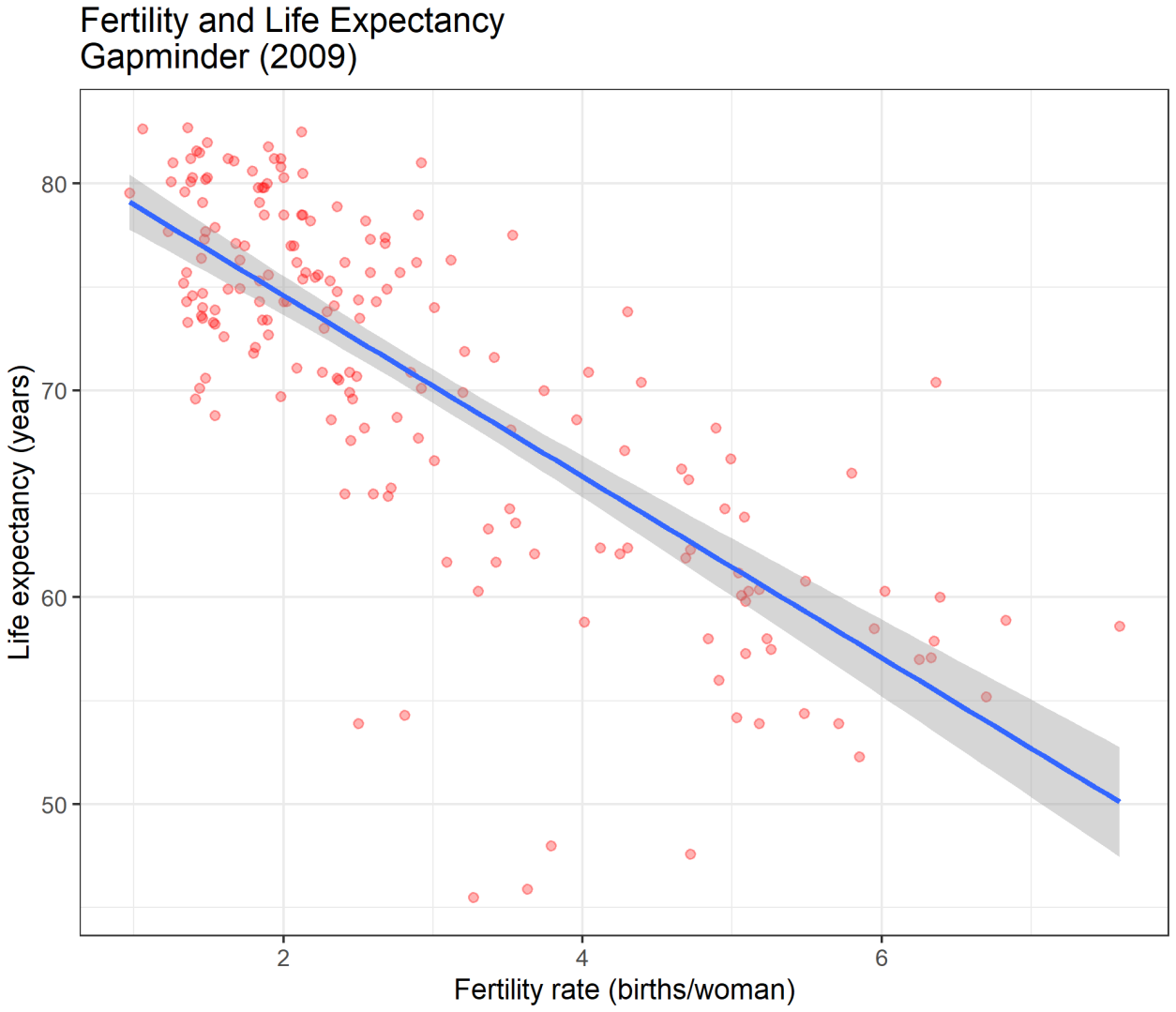
- x: total fertility
- y: life expectancy

Facets: none

Statistics: line of best fit, confidence interval

Coordinates: linear (x, y)

Theme: `ggplot2` default



Session 4

Data: Gapminder countries, 2009

Geometry: density chart

Aesthetics:

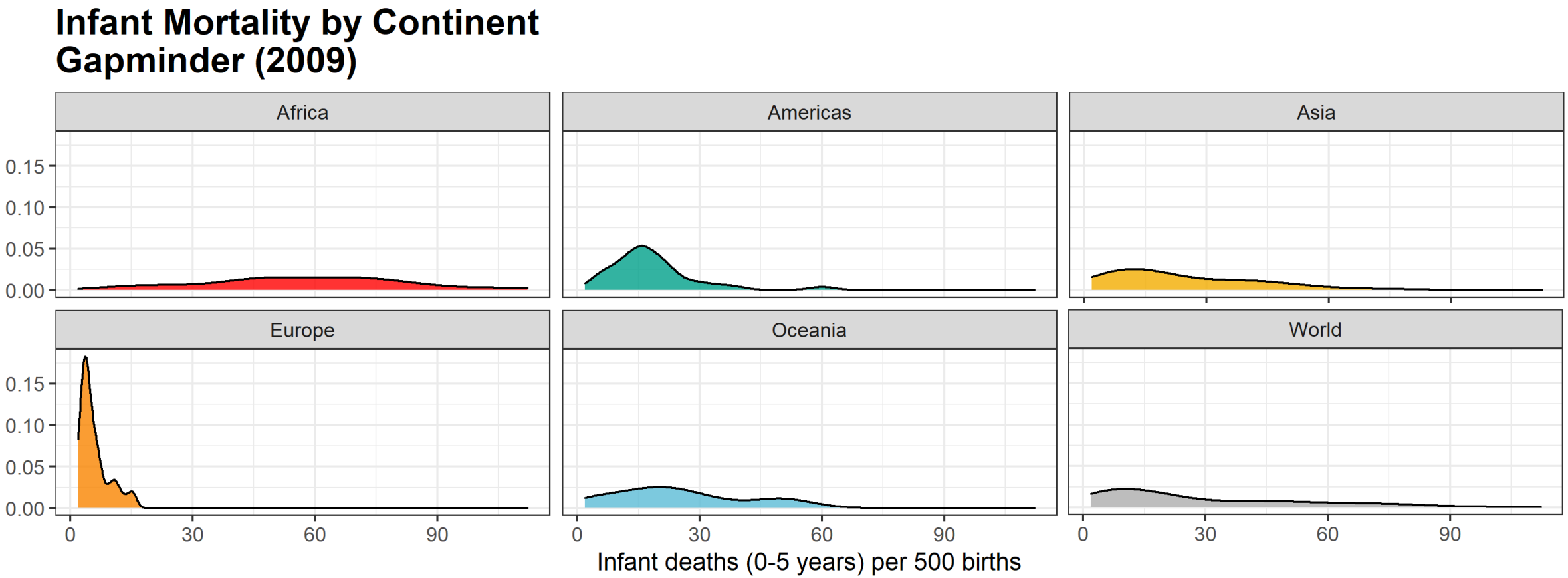
- x: infant mortality
- fill: continent

Facets: continent

Statistics: none

Coordinates: linear (x)

Theme: Darjeeling1



Data: Gapminder countries, 2009

Geometry: bubble chart

Aesthetics:

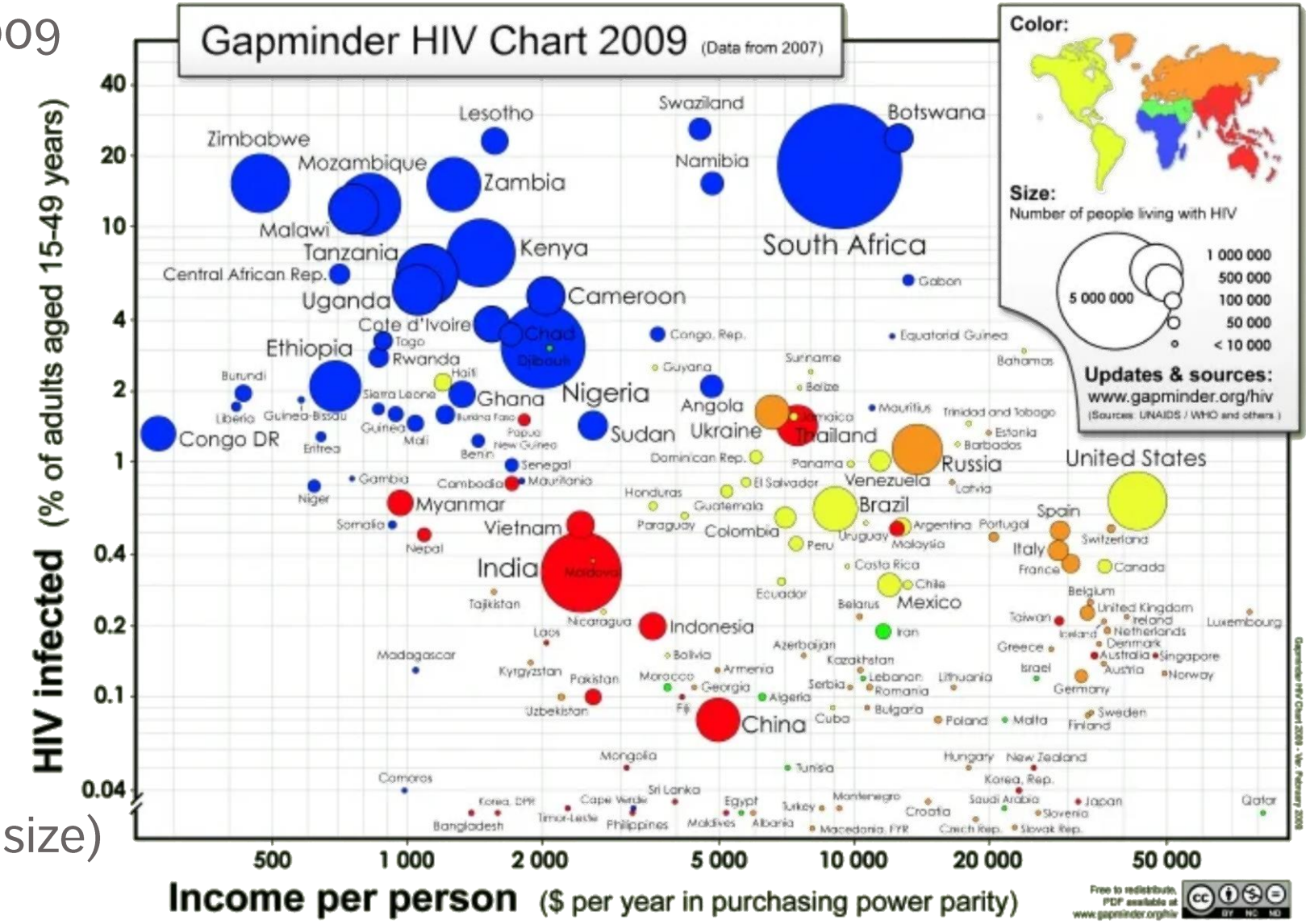
- x: income per person
- y: HIV infection rate
- fill: WHO region
- size: HIV infected population

Facets: none

Statistics: none

Coordinates: logarithmic (x, y, size)

Theme: old Gapminder World



Data: Gapminder countries, 2009

Geometry: boxplot chart

Aesthetics:

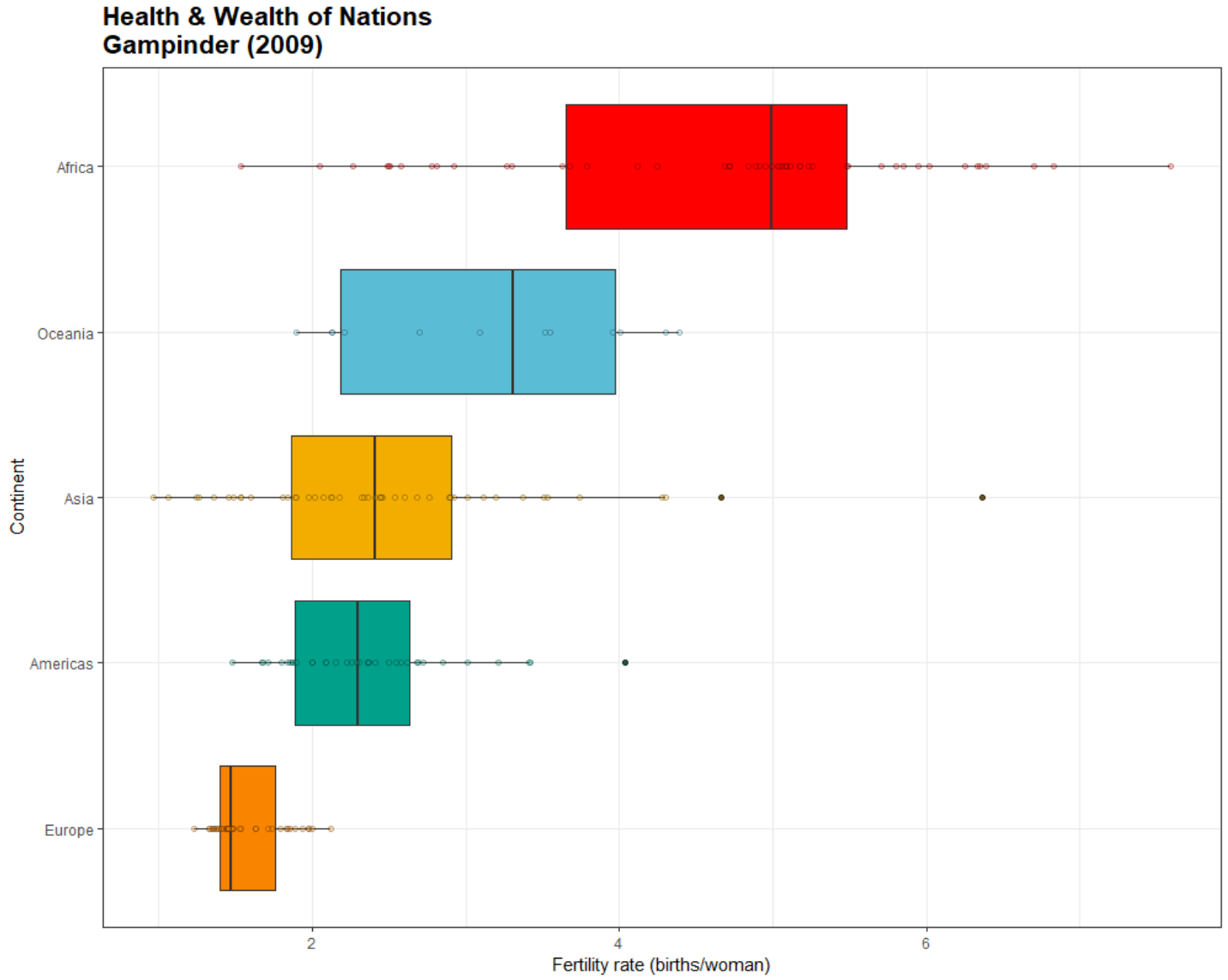
- x: fertility rate
- y: continent
- fill: continent

"Facets": continent

Statistics: 5-pt summary

Coordinates: linear (x)

Theme: Darjeeling1



Data: selected Gapminder countries, 1960-2011

Geometry: line chart

Aesthetics:

