Data visualization 2. descriptive statistics

STA 032: Gateway to data science Lecture 8

Jingwei Xiong

April 19, 2023

Reminders

- HW 2 due April 26 12pm.
- Please start the homework as soon as possible.

Today

- Facets
- Time series
- Descriptive statistics

Recap (With lecture7 example recaps)

- Data visualization with ggplot

> Remember, before using all tidyverse functions, you need to library(tidyverse) first!
>
> Remember, before using all ggplot2 functions, you need to library(ggplot2) first!

A note on piping and layering

- Pipe `%>%` used mainly in `dplyr` pipelines
    - Pipe the output of the previous line of code as the first input of the next line of code
- `+` used in `ggplot2` plots is used for "layering"
    - Create the plot in layers, separated by `+`

dplyr

❌

```
hotels +
  select(hotel, lead_time)
```

```
Error in select(hotel, lead_time): object 'hotel' not found
```

✅

```
hotels %>%
  select(hotel, lead_time)
```

```
# A tibble: 119,390 × 2
   hotel        lead_time
   <chr>            <dbl>
 1 Resort Hotel       342
 2 Resort Hotel       737
 3 Resort Hotel         7
 4 Resort Hotel        13
 5 Resort Hotel        14
 6 Resort Hotel        14
 7 Resort Hotel         0
 8 Resort Hotel         9
 9 Resort Hotel        85
10 Resort Hotel        75
# … with 119,380 more rows
```

ggplot2

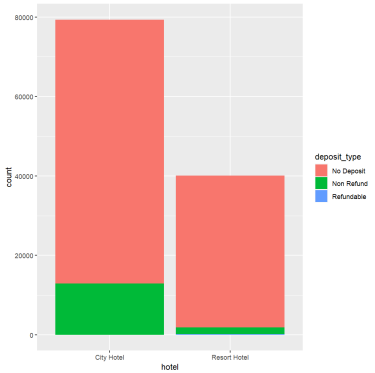
❌

```
ggplot(hotels, aes(x = hotel, fill = deposit_type)) %>%
  geom_bar()
```

```
Error in geom_bar():
! mapping must be created by aes()
ℹ Did you use %>% or |> instead of +?
```

✅

```
ggplot(hotels, aes(x = hotel, fill = deposit_type)) +
  geom_bar()
```



Code styling

Many of the styling principles are consistent across %>% and +:

- always a space before
- always a line break after (for pipelines with more than 2 lines)

❌

```
ggplot(hotels,aes(x=hotel,y=deposit_type))+geom_bar()
```

✅

```
ggplot(hotels, aes(x = hotel, y = deposit_type)) +
  geom_bar()
```

Today

- Finishing up on `ggplot()`
  - Faceting using `facet_grid()`
  - Time series plot
- Descriptive statistics

`facet_grid()`

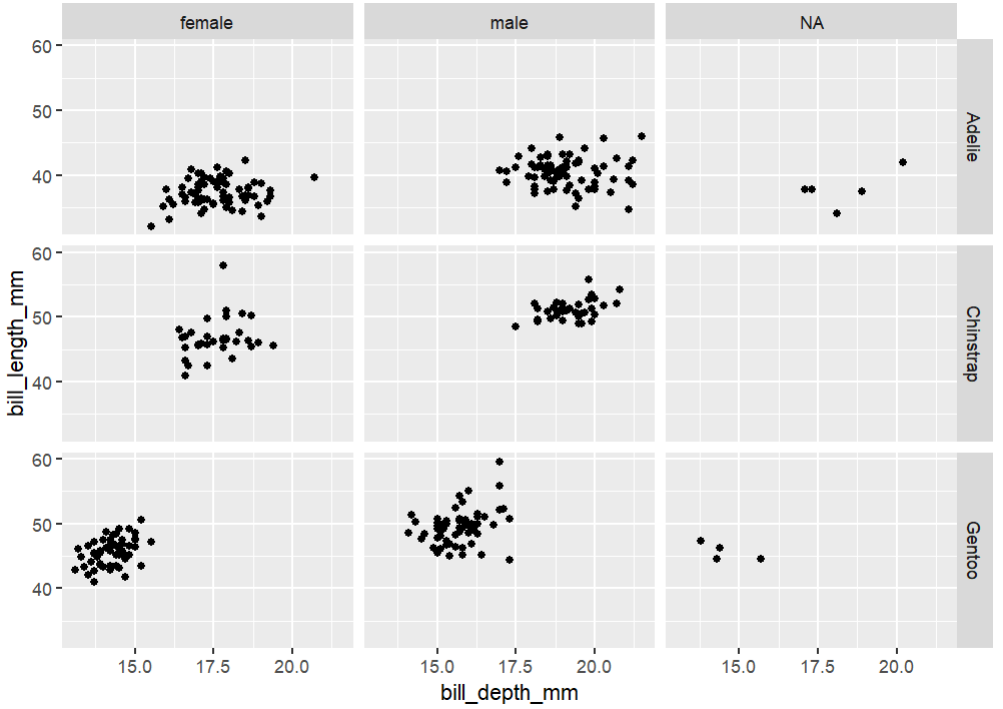
Overview | 2D grid 1 | 2D grid 2 | 1D grid 1 | 1D grid 2