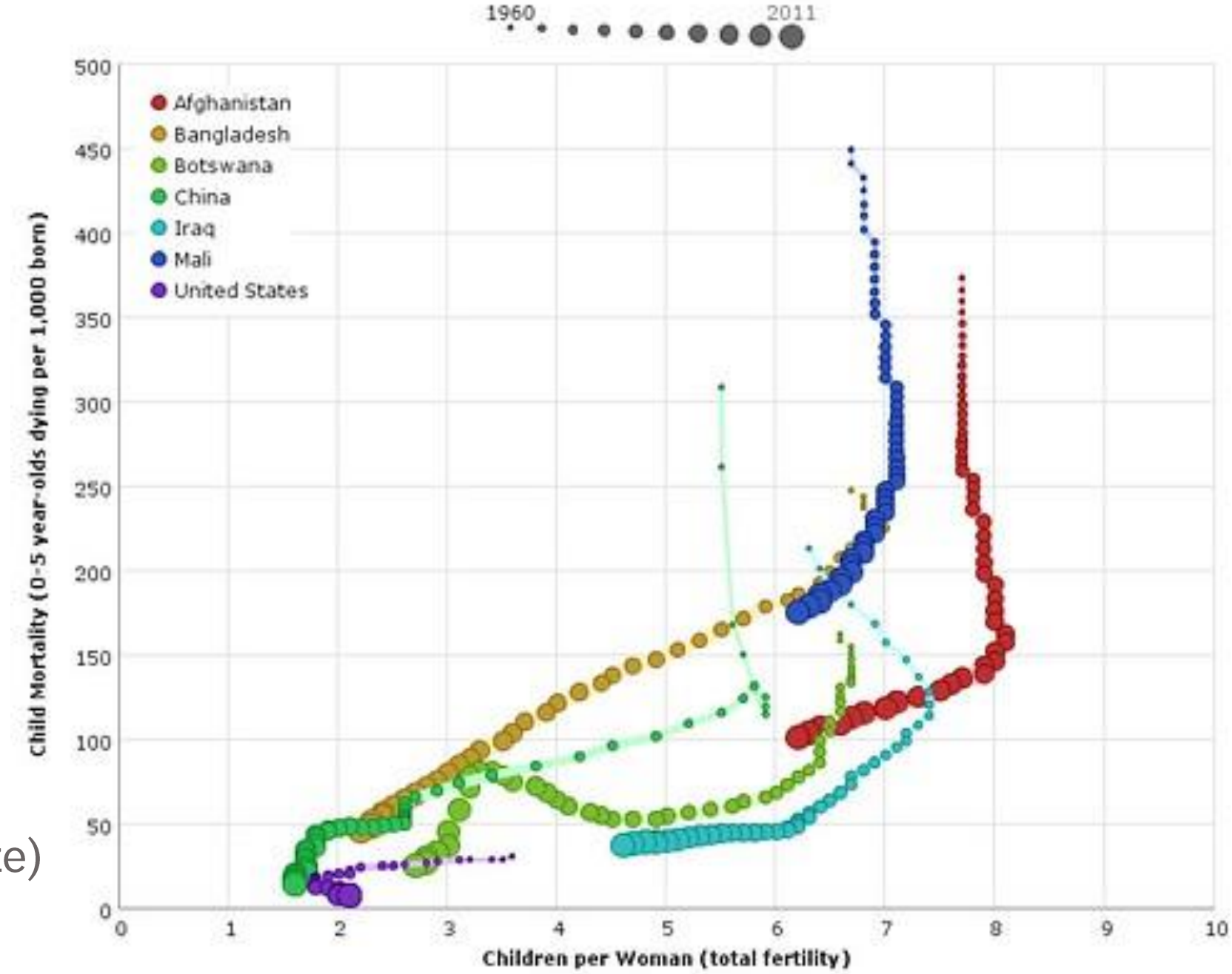
- x: total fertility
- y: percentage per country
- colour: country
- size: year

Facets: none

Statistics: none

Coordinates: linear (x, y, size)

Theme: custom



Data: Gapminder countries, 2012

Geometry: bubble chart

Aesthetics:

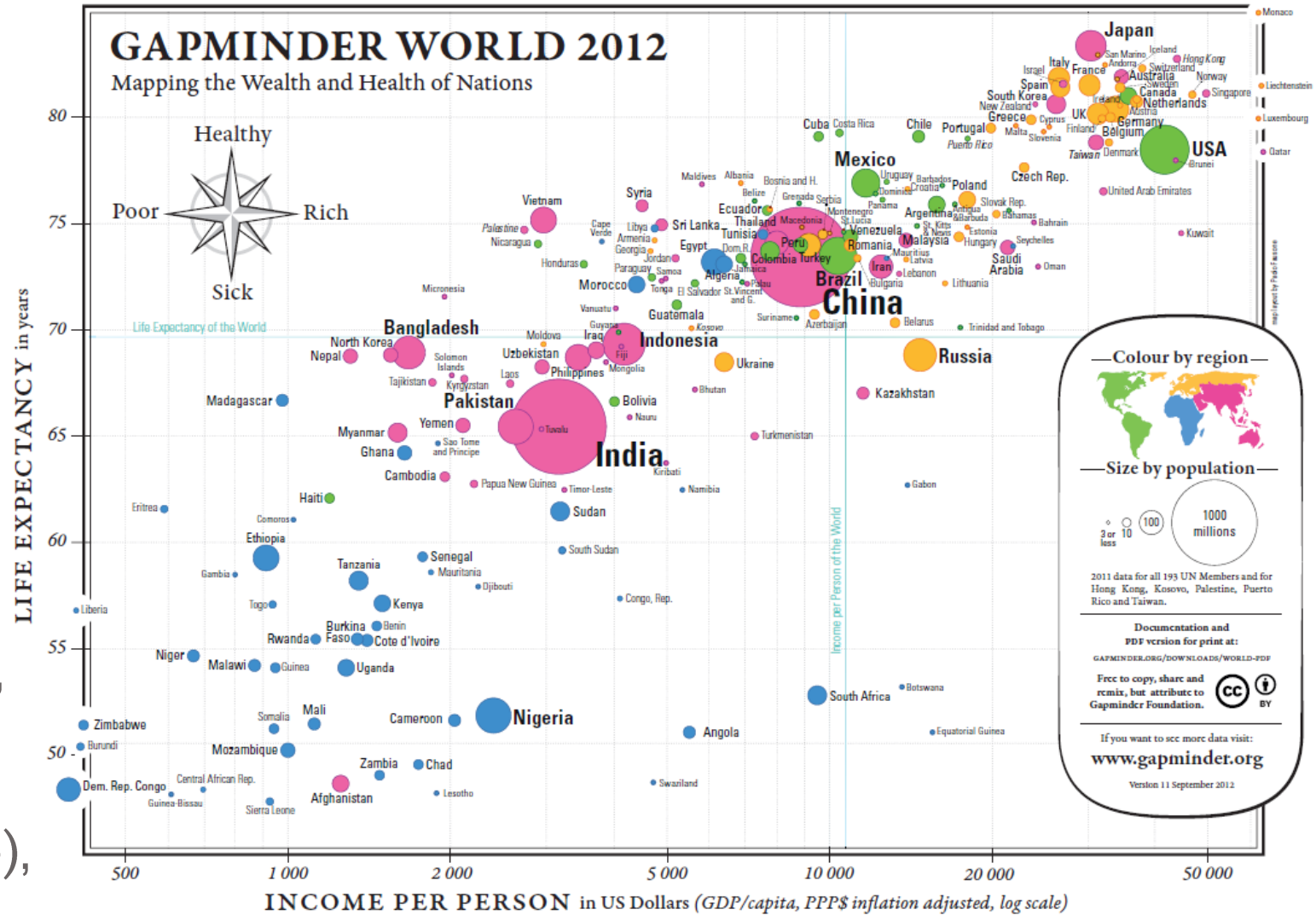
- x: income per person
- y: life expectancy
- fill: region
- size: population

Facets: none

Statistics: world life expectancy, world income per person

Coordinates: logarithmic (x, size), linear (y)

Theme: old Gapminder World



Suggested Reading

The Grammar of Graphics

The Practice of Data Visualization

Part II: The Foundations of Visual Design

5. Visual Design and Data Charts

5.4 The Grammar of Graphics

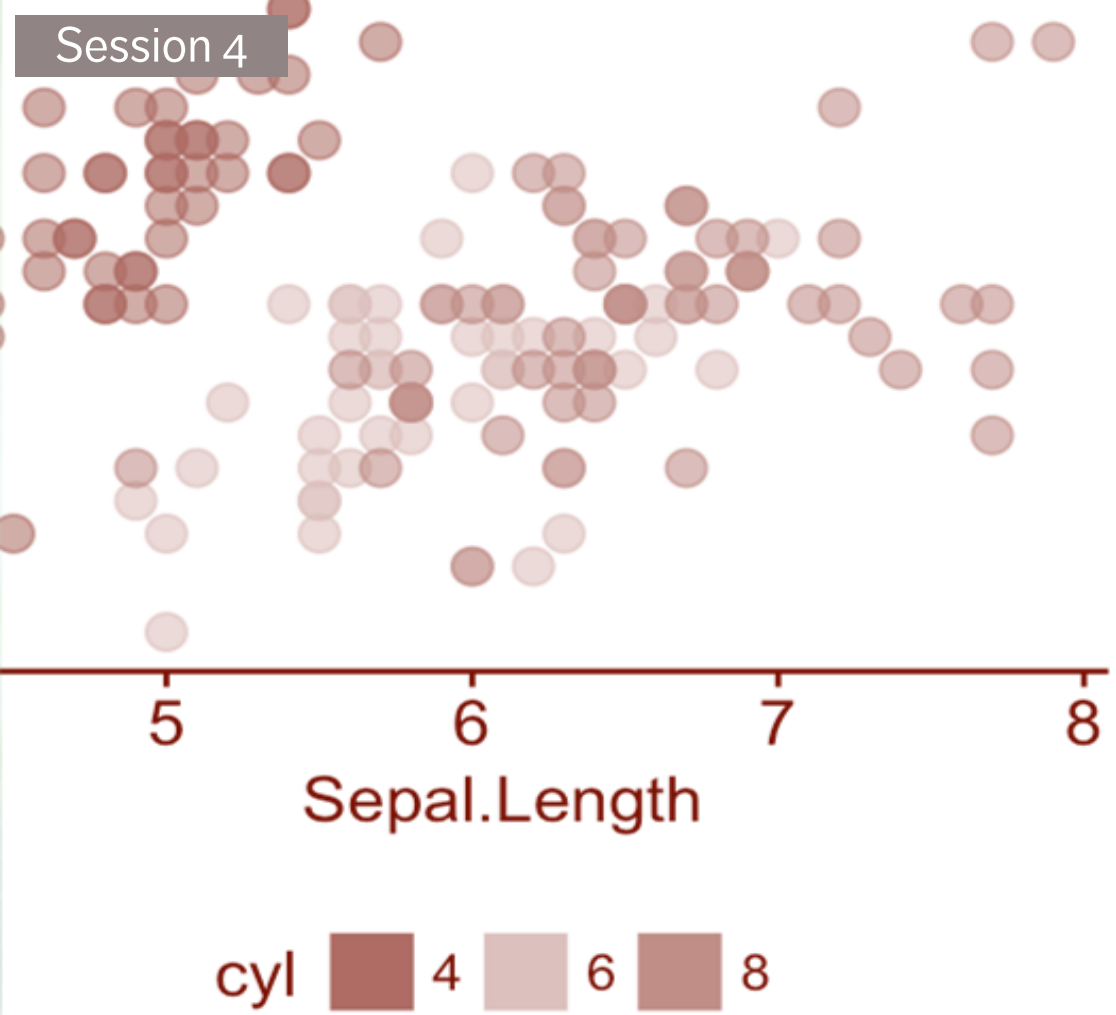
Exercises

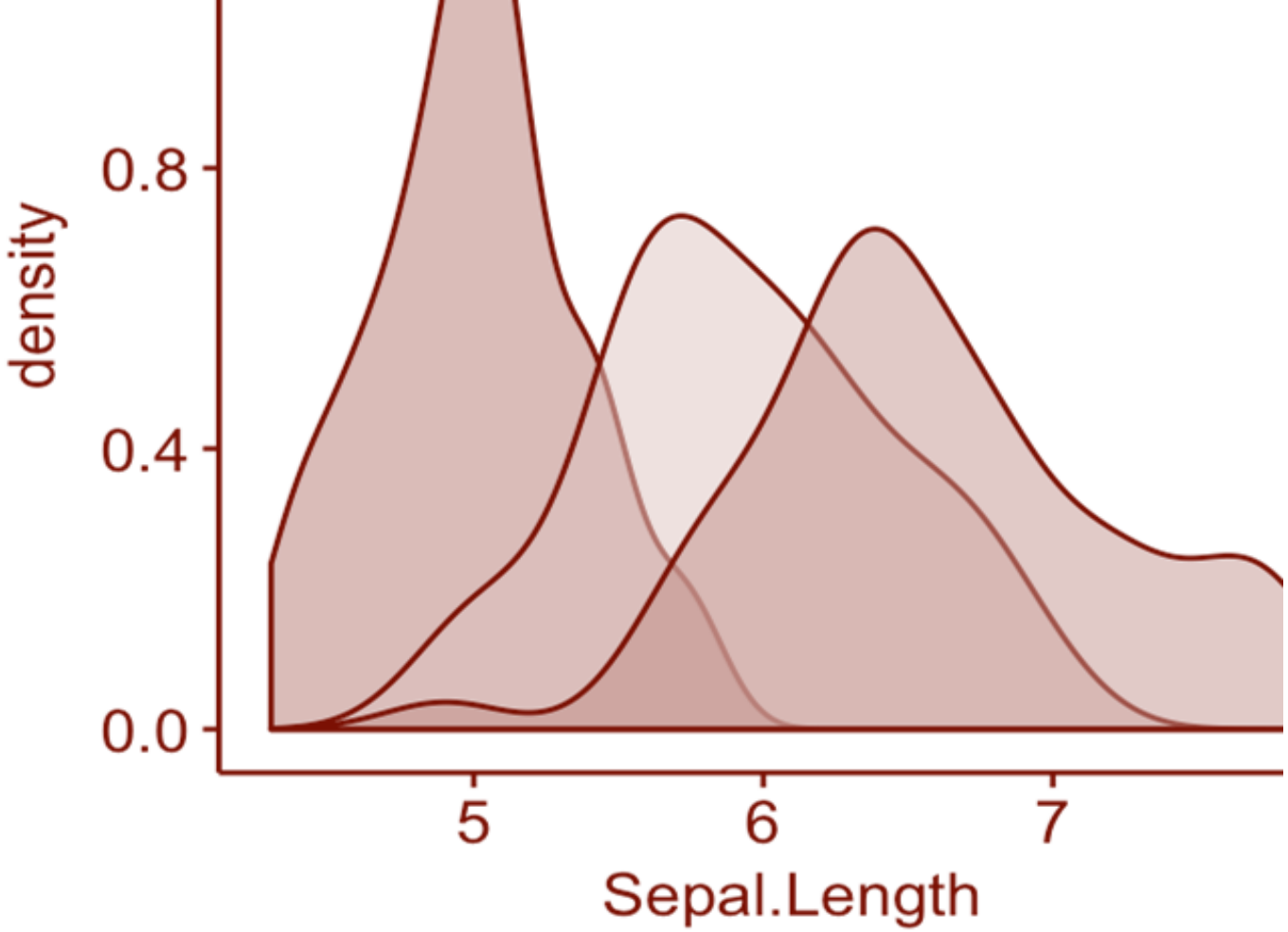
The Grammar of Graphics

Deconstruct the charts introduced in the first 9 modules in terms of the grammar of graphics.

What do some of the most effective charts have in common? What about the least effective ones?

Does that suggest a strategy?





11. Basics of ggplot2

A ggplot2 Primer

The layered grammar of graphics is implemented in **ggplot2**, a set of tools that map data to visual display elements, and that allow users to control the fine details of the plot display.

Most important aspect: **ggplot2** can be used to think about the **logical structure** of the plot.

A **ggplot2** graph has 2 main components (and optional terms):

- **aesthetic mappings** (**aes** – connections between data and plot elems.)
- **plot geometry** (**geom** – specifies the type of plot)
- *facets, *coordinates, *scales, *labels, *guides, etc.

ggplot2 Grammar

1. Tidy Data

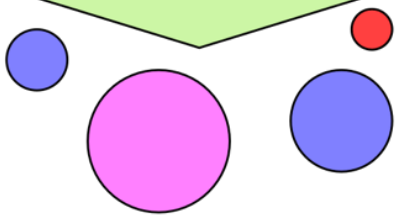
```
p ← ggplot(data = gapminder, ...
```

gdp	lifexp	pop	continent
340	65	31	Euro
227	51	200	Amer
909	81	80	Euro
126	40	20	Asia

2. Mapping

```
p ← ggplot(data = gapminder, mapping =
      aes(x = gdp, y = lifexp, size = pop,
          color = continent))
```

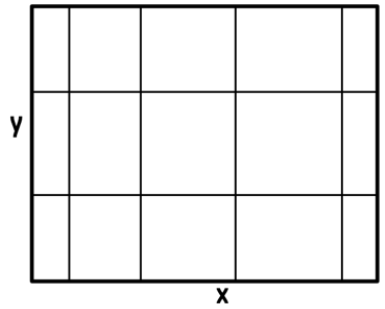
3. Geom



```
p + geom_point()
```

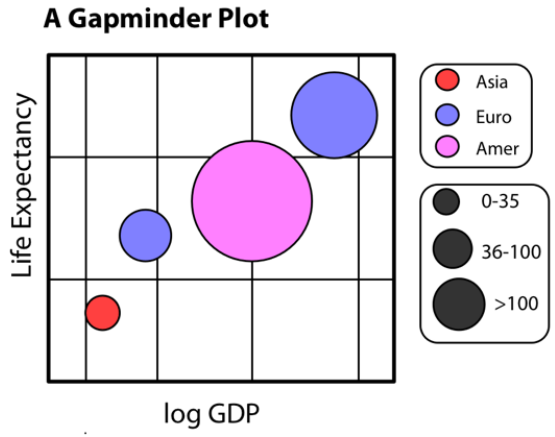
4. Co-Ordinates & Scales

```
p + coord_cartesian() + scale_x_log10()
```



5. Labels & Guides

```
p + labs(x = "log GDP", y = "Life
Expectancy", title = "A Gapminder Plot")
```



[Healey, K., *Data Visualization: A Practical Introduction*]

`ggplot2` Grammar – Geometries

The data source and variables to be plotted are specified via `ggplot()`.