- `facet_grid()`:
  - 2D grid
  - `rows ~ cols`
  - use `.` for no split (1D)
- Uses all levels, even if there are no observations; i.e., may produce empty plots

facet_grid()

Overview | 2D grid 1 | 2D grid 2 | 1D grid 1 | 1D grid 2

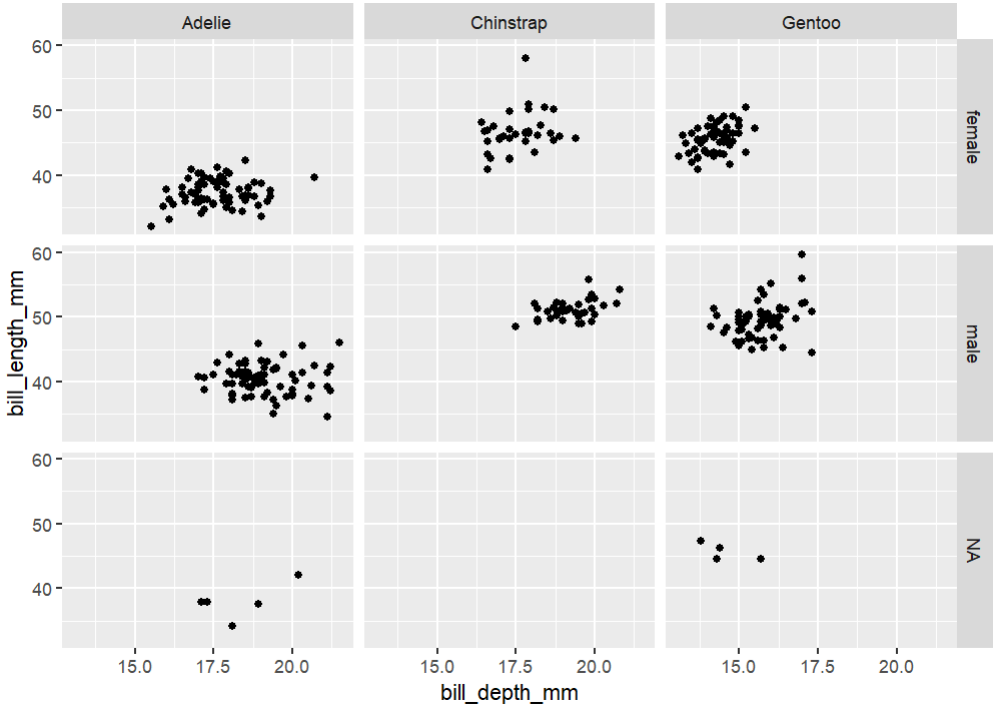
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(species ~ sex)
```



facet_grid()

Overview | 2D grid 1 | **2D grid 2** | 1D grid 1 | 1D grid 2

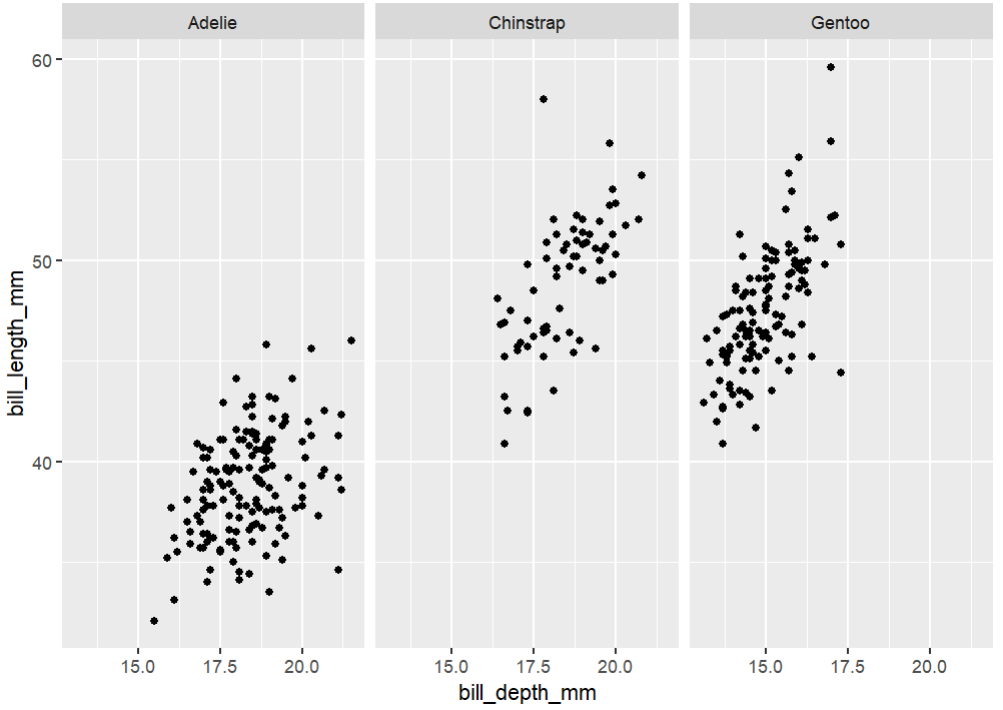
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(sex ~ species)
```