The various `geom()` functions specify **how** these variables are to be visually represented:

- using points, bars, lines, shaded regions, etc.

There are currently 37+ available geometries.

Function	Adds	Options
`geom_bar()`	bar chart	color, fill, alpha
`geom_boxplot()`	boxplot	color, fill, alpha, notch, width
`geom_density()`	density plot	color, fill, alpha, linetype
`geom_histogram()`	histogram	color, fill, alpha, linetype, binwidth
`geom_hline()`	horizontal lines	color, alpha, linetype, size
`geom_line()`	jittered points	color, size, alpha, shape
`geom_jitter()`	line graph	color, alpha, linetype, size
`geom_point()`	scatterplot	color, alpha, shape, size
`geom_rug()`	rug plot	color, side
`geom_smooth()`	fitted line	method, formula, color, fill, linetype, size
`geom_text()`	text annotations	many; see the help for this function
`geom_violin()`	violin plot	color, fill, alpha, linetype
`geom_vline()`	vertical lines	color, alpha, linetype, size

Option	Specifies
color	colour of points, lines, and borders around filled regions
fill	colour of filled areas such as bars and density regions
alpha	transparency of colors, ranging from 0 (fully transparent) to 1 (opaque)
linetype	pattern for lines (1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash)
size	point size and line width
shape	point shapes (same as pch, with 0 = open square, 1 = open circle, 2 = open triangle, and so on)
position	position of plotted objects such as bars and points. For bars, `dodge'' places grouped bar charts side by side,` stacked" vertically stacks grouped bar charts, and `fill''` `vertically stacks grouped bar charts and standardizes their heights to be equal; for points,` jitter" reduces point overlap
binwidth	bin width for histograms
notch	indicates whether box plots should be notched (TRUE/FALSE)
sides	placement of rug plots on the graph (`b'' = bottom,` l" = left, `t'' = top,` r" = right, bl" = both bottom and left, and so on)
width	width of box plots

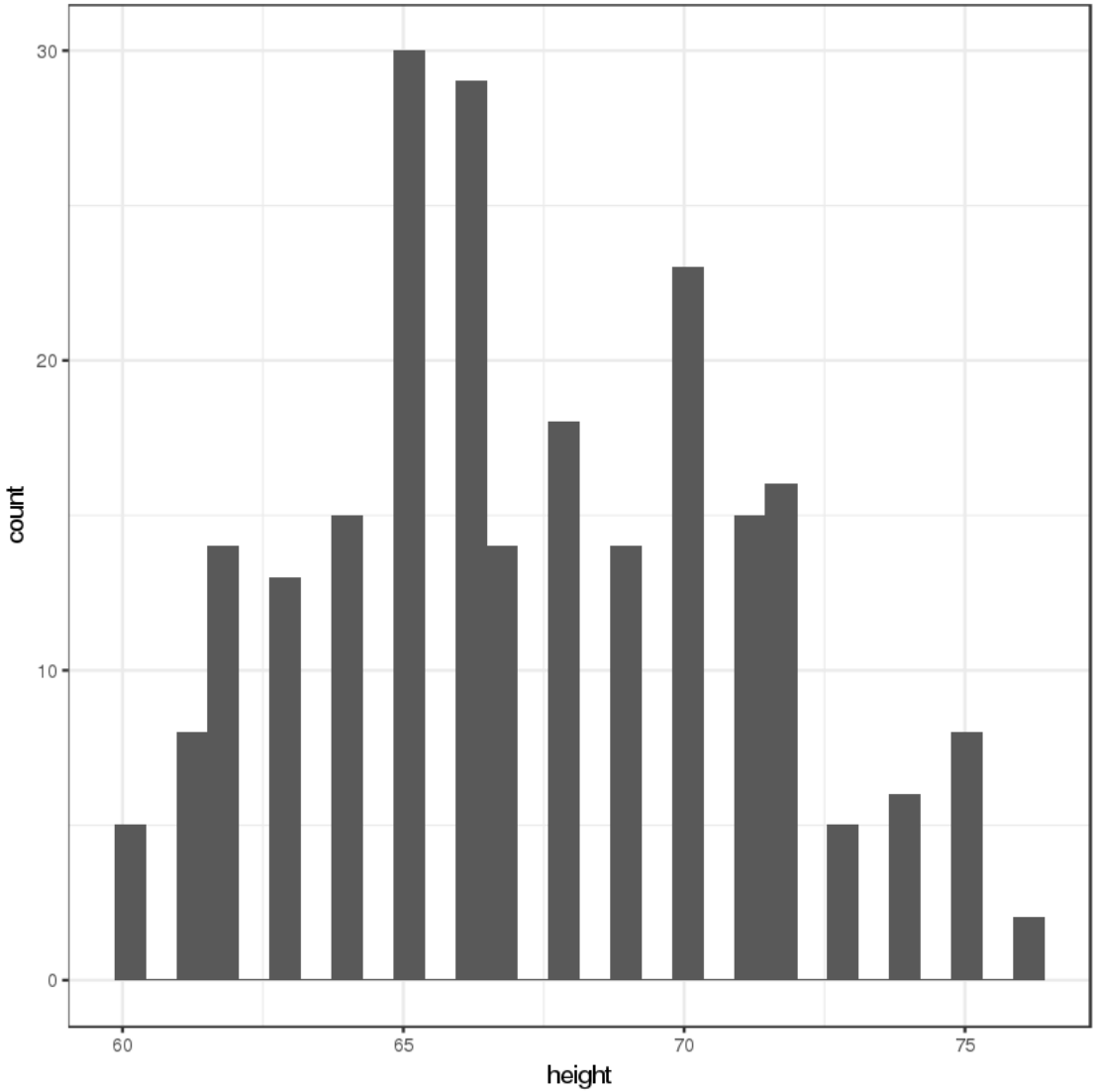
ggplot2 Grammar – geom()

```
library("ggplot2")
data(singer, package="lattice")
# Using data from the 1979 ed. of the
# New York Choral Society

# Histogram of heights
ggplot(singer, aes(x=height)) +
  geom_histogram()

# Boxplot of heights by voice part
ggplot(singer, aes(x=voice.part, y=height)) +
  geom_boxplot()
```

What do you expect the output to be?

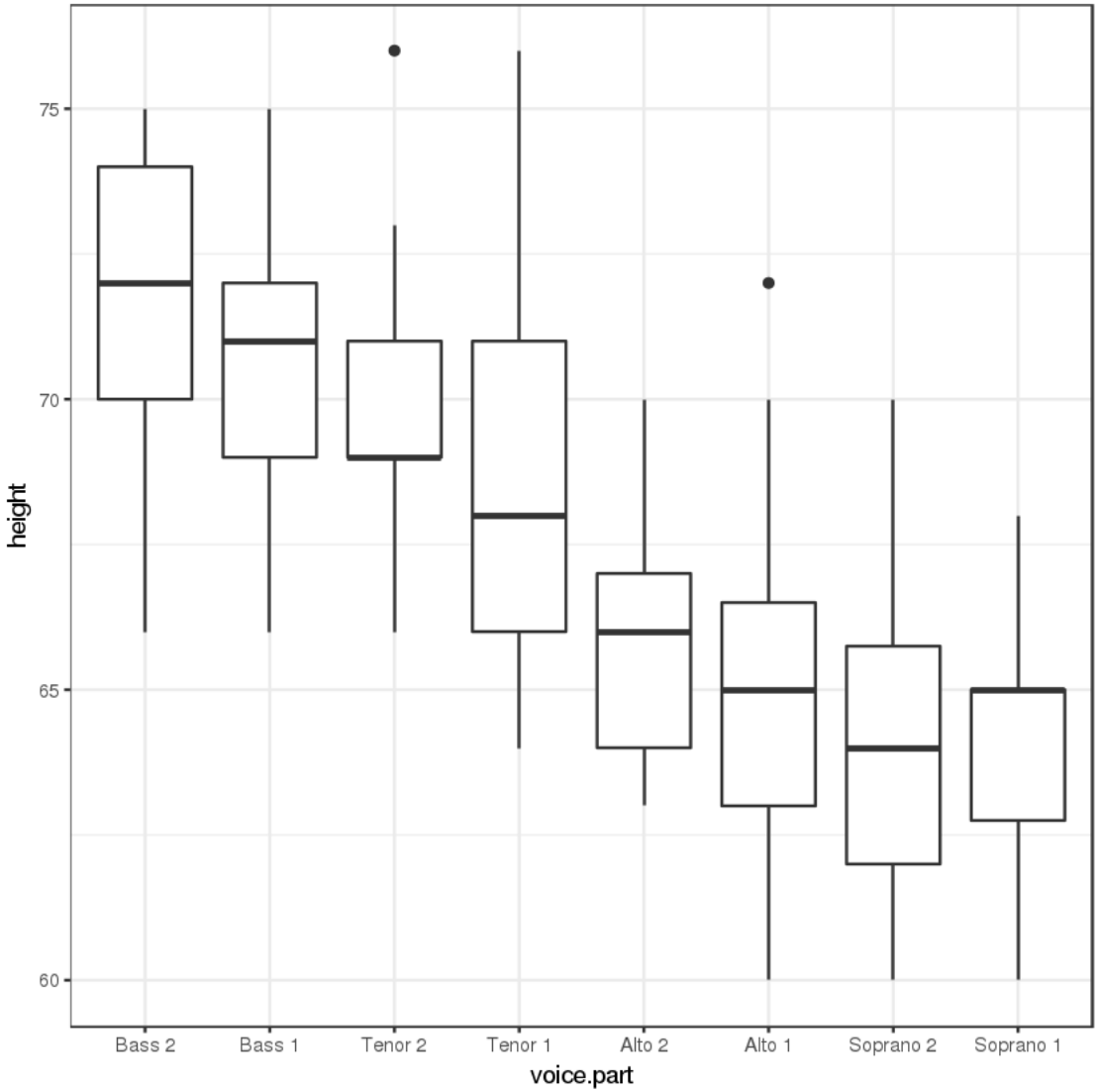
ggplot2 Grammar

```
library("ggplot2")
data(singer, package="lattice")
# Using data from the 1979 ed. of the
# New York Choral Society

# Histogram of heights
ggplot(singer, aes(x=height)) +
  geom_histogram()

# Boxplot of heights by voice part
ggplot(singer, aes(x=voice.part, y=height)) +
  geom_boxplot()
```



ggplot2 Grammar

```
library("ggplot2")
data(singer, package="lattice")
# Using data from the 1979 ed. of the
# New York Choral Society

# Histogram of heights
ggplot(singer, aes(x=height)) +
   geom_histogram()

# Boxplot of heights by voice part
ggplot(singer, aes(x=voice.part, y=height)) +
   geom_boxplot()
```



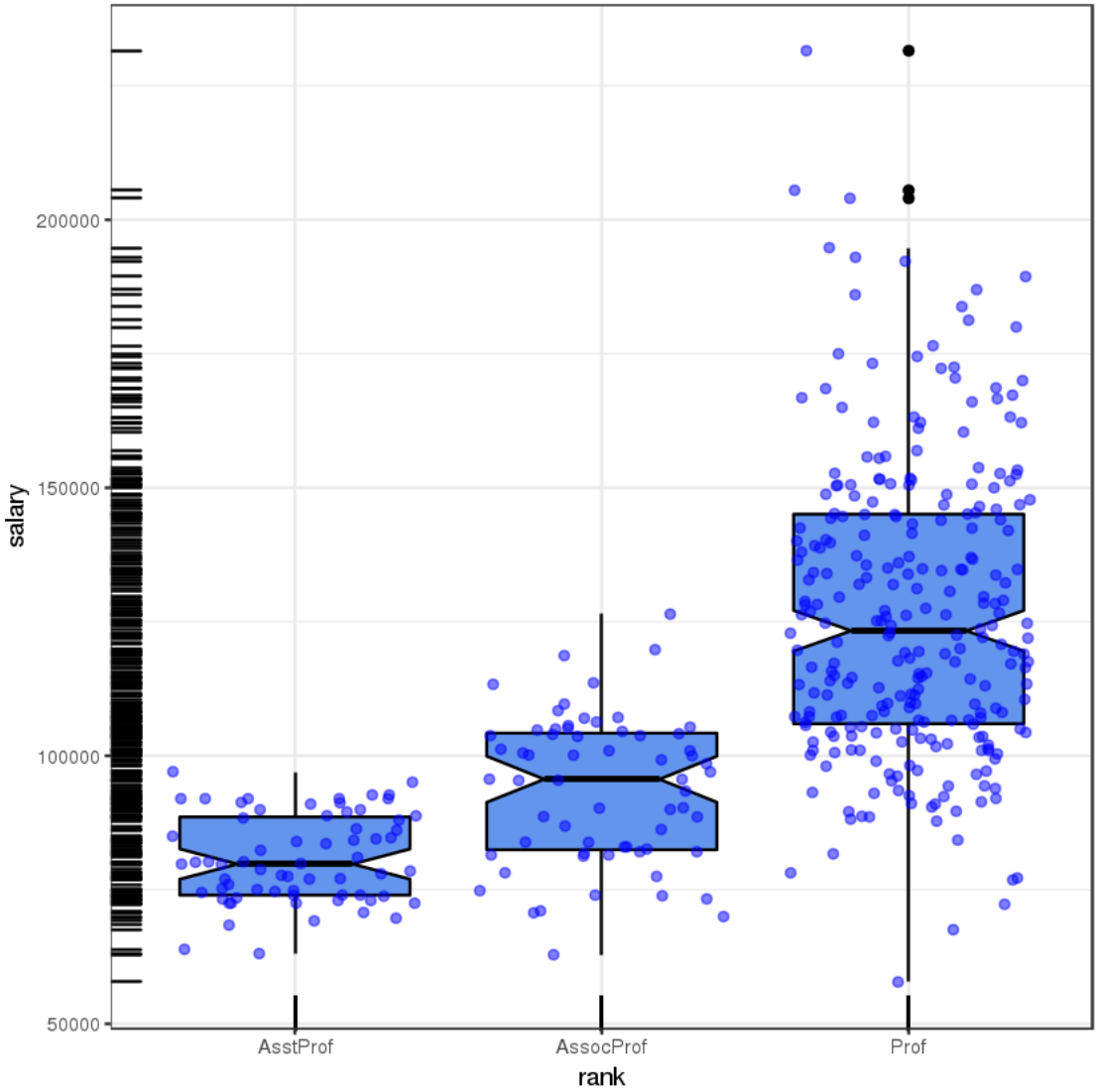
ggplot2 Grammar – geom()

```
library(ggplot2)
data(Salaries, package="car")
# Using data on salaries of a sample of
# US university professors (2018-2019)
# var: rank, sex, yrs.since.phd, yrs.service, salary

ggplot(Salaries, aes(x=rank, y=salary)) +
  geom_boxplot(fill="cornflowerblue",color="black", notch=TRUE) +
  geom_point(position="jitter", color="blue", alpha=.5) +
  geom_rug(side="l", color="black")
```

What do you expect the output to be?

ggplot2 Grammar

```
library(ggplot2)
data(Salaries, package="car")
# Using data on salaries of a sample of
# US university professors (2018-2019)
# var: rank, sex, yrs.since.phd, yrs.service, salary

ggplot(Salaries, aes(x=rank, y=salary)) +
  geom_boxplot(fill="cornflowerblue",color="black", notch=TRUE) +
  geom_point(position="jitter", color="blue", alpha=.5) +
  geom_rug(side="l", color="black")
```



`ggplot2` Grammar – Aesthetics

Aesthetics refer to the displayed attributes of the data.

They map the data to an attribute (such as the size or shape of a marker) and generate an appropriate legend.

Aesthetics are specified with the `aes()` function, either within the data call or within a `geom()` call. If they're specified within `ggplot()` then they apply to all specified geometries.