facet_grid()

Overview | 2D grid 1 | 2D grid 2 | **1D grid 1** | 1D grid 2

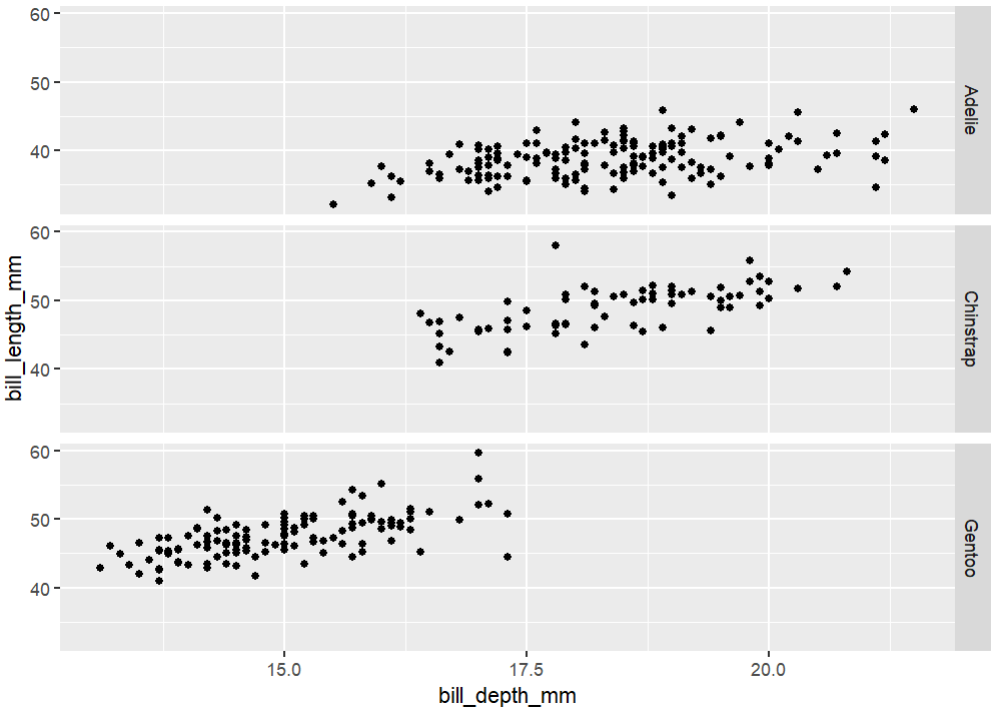
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(. ~ species)
```



facet_grid()

Overview | 2D grid 1 | 2D grid 2 | 1D grid 1 | **1D grid 2**

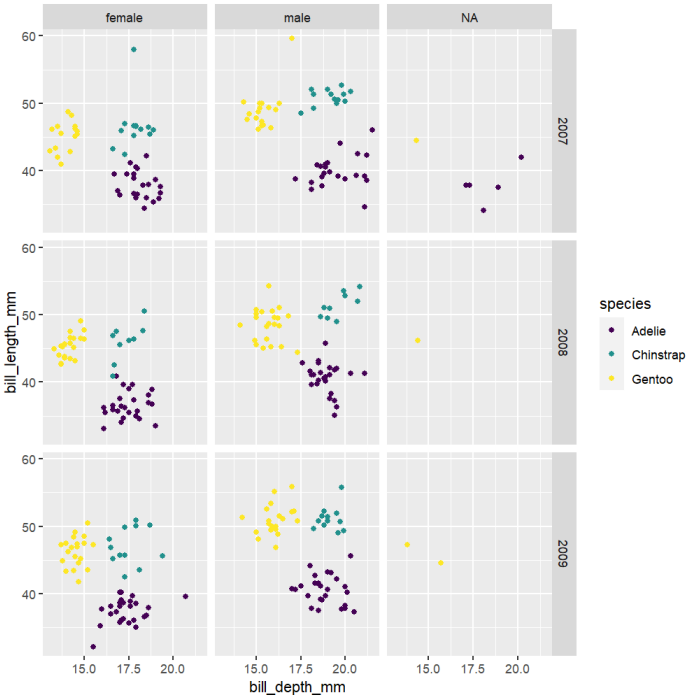
```
ggplot(penguins, aes(x = bill_depth_mm, y = bill_length_mm)) +
  geom_point() +
  facet_grid(species ~ .)
```



Facet can be used with color

- Perfect for complex comparison

```
ggplot(
  penguins,
  aes(x = bill_depth_mm,
      y = bill_length_mm,
      color = species)) +
  geom_point() +
  facet_grid(year ~ sex) +
  scale_color_viridis_d()
```



facet_wrap

Overview | Code | Plot

- To explore how the fertility against life_expectancy happened through the years, we can make the plot for several years.
- `facet_wrap` allows us to display multiple rows and columns of plots so that each has viewable dimensions.
- You can change the column numbers in `ncol=?`
- `facet_grid` 1D will make it too thin to show the data.
- The plot shows how most Asian countries have improved at a much faster rate than European ones.
- Default scale are fixed.

facet_wrap

Overview | Code | Plot

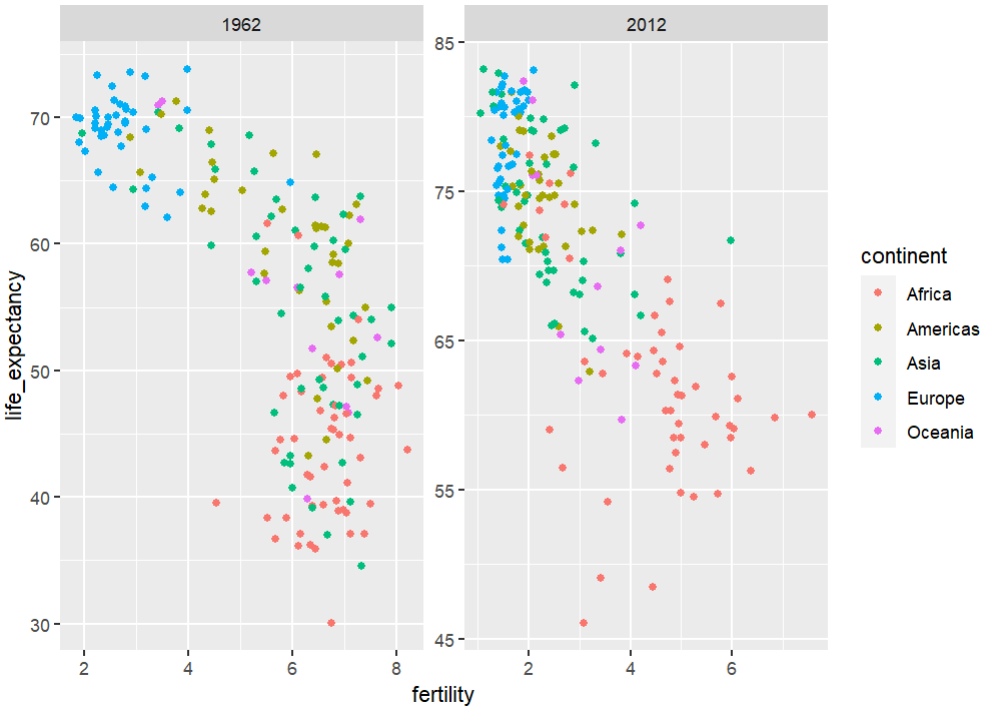
```
library(dslabs)
data(gapminder)
years <- c(1962,1970, 1980, 1990, 2000, 2012)
continents <- c("Europe", "Asia")
gapminder |>
  filter(year %in% years & continent %in% continents) |>
  ggplot( aes(fertility, life_expectancy, col = continent)) +
  geom_point() +
  facet_wrap(~year, ncol = 3)
```

facet_wrap

Overview Code Plot

Fixed scales or free scales

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



Time series plots

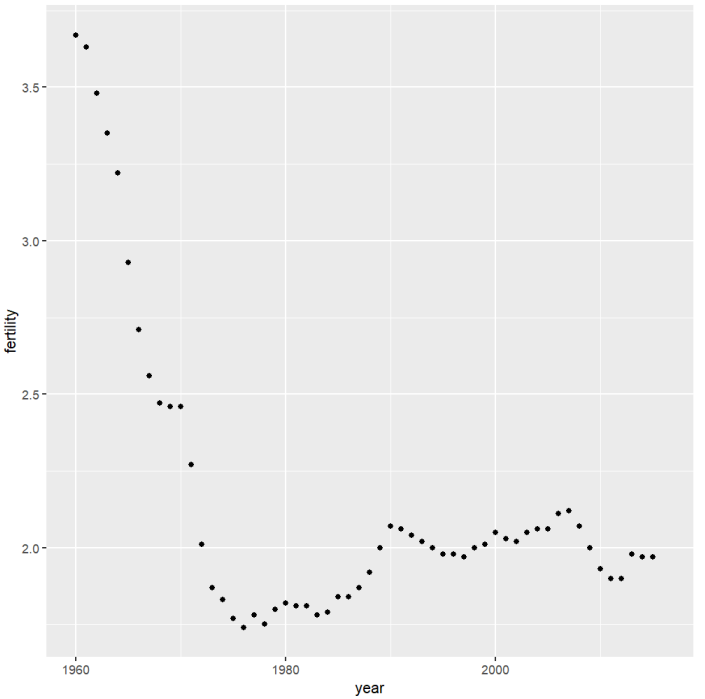
Overview | Points | Curve | Color

- Time series plots have time on the x-axis and an outcome of interest on the y-axis
- Effective for exploring temporal changes
- Use `geom_line()` to connect the points with lines and create curves for each series
- `color` aesethic assigns different colors to different series and groups the data automatically

Time series plots

Overview | **Points** | Curve | Color

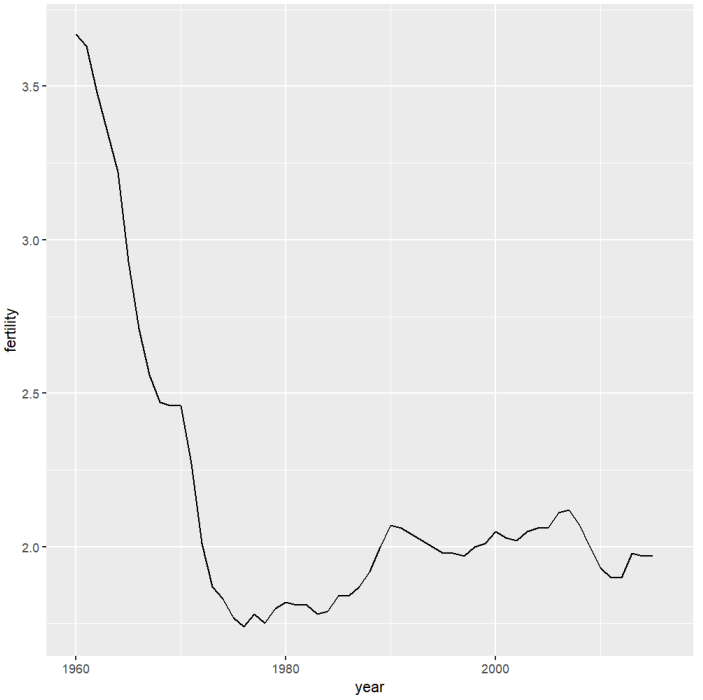
```
gapminder |>
  filter(country == "United States") |>
  ggplot(aes(year, fertility)) +
  geom_point()
```



Time series plots

Overview | Points | **Curve** | Color

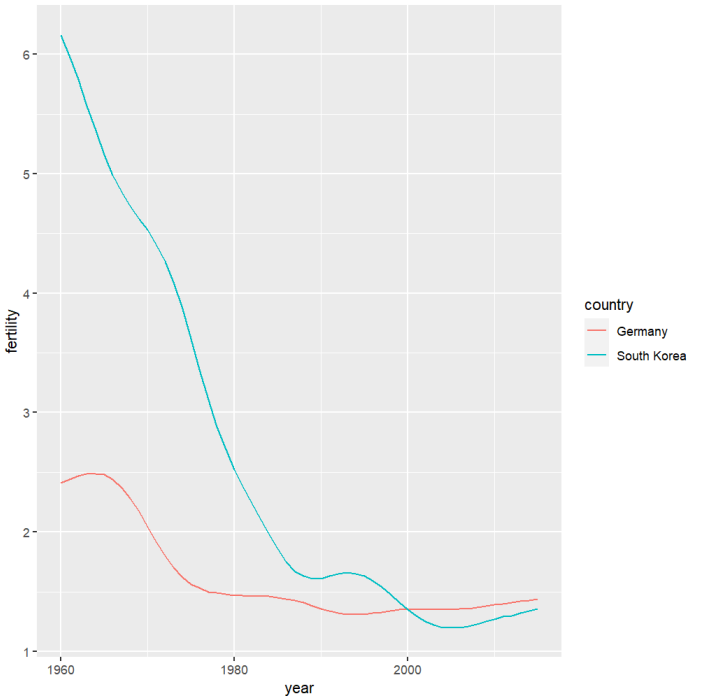
```
gapminder |>
  filter(country == "United States") |>
  ggplot(aes(year, fertility)) +
  geom_line()
```



Time series plots

Overview | Points | Curve | **Color**