`ggplot2` Grammar – Aesthetics

The aesthetics available to `geom_point()` (scatterplot), as an example, are:

- `x` , `y` , `alpha` , `colour` , `fill` , `shape`, `size`

Important difference between specifying characteristics (like colour, shape) inside and outside the `aes()` call

- **inside:** assigned colour or shape automatically based on the data
- **outside**: not mapped to data

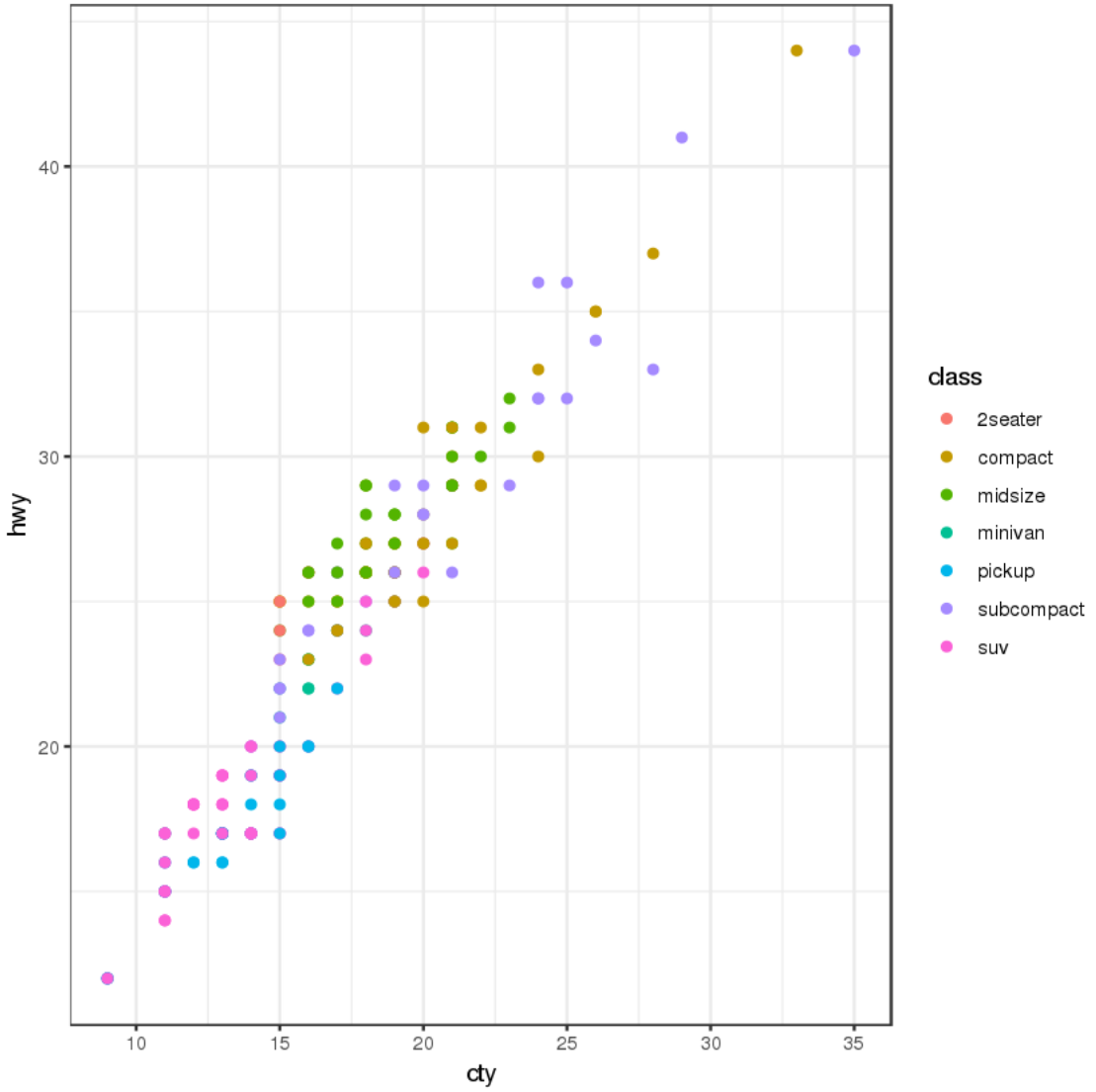
ggplot2 Grammar

```
library(ggplot2)
# Using the mpg dataset

# specifying characteristics inside aes()
ggplot(mpg, aes(cty, hwy)) +
  geom_point(aes(colour = class))

# specifying characteristics inside aes()
ggplot(mpg, aes(cty, hwy)) +
  geom_point(colour = "red")
```

What do you expect the output to be?

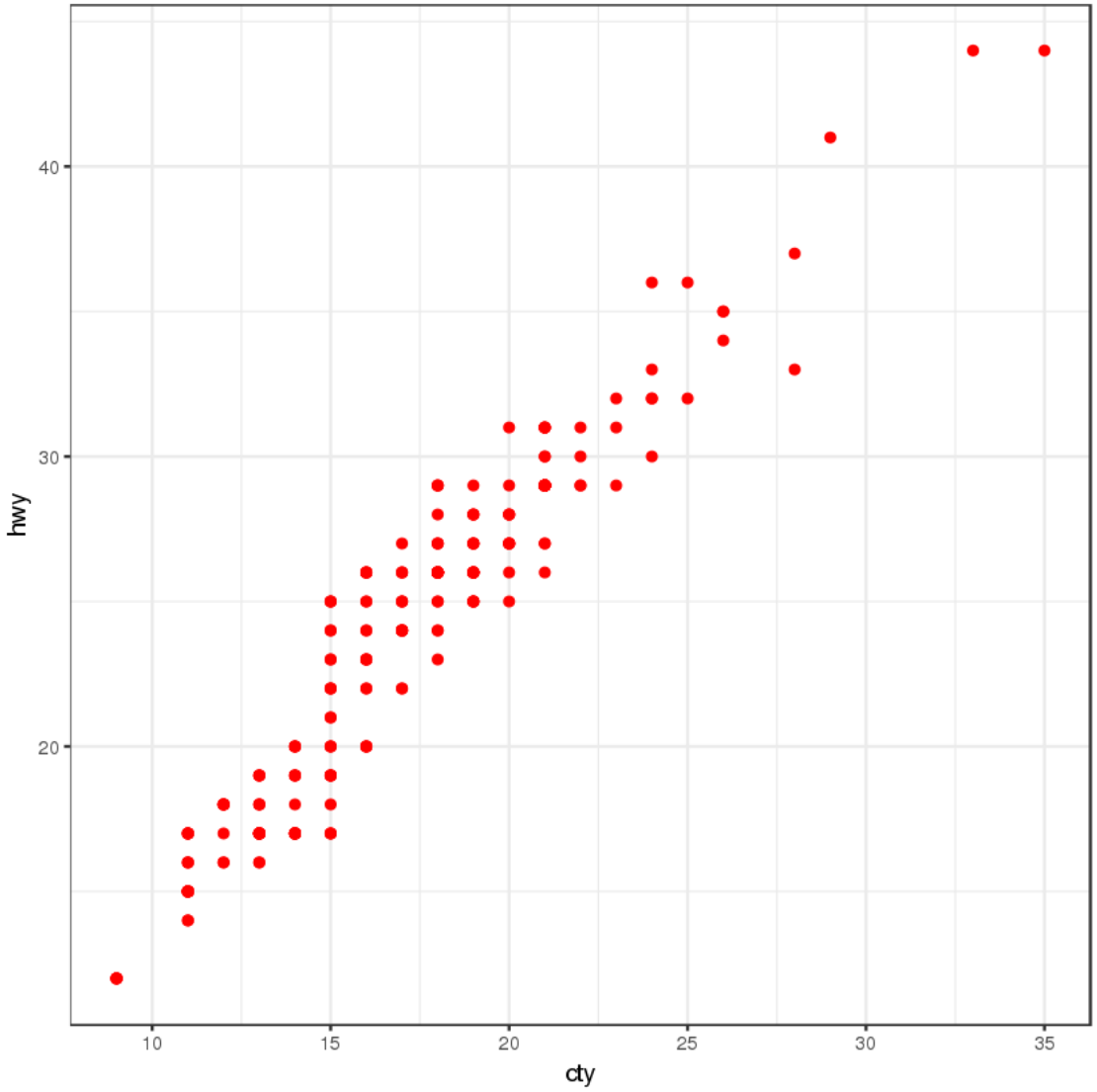
ggplot2 Grammar

```
library(ggplot2)
# Using the mpg dataset

# specifying characteristics inside aes()
ggplot(mpg, aes(cty, hwy)) +
  geom_point(aes(colour = class))

# specifying characteristics inside aes()
ggplot(mpg, aes(cty, hwy)) +
  geom_point(colour = "red")
```



ggplot2 Grammar

```
library(ggplot2)
# Using the mpg dataset

# specifying characteristics inside aes()
ggplot(mpg, aes(cty, hwy)) +
  geom_point(aes(colour = class))

# specifying characteristics inside aes()
ggplot(mpg, aes(cty, hwy)) +
  geom_point(colour = "red")
```



ggplot2 Grammar – Facets

In ggplot2 parlance, small multiples are referred to as facets

- `facet_wrap(), facet_grid()`

By default, all panels (one for each factor) share the same axes (scale-wise).

Separating the graph into a sequence of smaller, side-by-side plots makes it easier to enact comparisons.

Wraps only display those small multiples for which there is data, grids display all multiples, even the empty ones.

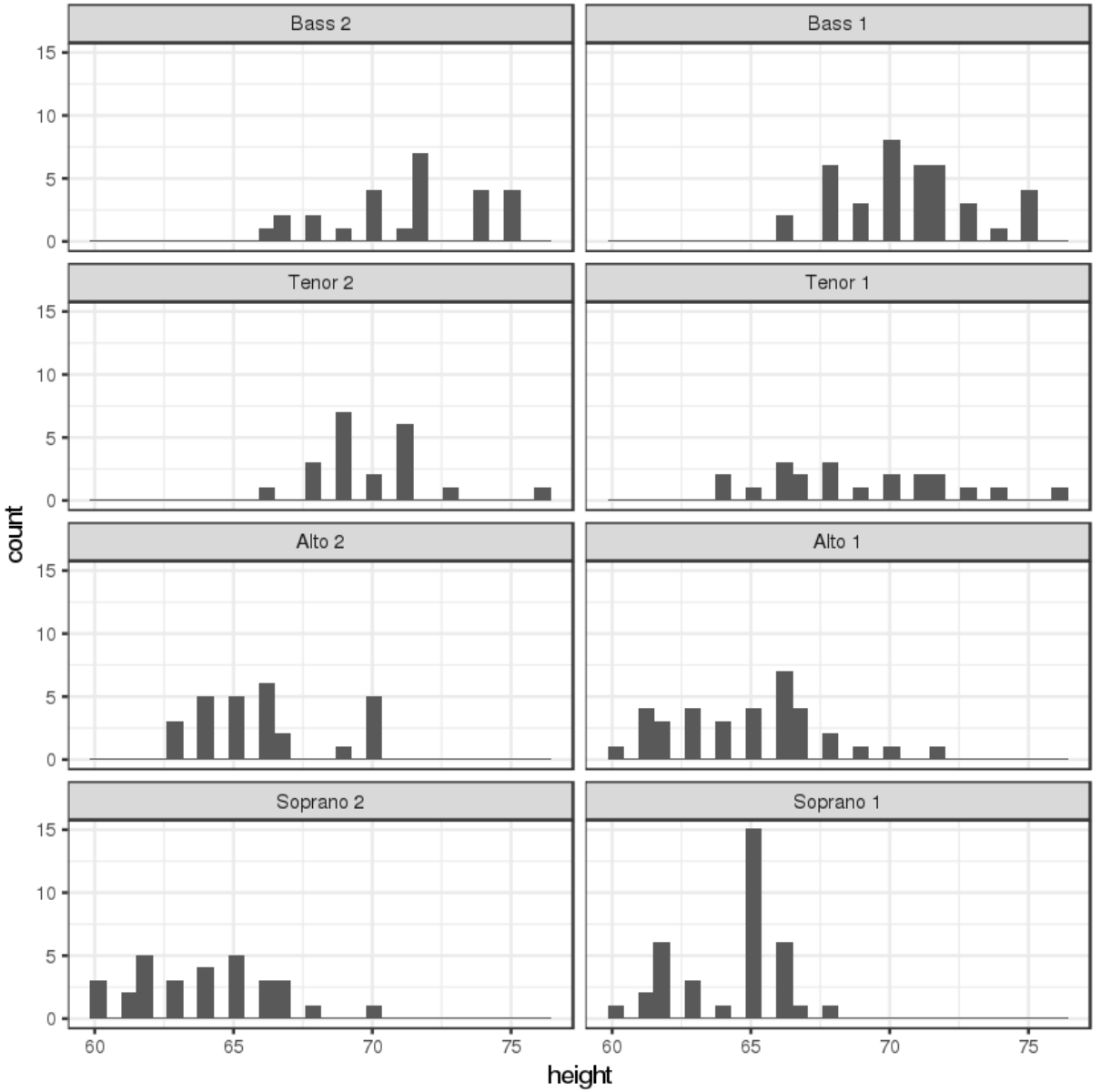
ggplot2 Grammar

```
data(singer, package="lattice")
library(ggplot2)
ggplot(data=singer, aes(x=height)) +
      geom_histogram() +
      facet_wrap(~voice.part, nrow=4)
```

What do you expect the output to be?

ggplot2 Grammar

```
data(singer, package="lattice")
library(ggplot2)
ggplot(data=singer, aes(x=height)) +
      geom_histogram() +
      facet_wrap(~voice.part, nrow=4)
```



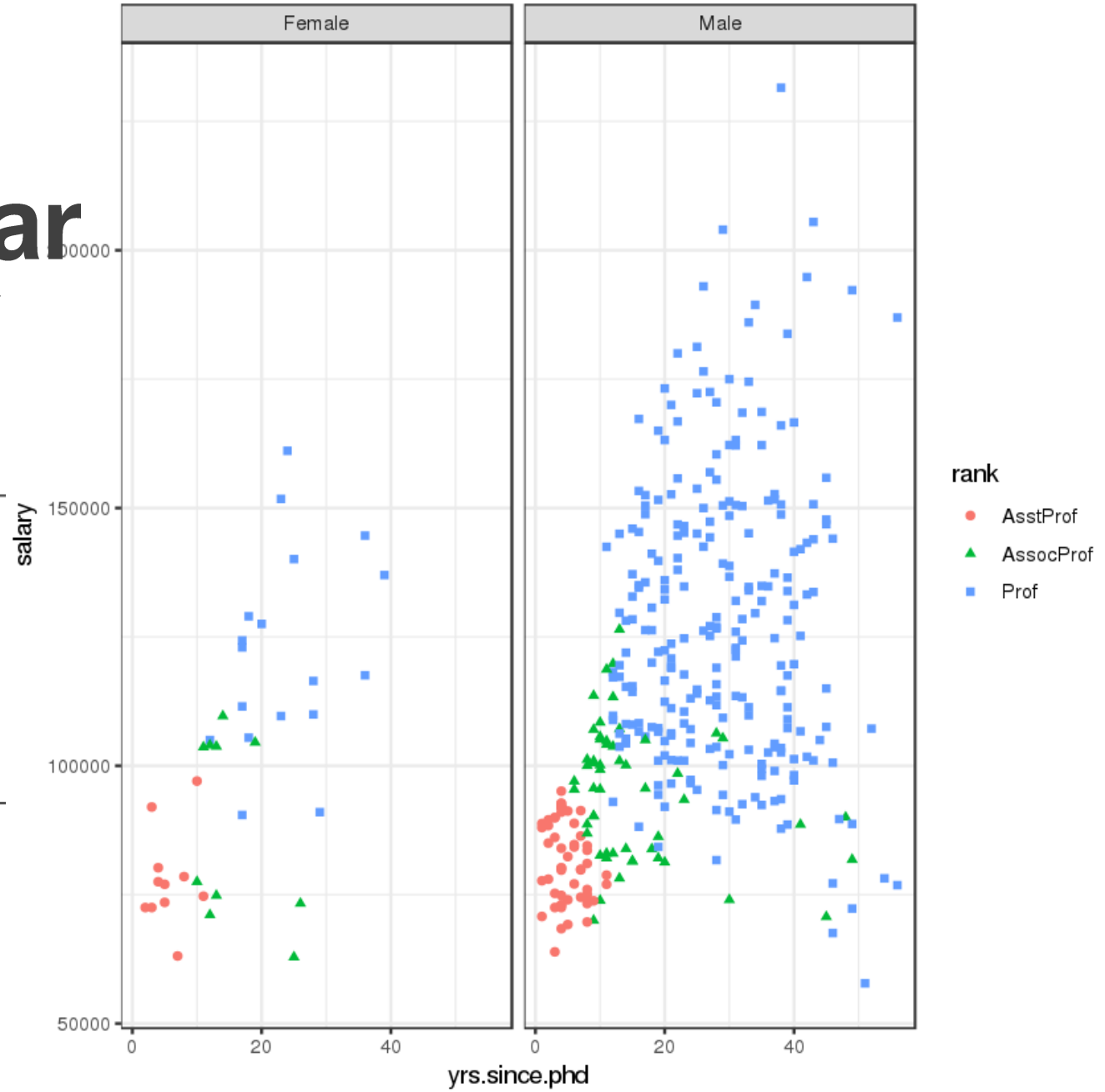
ggplot2 Grammar

```
data(Salaries, package="car")
library(ggplot2)
ggplot(Salaries, aes(x=yrs.since.phd,
 y=salary, color=rank, shape=rank)) +
   geom_point() +
   facet_grid(~sex)
```