```
countries <- c("South Korea","Germany")
gapminder |>
  filter(country %in% countries &
           !is.na(fertility)) |>
  ggplot(aes(year,fertility,
             col = country)) +
  geom_line()
```



Labels instead of legends

- We can use label instead of legends using the `geomtextpath` package.

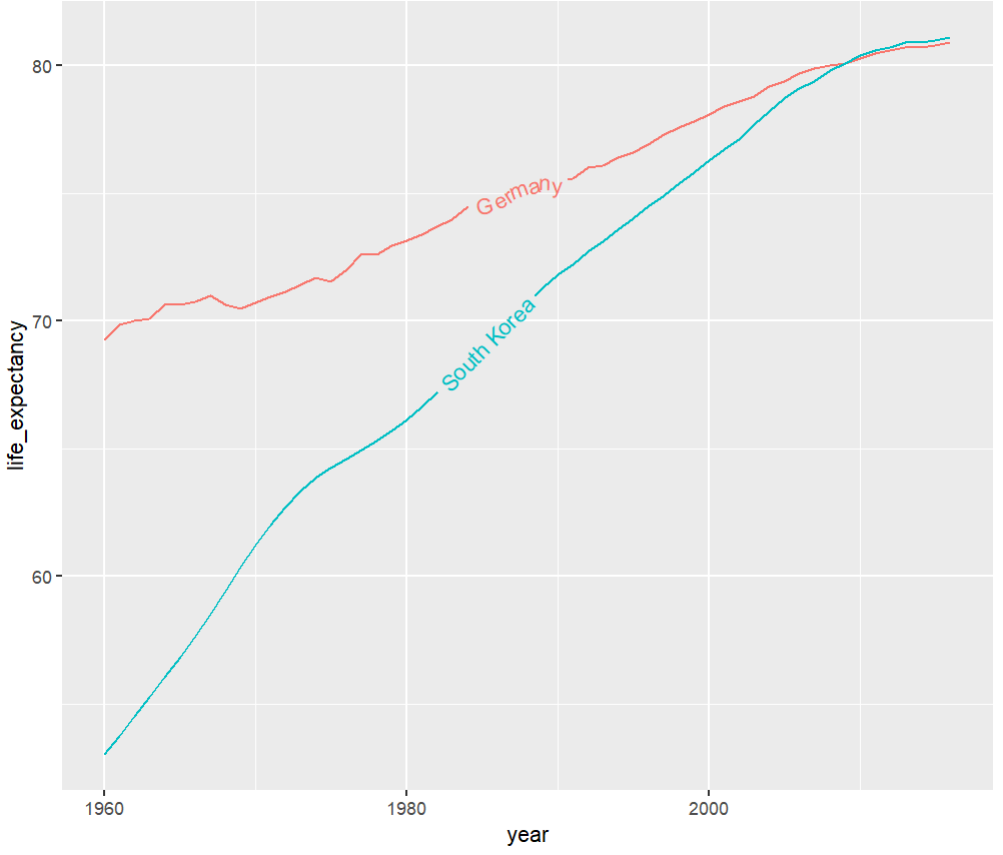
code | plot | words

```
library(geomtextpath)
gapminder |>
  filter(country %in% countries) |>
  ggplot(aes(year, life_expectancy,
             col = country, label = country)) +
  geom_textpath() +
  theme(legend.position = "none")
```

Labels instead of legends

- We can use label instead of legends using the `geomtextpath` package.

code | plot | words



Labels instead of legends

- We can use label instead of legends using the `geomtextpath` package.

code plot words

For trend plots we recommend labeling the lines rather than using legends since the viewer can quickly see which line is which country. This suggestion actually applies to most plots: labeling is usually preferred over legends.

We demonstrate how we can do this using the `geomtextpath` package. We define a data table with the label locations and then use a second mapping just for these labels:

The plot clearly shows how an improvement in life expectancy followed the drops in fertility rates. In 1960, Germans lived 15 years longer than South Koreans, although by 2010 the gap is completely closed. It exemplifies the improvement that many non-western countries have achieved in the last 40 years.

Descriptive statistics

- We've now learned about data manipulation and visualization tools
- What visualizations to do and what summary statistics to actually calculate?
- **Descriptive statistics** are numbers that are used to summarize and describe data
- **Numerical** or **graphical** ways to display the data
- Why is this a useful thing to do?

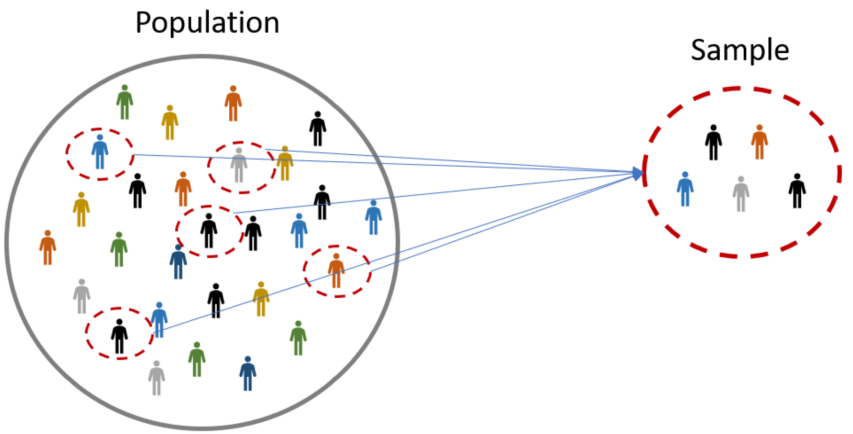
> Descriptive statistics are a useful tool in data analysis, as they help us understand and communicate the patterns and characteristics of a dataset.

Terminology of statistics

- A **subject**: A person, place or thing from which we collect data.
- A **population**: The collection of all subjects of interest.
- A **sample**: A subset of the population, from which we have collected data

  > Sample is a subgroup of population!
  >
  > **Sample size**: number of subjects in a sample

- Ideally, a sample should be **representative** of the population.



Terminology: continued

- A **variable**: A characteristic of a subject
- A **distribution** of a variable: the way the values of the variable are spread out or distributed over all possible values.
- **Univariate** data analysis: distribution of single variable
- **Bivariate** data analysis: relationship between two variables
- **Multivariate** data analysis: relationship between many variables at once, usually focusing on the relationship between two while conditioning for others

Terminology: Types of variables

- **Numerical** variables
  - E.g., age, length, temperature
  - **Continuous** variables can take on an infinite number of values
  - **Discrete** Numbers can only take on a *finite* or *countably infinite set* (Such as all integers) of possible values
- **Categorical** variables
  - E.g., year in college, type of bike, meal
  - **Ordinal** variables have levels that have a natural ordering
  - **Nominal** variables without any order

    > All genders, ethnicity, religions are equal!

Examples

murders | heights help page | heights | Small quiz | solution

For the `murders` dataset in `dslabs` package, we take a sample of 3 states:

```
library(dslabs)
head(murders, 3)
```

```
    state abb region population total
1 Alabama  AL  South    4779736   135
2  Alaska  AK   West     710231    19
3 Arizona  AZ   West    6392017   232
```

- **subject** is one US state.
- **population** is ALL US state.
- **sample** is Alabama, Alaska, Arizona for this small sample presented here.
- **variable** include: region, population, total (murders)
- abb, region are **categorical; nominal**
- population and total are **numeric; discrete**

Examples

murders | heights help page | heights | Small quiz | solution

For the `heights` dataset in `dslabs` package, we use help to get information of `heights` dataset

```
?heights
```

- Self-Reported Heights

> Description: Self-reported heights in inches for males and females.
>
> Usage: data(heights)
>
> Format: An object of class "data.frame".
>
> Details:sex. Male or Female.
>
> height. Height in inches.

Examples

murders | heights help page | heights | Small quiz | solution

```
head(heights, 3)
```

```
   sex height
1 Male     75
2 Male     70
3 Male     68
```

```
# Here we have 1050 observations
dim(heights)
```

```
[1] 1050    2
```

So for this dataset:

- **subject** is one student (possibly in some school).
- **population** is all students inside that school.
- **sample** is the 1050 students inside the dataset.

- **variable** include: sex, height

- sex is **categorical; nominal**

Examples

murders | heights help page | heights | Small quiz | solution

What are the types of these variables?

- Number of people in each class
- Letter grades
- Shape of leaf
- Zip code: (95618 for Davis, 95776 for Woodland)
- Heights of newborn babies

Examples

murders | heights help page | heights | Small quiz | solution

- Number of people in each class **discrete**
- Letter grades **ordinal**
- Shape of leaf **nominal**
- Zip code: (95618 for Davis, 95776 for Woodland) **nominal**
- Heights of newborn babies **continuous**

Data: Lending Club

- Lending Club is a platform that allows individuals to lend to other individuals
- Data are available in the `openintro` package, called `loans_full_schema`
- Includes 10,000 loans made through the Lending Club; has 55 columns