What do you expect the output to be?

ggplot2 Grammar

```
data(Salaries, package="car")
library(ggplot2)
ggplot(Salaries, aes(x=yrs.since.phd,
 y=salary, color=rank, shape=rank)) +
   geom_point() +
   facet_grid(~sex)
```



Suggested Reading

Basics of `ggplot2`

The Practice of Data Visualization

Part IV: Some Practical Aspects of Data Visualization

12. `ggplot2` Visualizations in R
    - 12.1 Basics of `ggplot2`'s Grammar
    - 12.2 `ggplot2` Miscellanea

Exercises

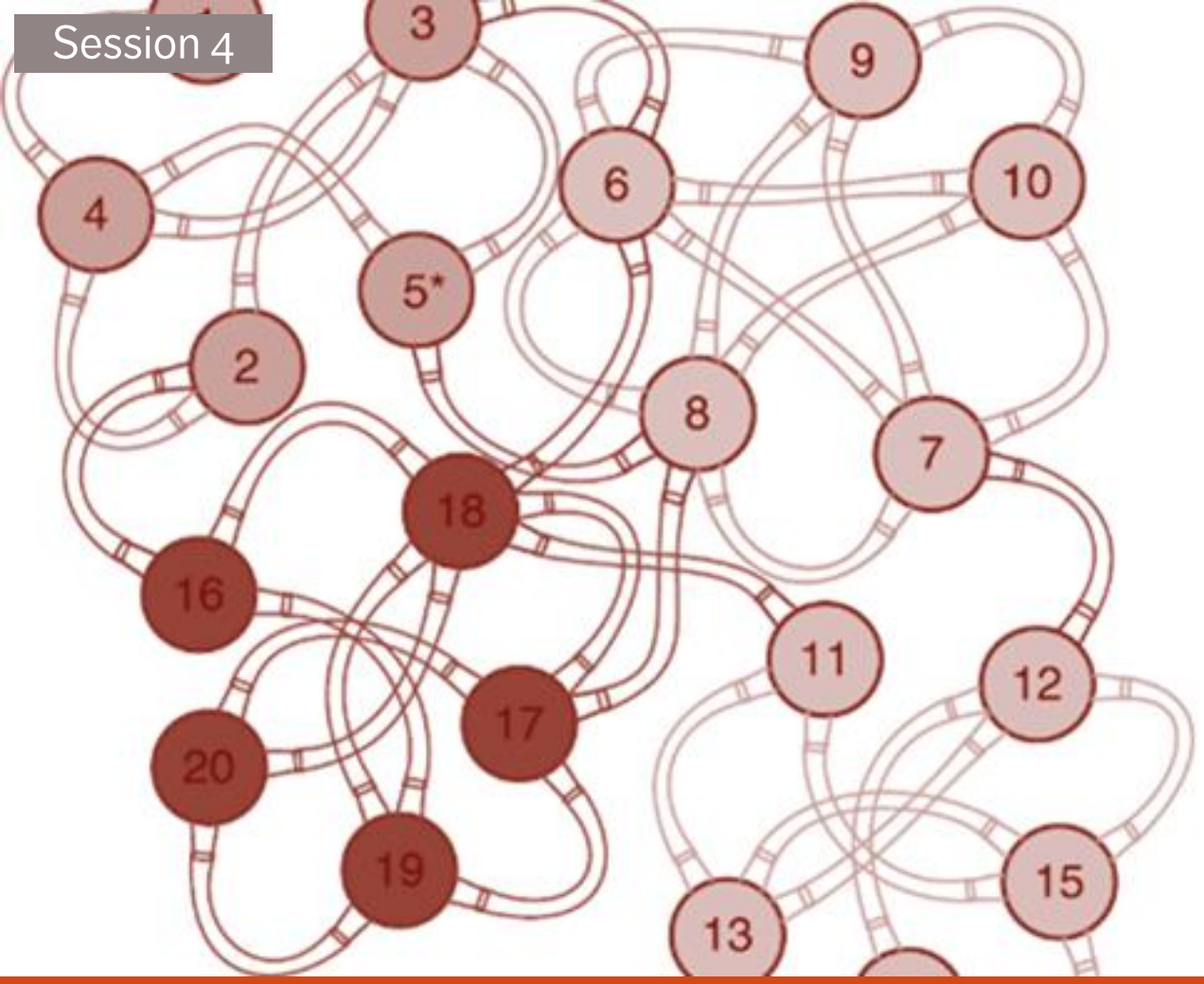
Basics of `ggplot2`

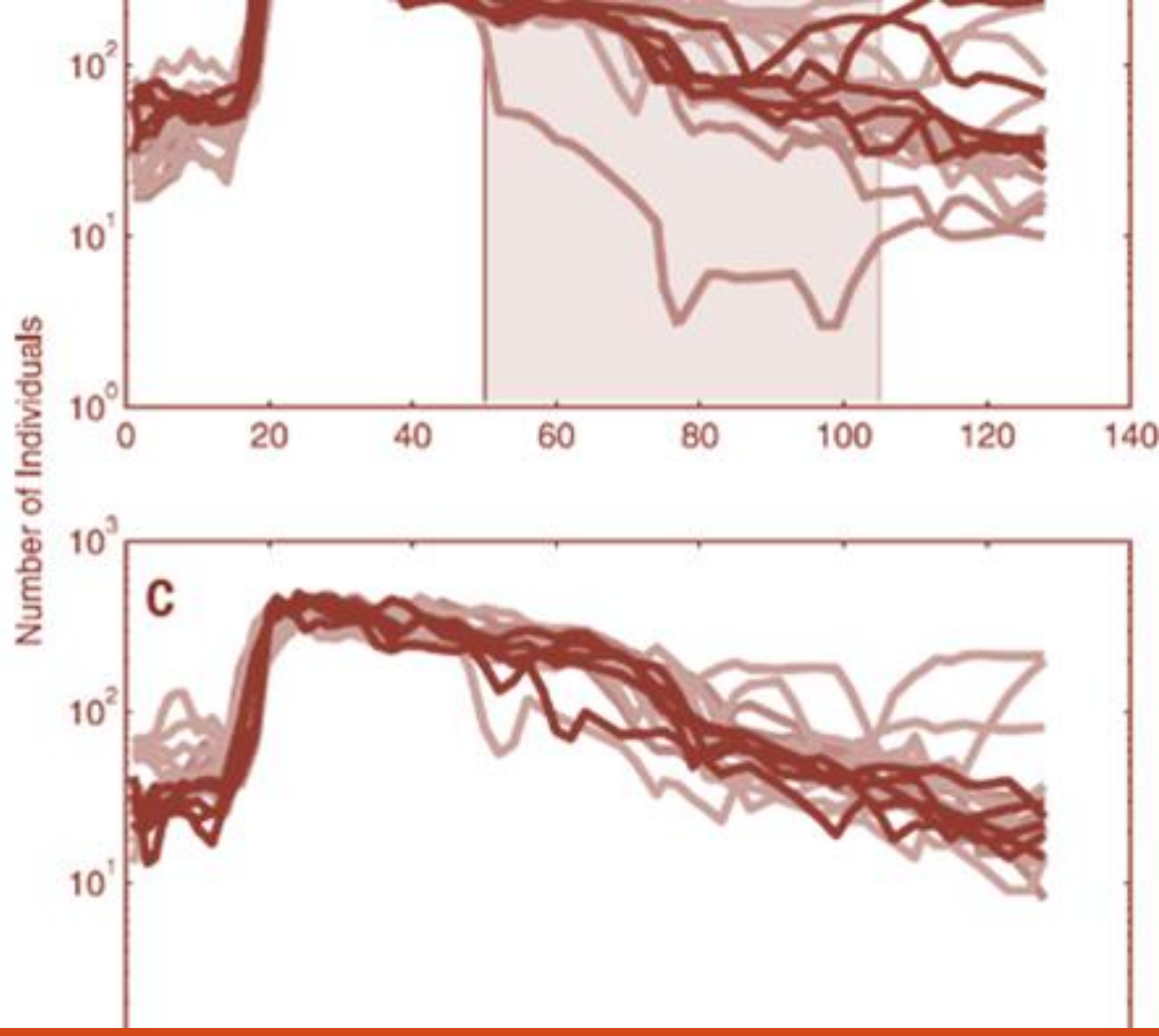
Create some simple `ggplot2` visualizations with data available in R. The emphasis is on becoming familiar with various geometries, their aesthetics, and the use of facets. You may use the examples found in this module as the basis of your work.

The available datasets are obtained by running

```
> data()
```

at the R command prompt.





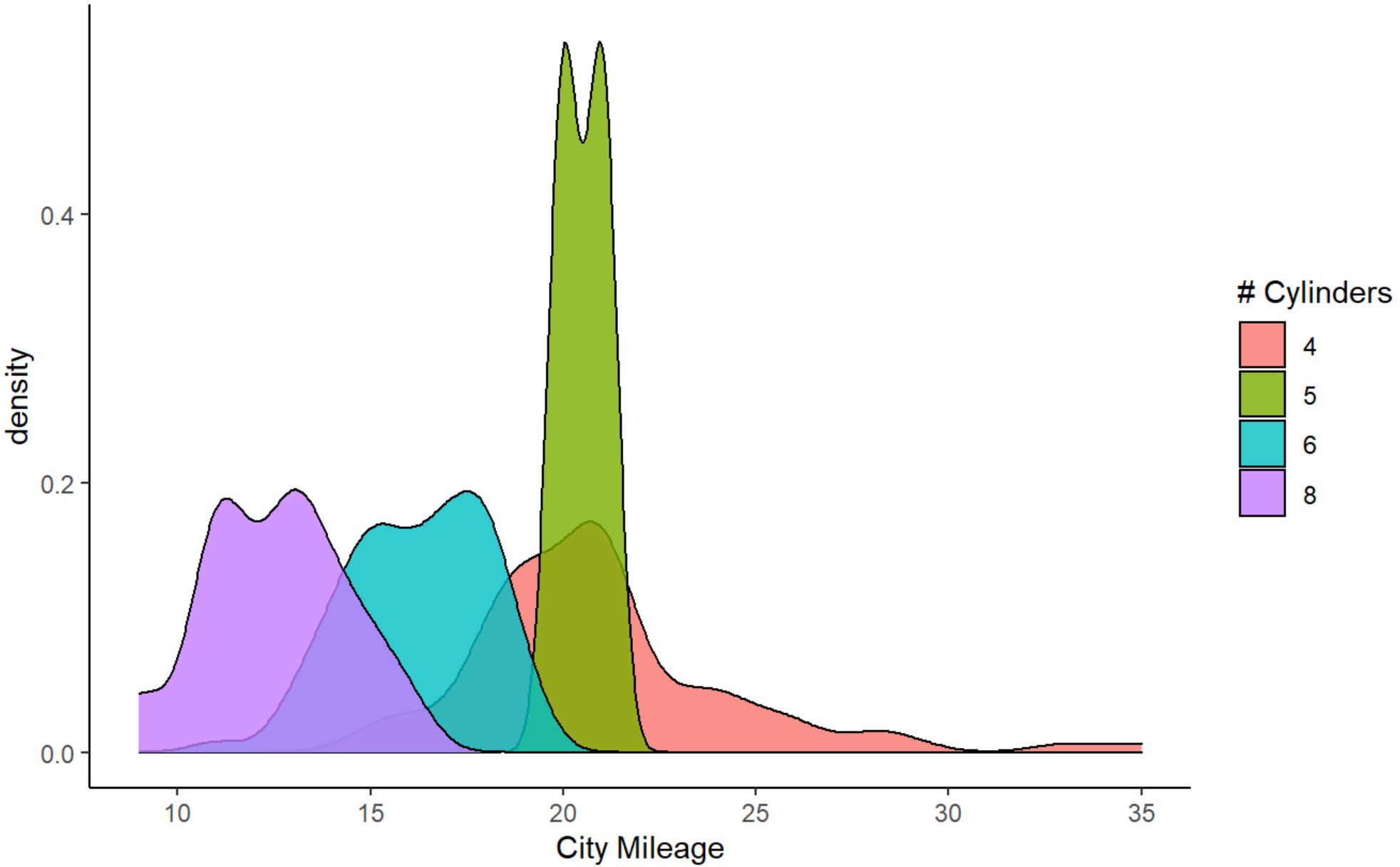
12. ggplot2 Examples and Miscellanea

Example 1

```
1  library(ggplot2)
2  theme_set(theme_classic())
3
4  # Plot
5  g <- ggplot(mpg, aes(cty))
6  g + geom_density(aes(fill=factor(cyl)), alpha=0.8) +
7      labs(title="Density Plot",
8           subtitle="City Mileage Grouped by Number of cylinders",
9           caption="Source: mpg",
10          x="City Mileage",
11          fill="# Cylinders")
```