```
library(openintro)
dplyr::glimpse(loans_full_schema)
```

```
Rows: 10,000
Columns: 55
$ emp_title                     <chr> "global config engineer ", "warehouse…
$ emp_length                    <dbl> 3, 10, 3, 1, 10, NA, 10, 10, 10, 3, 1…
$ state                         <fct> NJ, HI, WI, PA, CA, KY, MI, AZ, NV, I…
$ homeownership                 <fct> MORTGAGE, RENT, RENT, RENT, RENT, OWN…
$ annual_income                 <dbl> 90000, 40000, 40000, 30000, 35000, 34…
$ verified_income               <fct> Verified, Not Verified, Source Verifi…
$ debt_to_income                <dbl> 18.01, 5.04, 21.15, 10.16, 57.96, 6.4…
$ annual_income_joint           <dbl> NA, NA, NA, NA, 57000, NA, 155000, NA…
$ verification_income_joint     <fct> , , , , Verified, , Not Verified, , ,…
$ debt_to_income_joint          <dbl> NA, NA, NA, NA, 37.66, NA, 13.12, NA,…
$ delinq_2y                     <int> 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0…
$ months_since_last_delinq      <int> 38, NA, 28, NA, NA, 3, NA, 19, 18, NA…
$ earliest_credit_line          <dbl> 2001, 1996, 2006, 2007, 2008, 1990, 2…
$ inquiries_last_12m            <int> 6, 1, 4, 0, 7, 6, 1, 1, 3, 0, 4, 4, 8…
$ total_credit_lines            <int> 28, 30, 31, 4, 22, 32, 12, 30, 35, 9,…
$ open_credit_lines             <int> 10, 14, 10, 4, 16, 12, 10, 15, 21, 6,…
$ total_credit_limit            <int> 70795, 28800, 24193, 25400, 69839, 42…
$ total_credit_utilized         <int> 38767, 4321, 16000, 4997, 52722, 3898…
$ num_collections_last_12m      <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0…
$ num_historical_failed_to_pay  <int> 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0…
$ months_since_90d_late         <int> 38, NA, 28, NA, NA, 60, NA, 71, 18, N…
```

Selected variables

```
loans <- loans_full_schema %>%
  select(loan_amount, interest_rate, term, grade,
         state, annual_income, homeownership, debt_to_income,
         issue_month)
glimpse(loans)
```

```
Rows: 10,000
Columns: 9
$ loan_amount    <int> 28000, 5000, 2000, 21600, 23000, 5000, 24000, 20000, 2
$ interest_rate  <dbl> 14.07, 12.61, 17.09, 6.72, 14.07, 6.72, 13.59, 11.99,
$ term           <dbl> 60, 36, 36, 36, 36, 36, 60, 60, 36, 36, 60, 60, 36, 60
$ grade          <fct> C, C, D, A, C, A, C, B, C, A, C, B, C, B, D, D, D, F,
$ state          <fct> NJ, HI, WI, PA, CA, KY, MI, AZ, NV, IL, IL, FL, SC, CO
$ annual_income  <dbl> 90000, 40000, 40000, 30000, 35000, 34000, 35000, 11000
$ homeownership  <fct> MORTGAGE, RENT, RENT, RENT, RENT, OWN, MORTGAGE, MORTG
$ debt_to_income <dbl> 18.01, 5.04, 21.15, 10.16, 57.96, 6.46, 23.66, 16.19,
$ issue_month    <fct> Mar-2018, Feb-2018, Feb-2018, Jan-2018, Mar-2018, Jan-
```

Selected variables

Variable	Description
`loan_amount`	Amount of the loan received, in US dollars
`interest_rate`	Interest rate on the loan, in an annual percentage
`term`	The length of the loan, which is always set as a whole number of months
`grade`	Loan grade, which takes a values A through G and represents the quality of the loan and its likelihood of being repaid
`state`	US state where the borrower resides
`annual_income`	Borrower's annual income, including any second income, in US dollars
`homeownership`	Indicates whether the person owns, owns but has a mortgage, or rents
`debt_to_income`	Debt-to-income ratio
`issue_month`	Month the loan was issued

Variable types

Variable	Description
`loan_amount`	Amount of the loan received, in US dollars
`interest_rate`	Interest rate on the loan, in an annual percentage
`term`	The length of the loan, which is always set as a whole number of months
`grade`	Loan grade, which takes a values A through G and represents the quality of the loan and its likelihood of being repaid
`state`	US state where the borrower resides
`annual_income`	Borrower's annual income, including any second income, in US dollars
`homeownership`	Indicates whether the person owns, owns but has a mortgage, or rents
`debt_to_income`	Debt-to-income ratio
`issue_month`	Month the loan was issued

- Numerical variables: Continuous or discrete?
- Categorical: Ordinal or not?

Variable types

Variable	Type
loan_amount	numerical, continuous
interest_rate	numerical, continuous
term	numerical, discrete
grade	categorical, ordinal
state	categorical, not ordinal
annual_income	numerical, continuous
homeownership	categorical, not ordinal
debt_to_income	numerical, continuous
issue_month	date

Following lectures: Describing numerical distributions

- **Visual summaries**:
  - Histogram
  - Boxplot
  - Density plot
  - Line graph
- Measures of **central tendency**: mean, median, mode
- **Shape**:
  - Skewness: right-skewed, left-skewed, symmetric
  - Modality: unimodal, bimodal, multimodal, uniform
- Measures of **Spread**: variance and standard deviation, range and interquartile range (IQR)
- **Unusual observations**
- A **summary statistic** is a single number summarizing a large amount of data

Readings

- Chapter 10:Data visualization in practice
- Open Intro Statistics Chapter 1