Density Plot

City Mileage Grouped by Number of cylinders

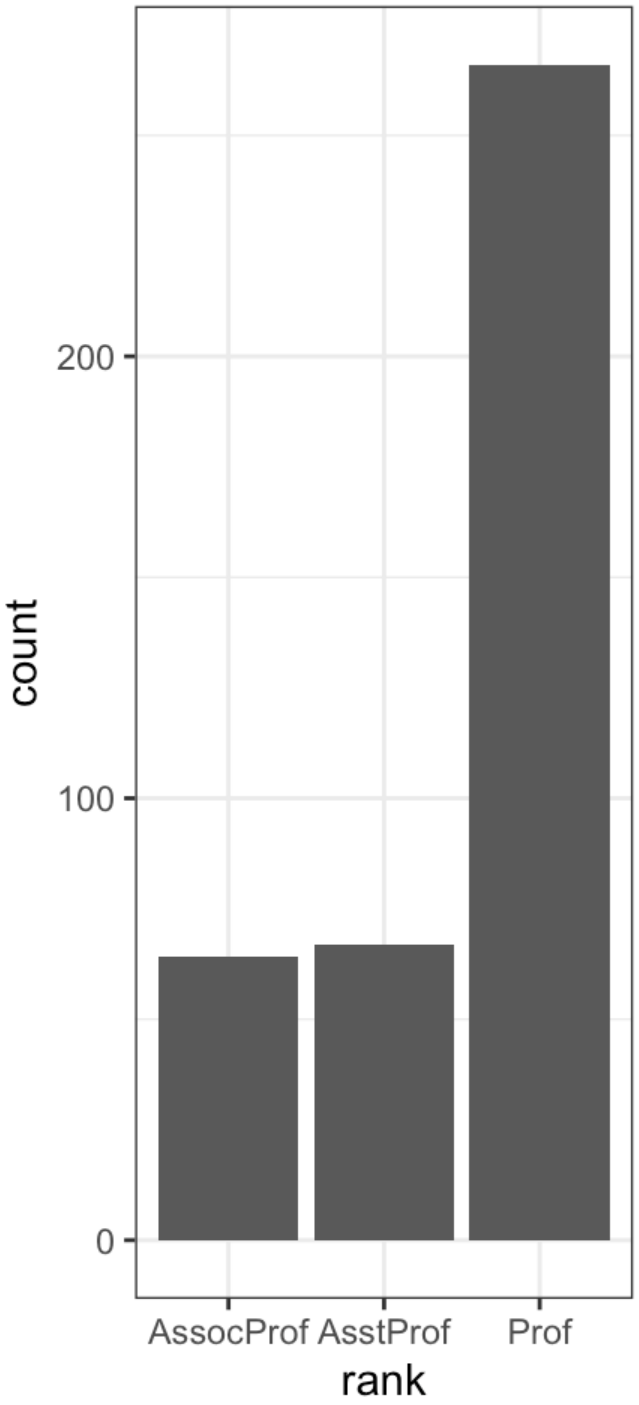


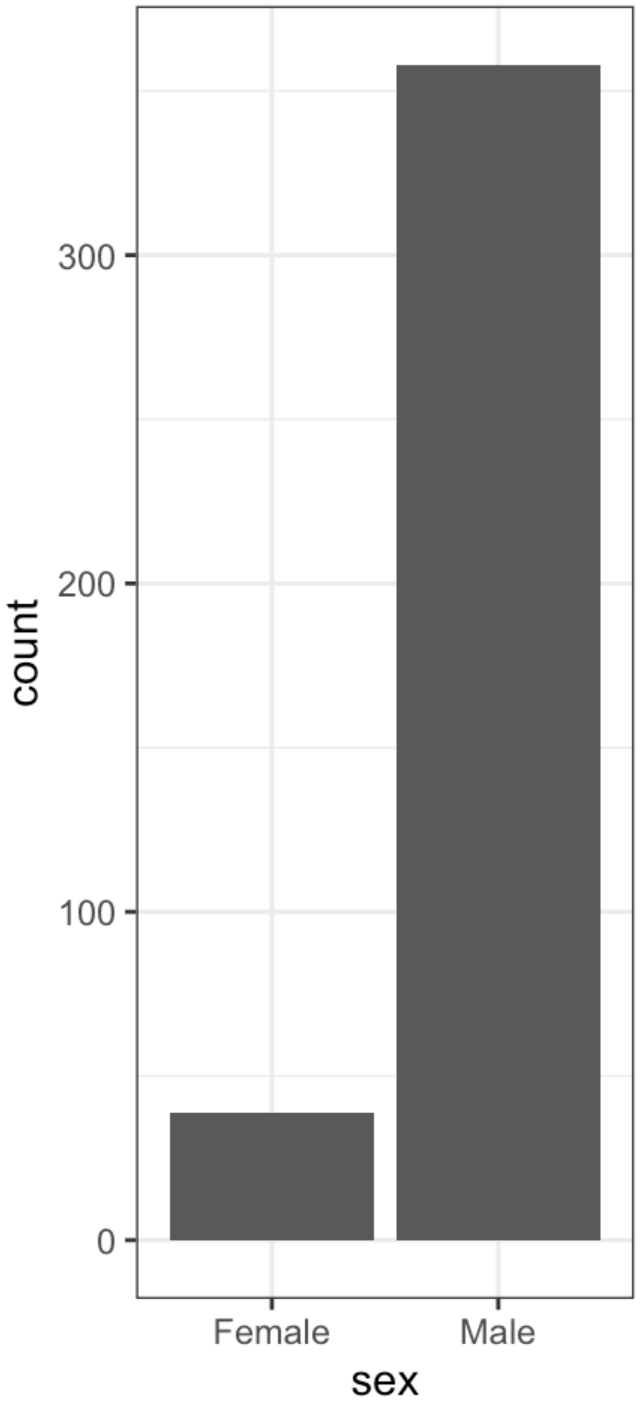
Source: mpg

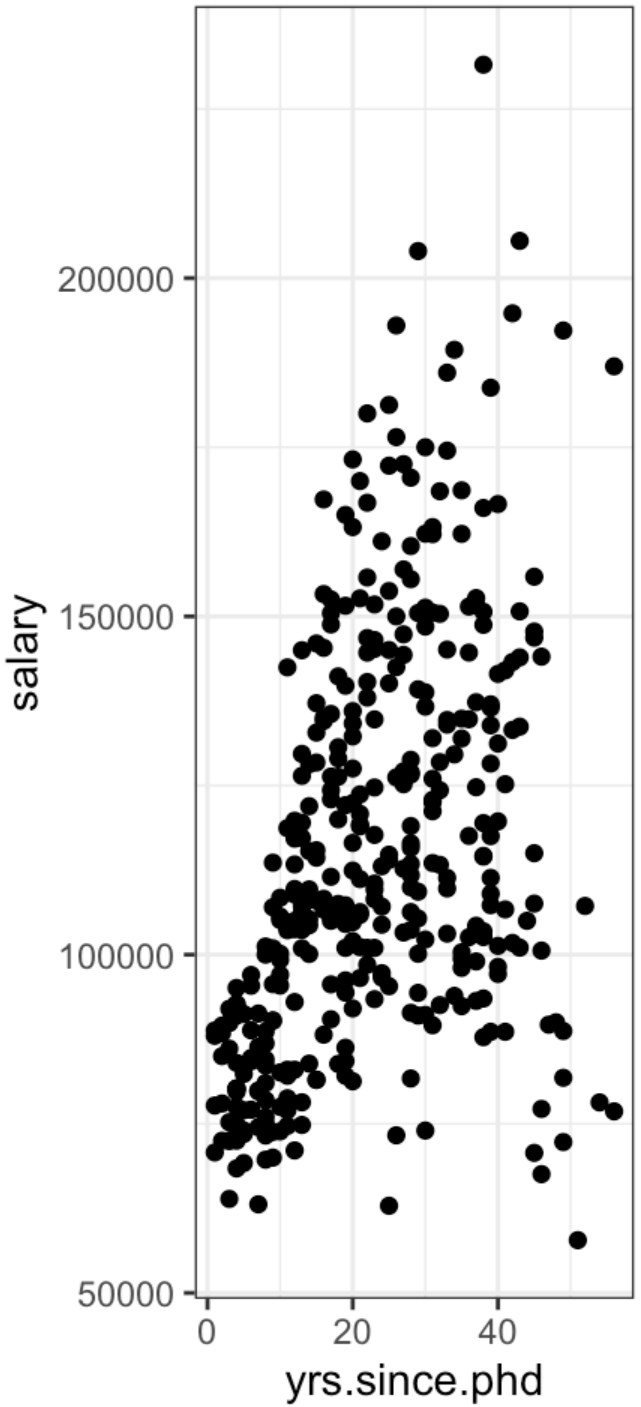
Example 2

```
1 library(ggplot2)
2 p1 <- ggplot(data=Salaries, aes(x=rank)) + geom_bar()
3 p2 <- ggplot(data=Salaries, aes(x=sex)) + geom_bar()
4 p3 <- ggplot(data=Salaries, aes(x=yrs.since.phd, y=salary)) + geom_point()
5
6 library(gridExtra)
7 grid.arrange(p1, p2, p3, ncol=3)
```

Session 4

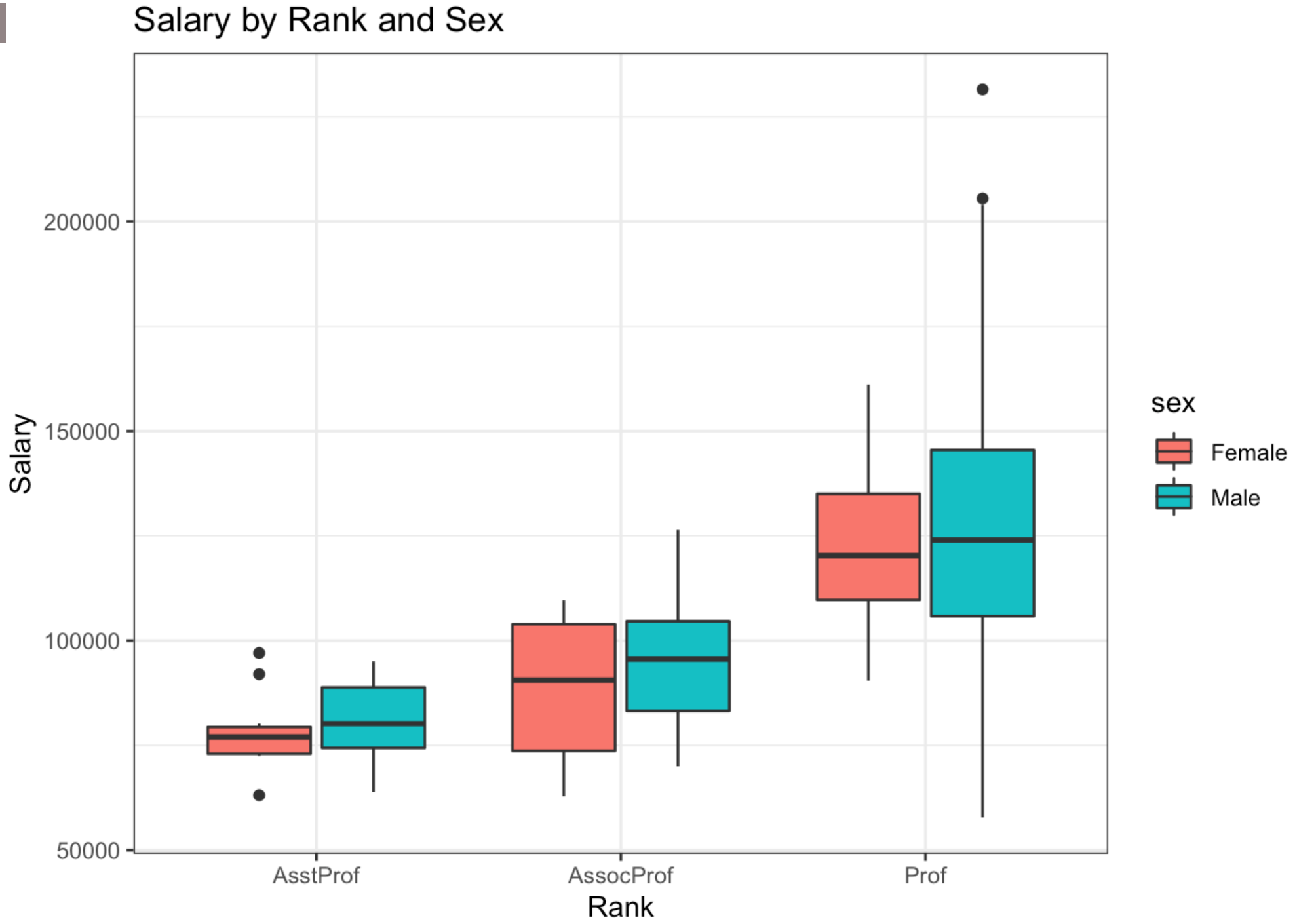

count
200
100
0
AssocProf
AsstProf
Prof
rank


count
300
200
100
0
Female
Male
sex


salary
200000
150000
100000
50000
0
20
40
yrs.since.phd

Example 3

```
1  library(ggplot2)
2  mytheme <- theme(plot.title=element_text(face="bold", size=14, color="brown"),
3                   axis.title=element_text(size=10, color="brown"),
4                   axis.text=element_text(size=9, color="black"),
5                   panel.background=element_rect(fill="white",color="black"),
6                   panel.grid.major.y=element_line(color="grey", linetype=1),
7                   panel.grid.minor.y=element_line(color="grey", linetype=2),
8                   panel.grid.minor.x=element_blank(),
9                   legend.position="top")
10
11 ggplot(Salaries, aes(x=reorder(rank,salary), y=salary, fill=sex)) +
12        geom_boxplot() +
13        labs(title="Salary by Rank and Sex", x="Rank", y="Salary")
```

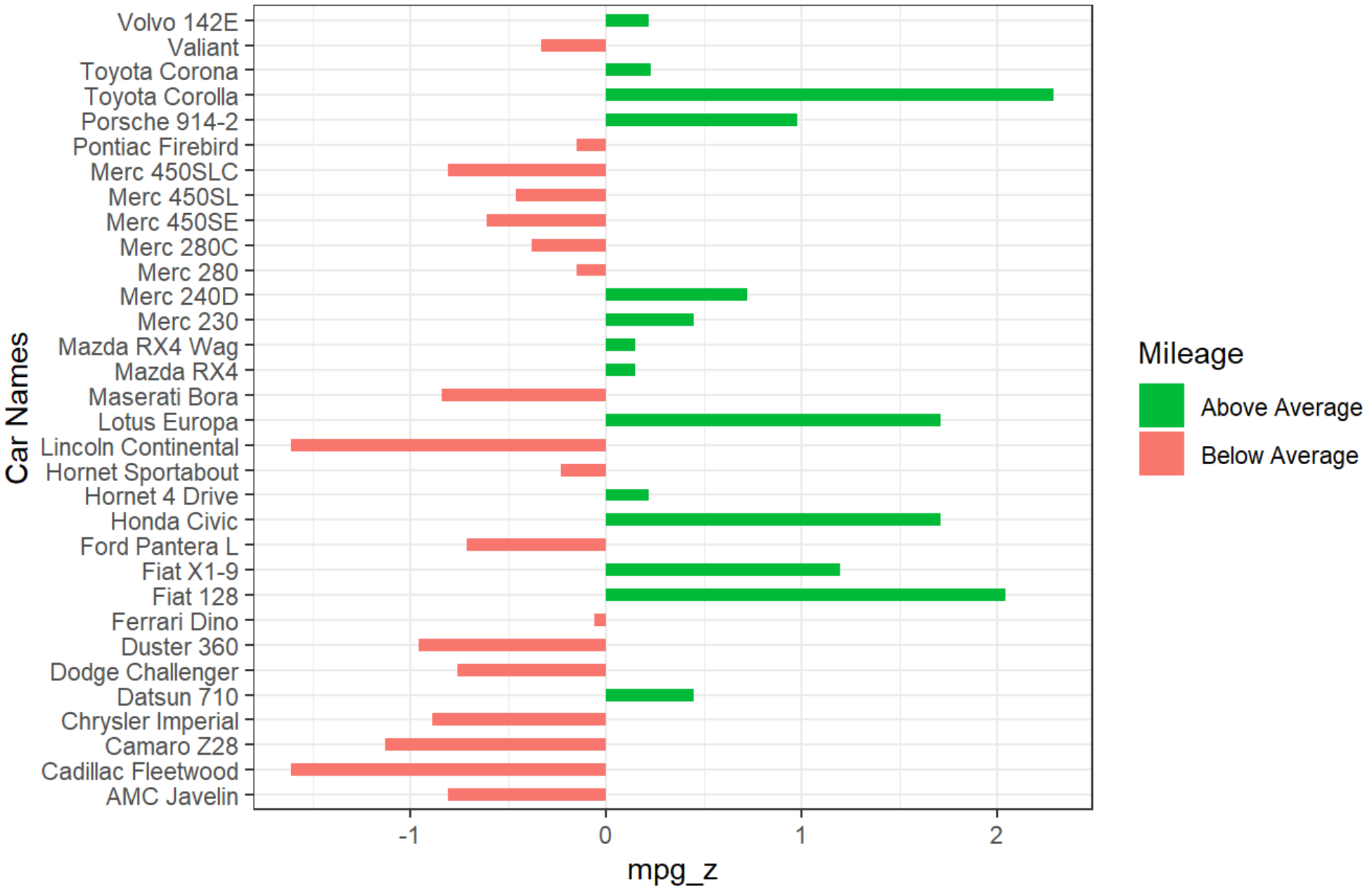

Salary by Rank and Sex
Salary
200000
150000
100000
50000
AsstProf
AssocProf
Prof
Rank
sex
Female
Male

Example 4

```r
library(ggplot2)
theme_set(theme_bw())

mtcars_new <- mtcars |>
  tibble::rownames_to_column(var = "car_name") |> # convert row names to a new column
  dplyr::mutate(car_name = as.factor(car_name),  # convert to factor to retain sorted order
  mpg_z = round(scale(mpg), 2),  # compute normalized mpg
  mpg_type = ifelse(mpg_z < 0, "below", "above")  # above / below avg flag
  ) |>
  dplyr::arrange(mpg_z) # sort

# Diverging Barcharts
ggplot(mtcars_new, aes(x= car_name, y=mpg_z, label=mpg_z)) +
  geom_bar(stat='identity', aes(fill=mpg_type), width=.5)  +
  scale_fill_manual(name="Mileage",
                    labels = c("Above Average", "Below Average"),
                    values = c("above"="#00ba38", "below"="#f8766d")) +
  labs(subtitle="Normalised mileage from 'mtcars'",
       title= "Diverging Bars",
       x = "Car Names") +
  coord_flip()
```

Diverging Bars

Normalised mileage from 'mtcars'

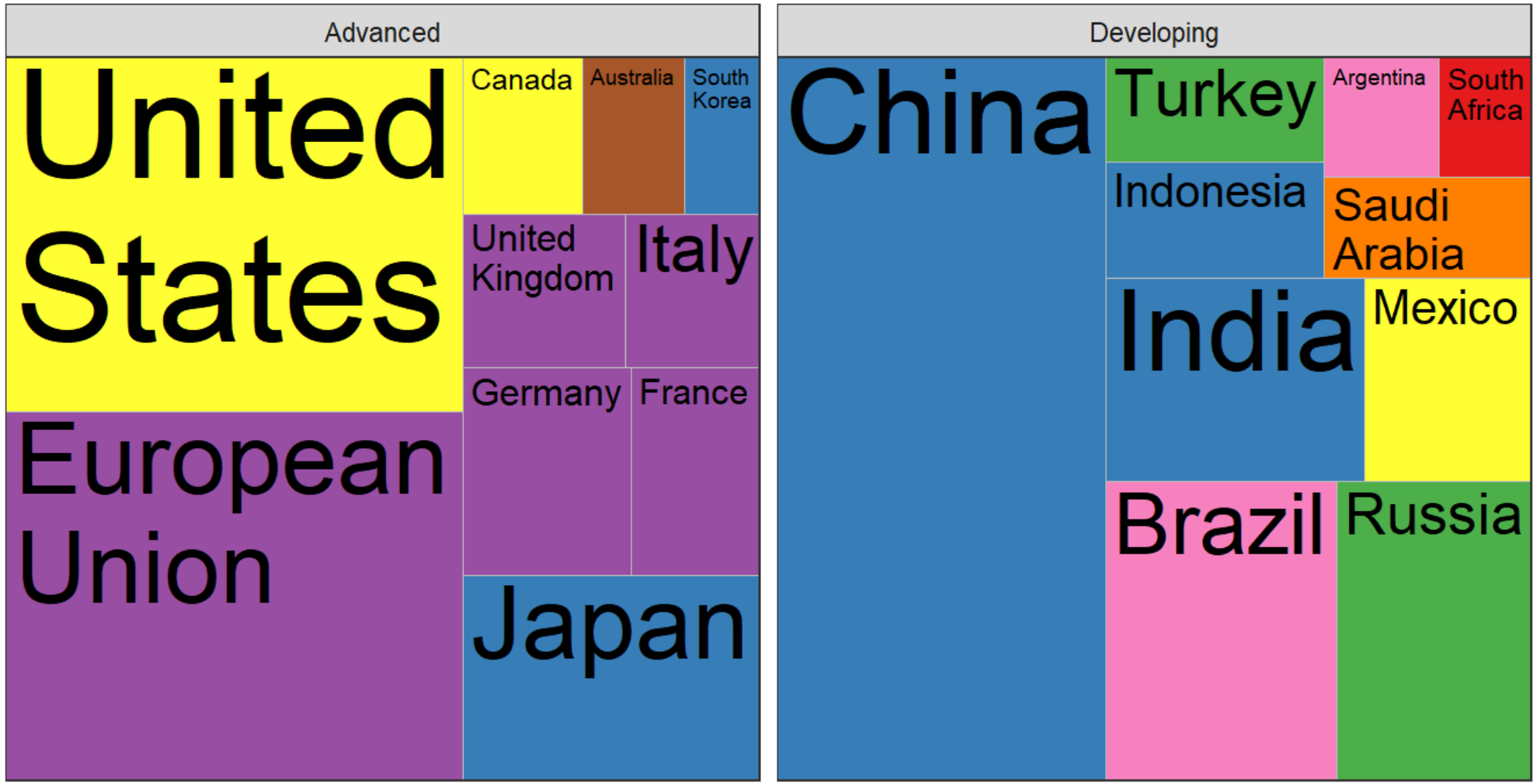


Example 5

```
library(devtools)
#devtools::install_github("wilkox/treemapify")
library(treemapify)
library(ggplot2)
data(G20)
head(G20)

ggplot(G20, aes(area = gdp_mil_usd, fill = region, label = country)) +
  geom_treemap() +
  geom_treemap_text(grow = T, reflow = T, colour = "black") +
  facet_wrap( ~ econ_classification) +
  scale_fill_brewer(palette = "Set1") +
  theme(legend.position = "bottom") +
  labs(
    title = "The G-20 major economies",
    caption = "The area of each country is proportional to its relative GDP
    within the economic group (advanced or developing)",
    fill = "Region"
  )
```

Session 4

The G-20 major economies

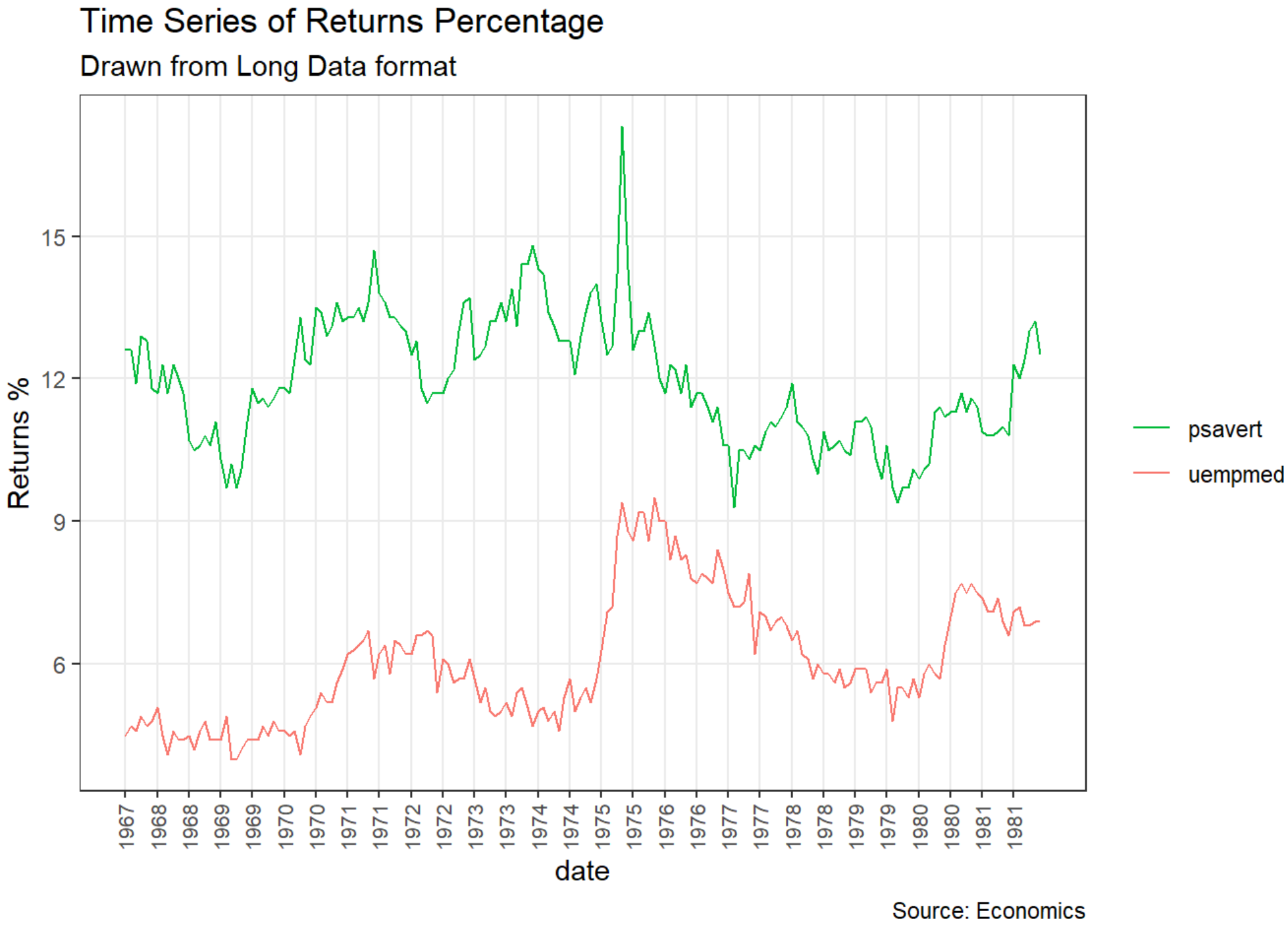


The area of each country is proportional to its relative GDP within the economic group (advanced or developing)

Example 6

```r
data(economics_long, package = "ggplot2")
head(economics_long)

library(ggplot2)
library(lubridate)
theme_set(theme_bw())
df <- economics_long[economics_long$variable %in% c("psavert", "uempmed"), ]
df <- df[lubridate::year(df$date) %in% c(1967:1981), ]

# labels and breaks for X axis text
brks <- df$date[seq(1, length(df$date), 12)]
lbls <- lubridate::year(brks)
```

```r
ggplot(df, aes(x=date)) +
  geom_line(aes(y=value, col=variable)) +
  labs(title="Time Series of Returns Percentage",
       subtitle="Drawn from Long Data format",
       caption="Source: Economics",
       y="Returns %",
       color=NULL) +  # title and caption
  scale_x_date(labels = lbls, breaks = brks) +  # change to monthly ticks and
  scale_color_manual(labels = c("psavert", "uempmed"),
                     values = c("psavert"="#00ba38", "uempmed"="#f8766d")) +
  theme(axis.text.x = element_text(angle = 90, vjust=0.5, size = 8),  # rotate
        panel.grid.minor = element_blank())  # turn off minor grid
```


Time Series of Returns Percentage
Drawn from Long Data format
Returns %
15
12
9
6
1967
1968
1968
1969
1969
1970
1970
1971
1971
1972
1972
1973
1973
1974
1974
1975
1975
1976
1976
1977
1977
1978
1978
1979
1979
1980
1980
1981
1981
date
psavert
uempmed
Source: Economics

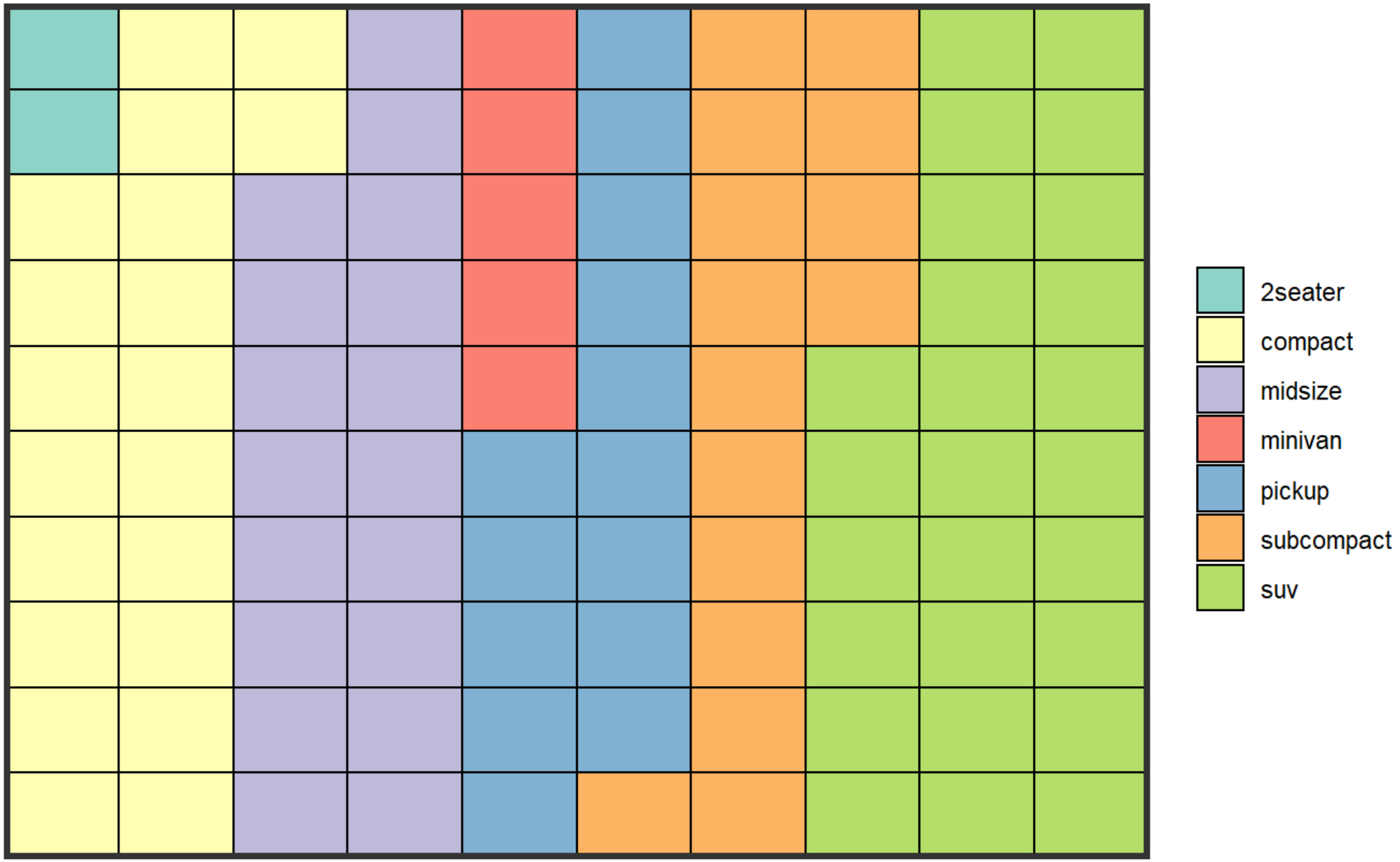
Example 7

```r
library(ggplot2)
var <- mpg$class  # the categorical data
## Prep data (nothing to change here)
nrows <- 10
df <- expand.grid(y = 1:nrows, x = 1:nrows)
categ_table <- round(table(var) * ((nrows*nrows)/(length(var))))
categ_table

df$category <- factor(rep(names(categ_table), categ_table))
ggplot(df, aes(x = x, y = y, fill = category)) +
        geom_tile(color = "black", size = 0.5) +
        scale_x_continuous(expand = c(0, 0)) +
        scale_y_continuous(expand = c(0, 0), trans = 'reverse') +
        scale_fill_brewer(palette = "Set3") +
        labs(title="Waffle Chart", subtitle="'Class' of vehicles",
             caption="Source: mpg") +
        theme(panel.border = element_rect(size = 2),
              plot.title = element_text(size = rel(1.2)),
              axis.text = element_blank(),
              axis.title = element_blank(),
              axis.ticks = element_blank(),
              legend.title = element_blank(),
              legend.position = "right")
```

Session 4

Waffle Chart

'Class' of vehicles

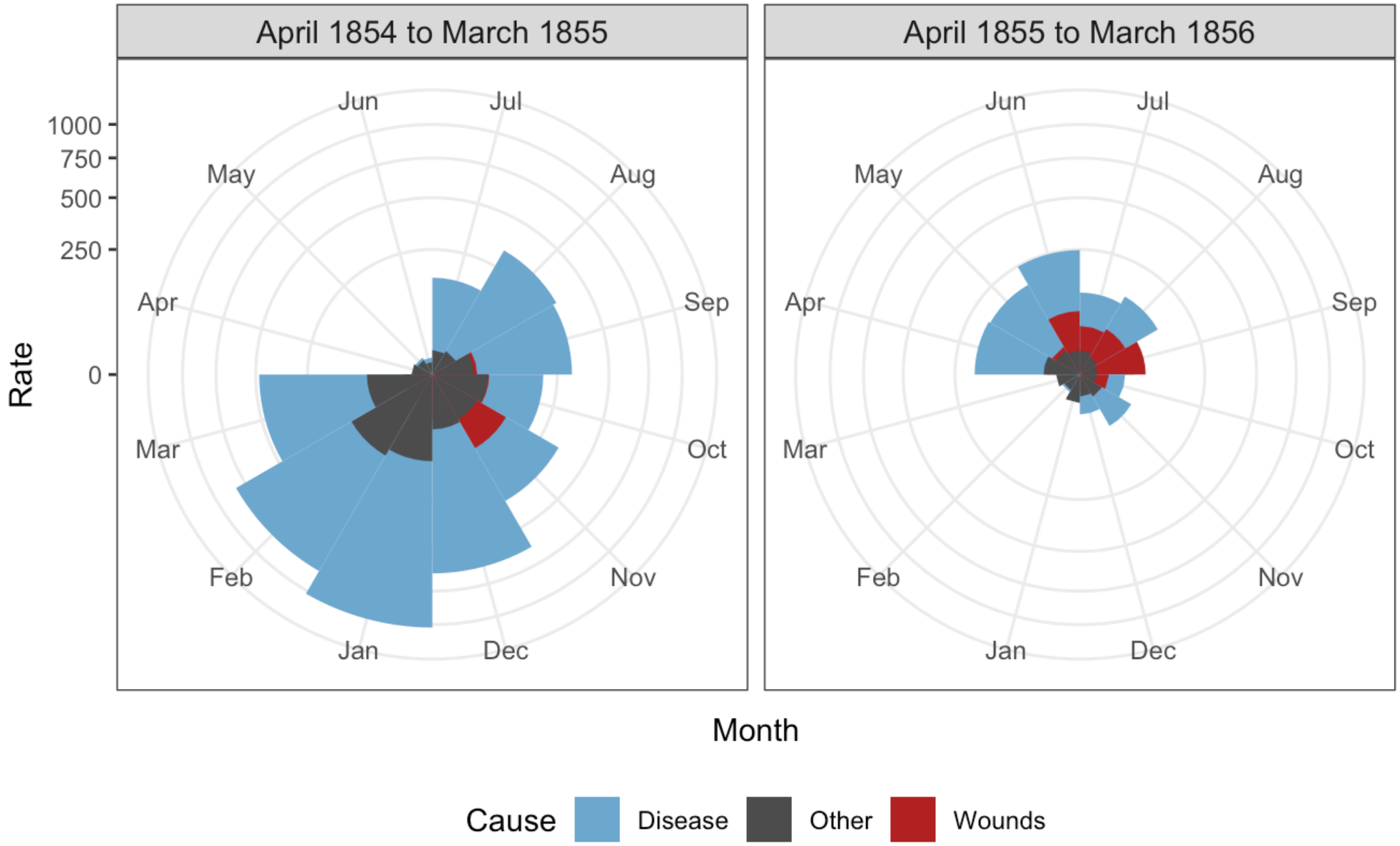


Source: mpg

Example 8

```
library(tidyverse)
library(HistData)

Nightingale %>%
  select(Date, Month, Year, contains("rate")) %>%
  pivot_longer(cols = 4:6, names_to = "Cause", values_to = "Rate") %>%
  mutate(Cause = gsub(".rate", "", Cause),
         period = ifelse(Date <= as.Date("1855-03-01"), "April 1854 to March 1855", "April 1855 to March 1856"),
         Month = fct_relevel(Month, "Jul", "Aug", "Sep", "Oct", "Nov", "Dec", "Jan", "Feb", "Mar", "Apr", "May", "Jun")) %>%
  ggplot(aes(Month, Rate)) +
  geom_col(aes(fill = Cause), width = 1, position = "identity") +
  coord_polar() +
  facet_wrap(~period) +
  scale_fill_manual(values = c("skyblue3", "grey30", "firebrick")) +
  scale_y_sqrt() +
  theme_bw() +
  theme(axis.text.x = element_text(size = 9),
        strip.text = element_text(size = 11),
        legend.position = "bottom",
        plot.margin = unit(c(10, 10, 10, 10), "pt"),
        plot.title = element_text(vjust = 5)) +
  ggtitle("Diagram of the Causes of Mortality in the Army in the East")
```

Diagram of the Causes of Mortality in the Army in the East



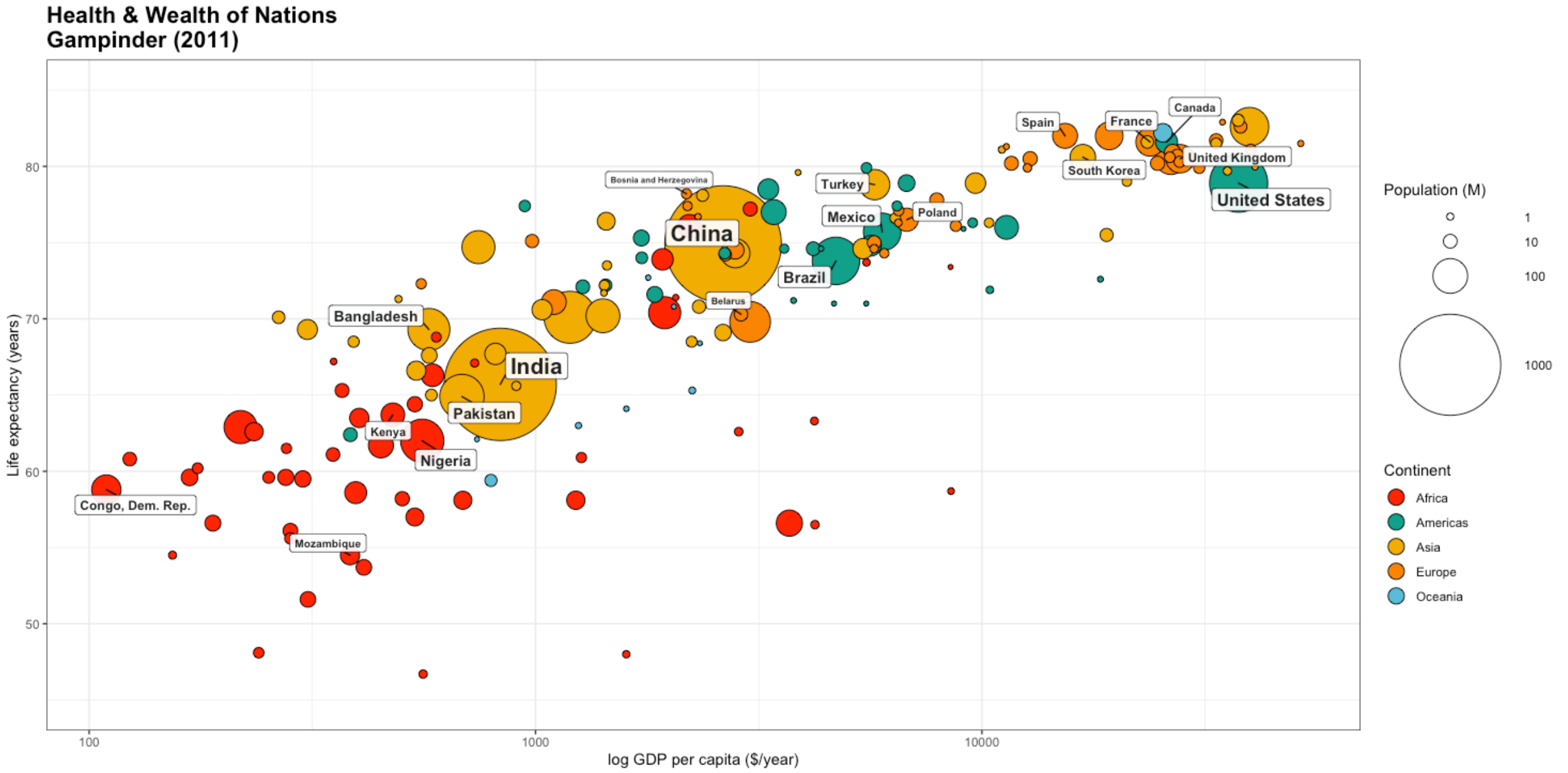
Example 9

```
library(tidyverse)
gapminder_ds <- dslabs::gapminder # for the dataset
library(wesanderson)  # for the colour palette
library(ggrepel)      # for country names on chart
```

```
yr <- 2009
chart_title <- paste("Health & Wealth of Nations \nGampinder (",yr,")",sep="")
```

```
# sort the countries by inverse population
gapminder_ds <- gapminder_ds |>
  dplyr::arrange(year, dplyr::desc(population))

# select which countries will have their names labelled
num_countries <- 20

# tidyverse + ggplot2
filtered_gapminder <- gapminder_ds %>%
  filter(year==yr) %>%
  mutate(pop_m = population/1e6, gdppc=gdp/population)

weights <- filtered_gapminder$pop_m/sum(filtered_gapminder$pop_m)
p <- sample(nrow(filtered_gapminder), num_countries, prob = weights)

filtered_gapminder$country_display <- ifelse(ifelse(1:185 %in% p, TRUE,FALSE),as.character(filtered_gapminder$country),"")
```

```
filtered_gapminder %>%
  ggplot(aes(x = gdppc,
             y=life_expectancy,
             size=pop_m)) +
  geom_point(aes(fill=continent), pch=21) +
  scale_fill_manual(values=wes_palette(n=5, name="Darjeeling1")) +
  scale_x_log10() +
  geom_label_repel(aes(label=country_display, size=sqrt(pop_m/pi)),
                   alpha=0.9,
                   fontface="bold",
                   min.segment.length = unit(0, 'lines'),
                   show.legend=FALSE) +
  ggtitle(chart_title) +
  theme(plot.title = element_text(size=14, face="bold")) +
  xlab('log GDP per capita ($/year)') +
  ylab('Life expectancy (years)') +
  ylim(45,85) +
  scale_size_continuous(range=c(1,40),
                        breaks = c(1,10,100,1000),
                        limits = c(0, 1500),
                        labels = c("1","10","100","1000")) +
  guides(fill = guide_legend(override.aes = list(size = 5))) +
  labs(fill="Continent", size="Population (M)") +
  theme_bw() +
  theme(plot.title = element_text(size=16, face="bold"))
```


Health & Wealth of Nations
Gampinder (2011)
Life expectancy (years)
80
70
60
50
100
1000
10000
log GDP per capita ($/year)
Bosnia and Herzegovina
Turkey
Spain
France
Canada
South Korea
United Kingdom
United States
China
Mexico
Poland
Brazil
Belarus
Bangladesh
India
Pakistan
Kenya
Nigeria
Congo, Dem. Rep.
Mozambique
Population (M)
1
10
100
1000
Continent
Africa
Americas
Asia
Europe
Oceania

Suggested Reading

`ggplot2` Examples and Miscellanea

The Practice of Data Visualization

Part IV: Some Practical Aspects of Data Visualization

12. `ggplot2` Visualizations in R

    12.3 Examples

Exercises

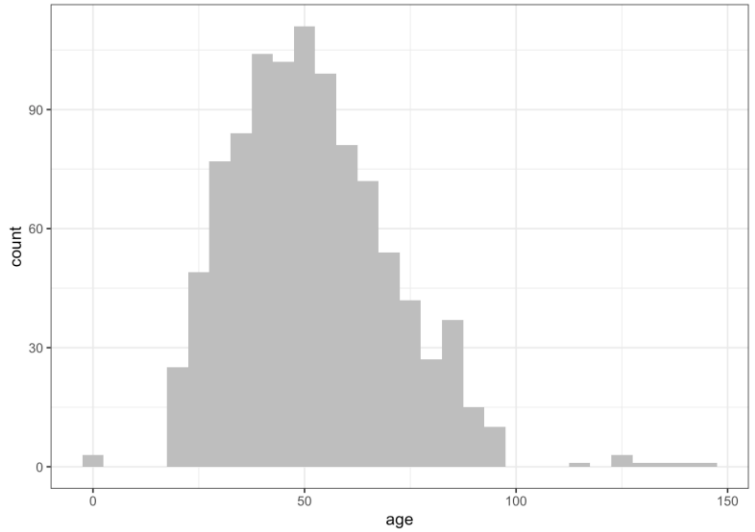
`ggplot2` Examples and Miscellanea

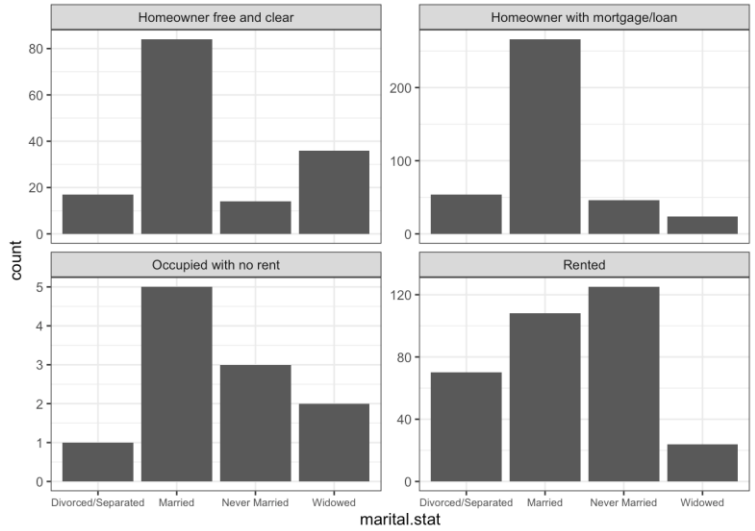
(Try this exercise without first consulting the U.S. Census PUMS Data example in DUDADS).

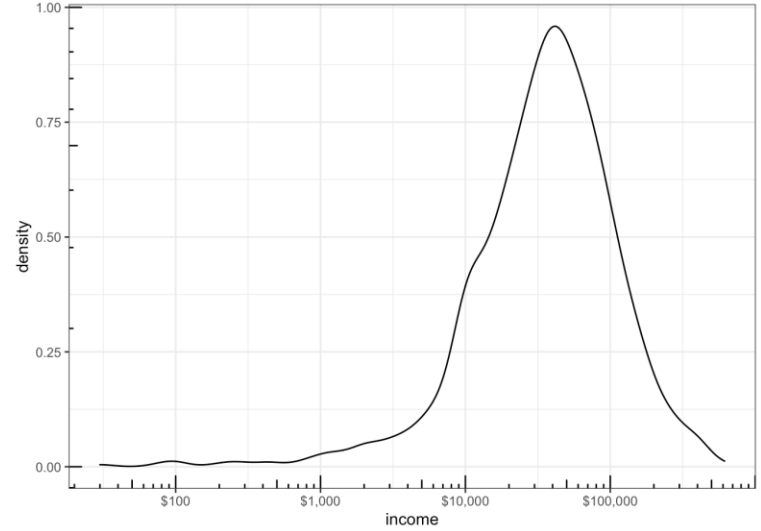
The custdata.tsv file is derived from U.S. Census PUMS data. This synthetic dataset contains customer information for individuals whose health insurance status is known.

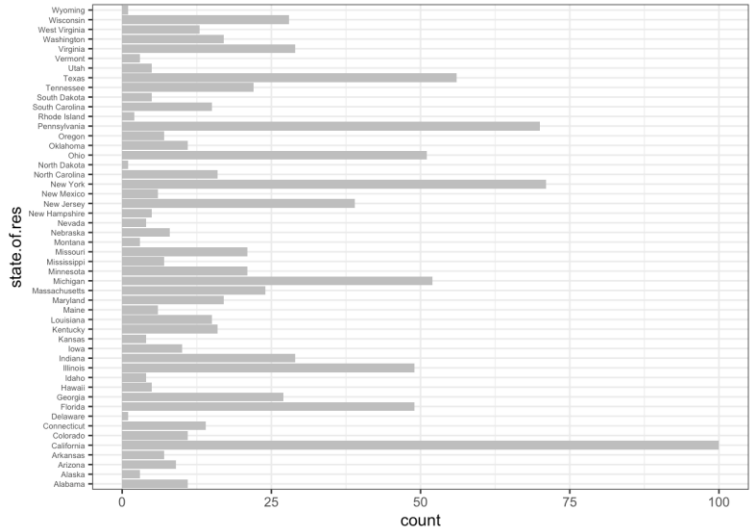
Recreate the following `ggplot2` charts (ignore missing values for the time being).

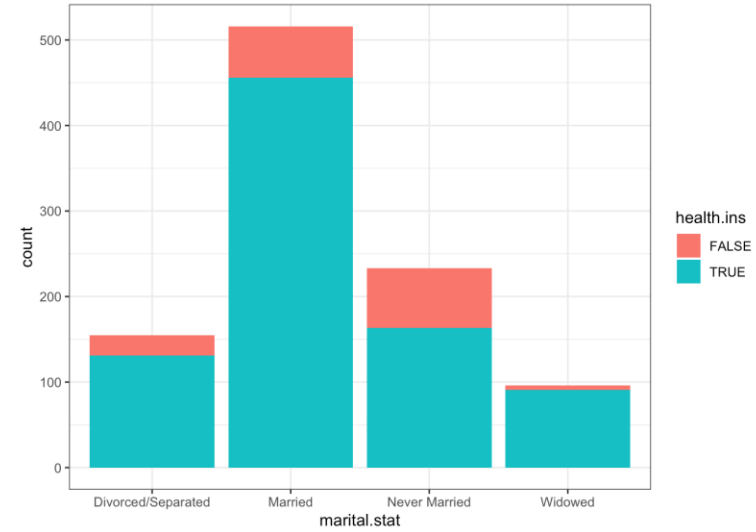
Session 4

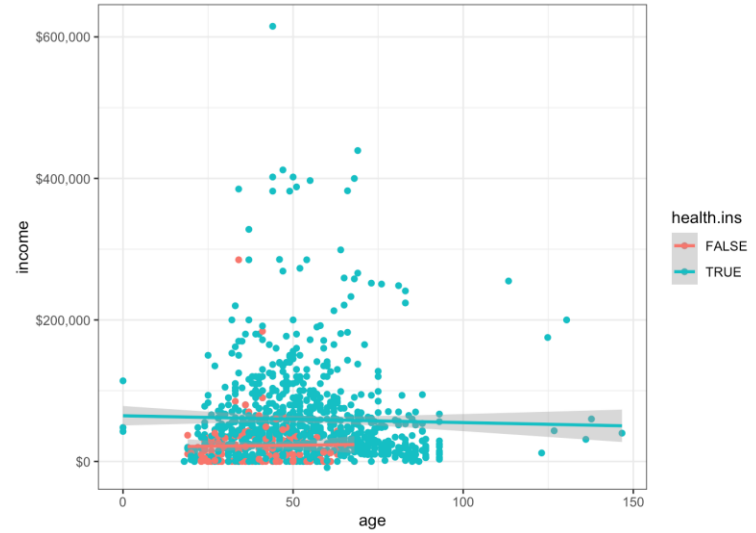

count
age
0
30
60
90
0
50
100
150


Homeowner free and clear
Homeowner with mortgage/loan
Occupied with no rent
Rented
count
marital.stat
Divorced/Separated
Married
Never Married
Widowed


density
income
0.00
0.25
0.50
0.75
1.00
$100
$1,000
$10,000
$100,000


state.of.res
count
0
25
50
75
100


count
marital.stat
health.ins
FALSE
TRUE
Divorced/Separated
Married
Never Married
Widowed


income
age
health.ins
FALSE
TRUE
$0
$200,000
$400,000
$600,000