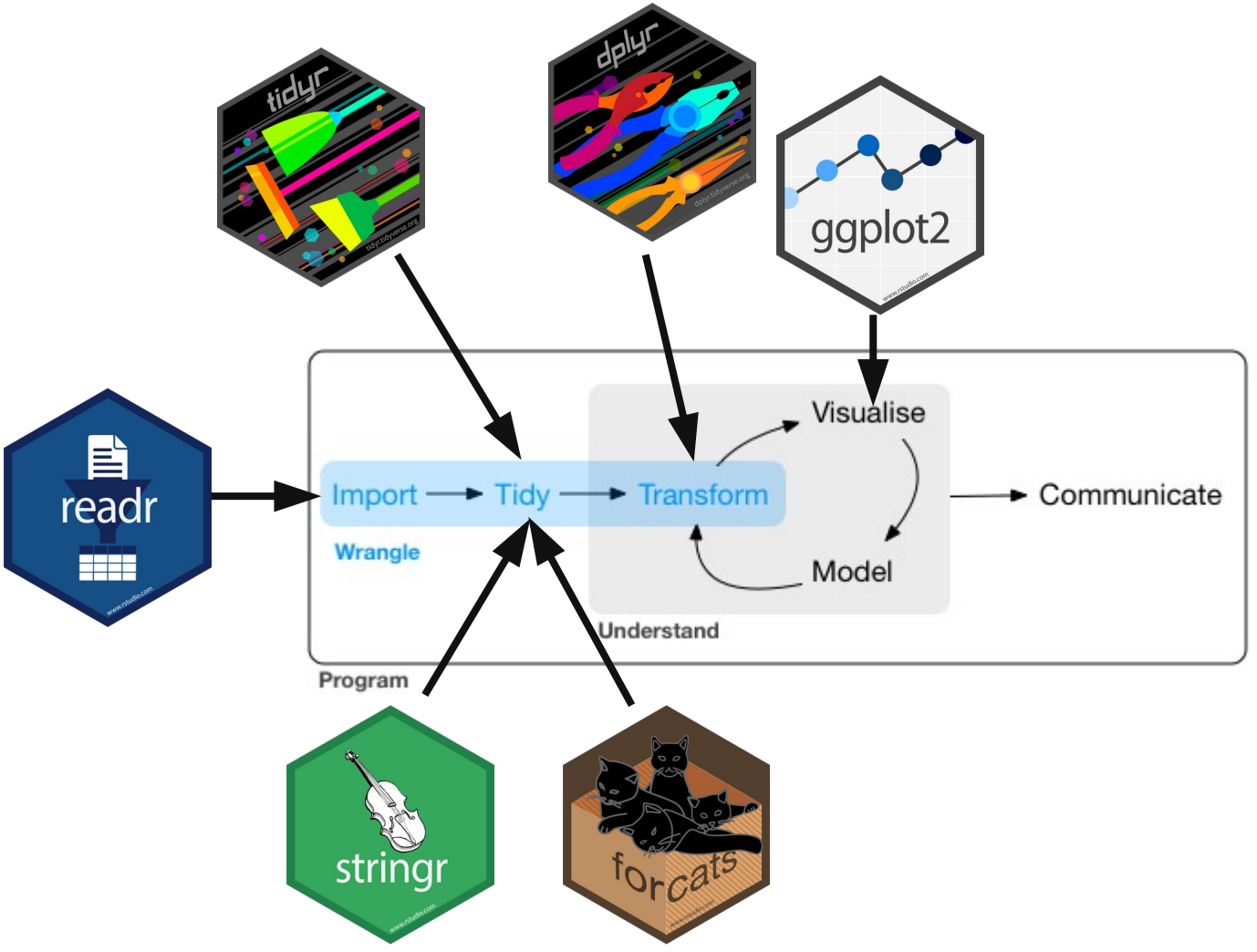
GG606

Exploration & Data Types

Homework

- Pick a year: https://doi.org/10.5683/SP3/OUWVZ3 (physical, chemical, biological)
- Use this: https://doi.org/10.5683/SP2/TNYTQL
  “NW-20-C2-Chronology-Dspec50-2019-with-self-attenuation-SimpleView.xlsx”
  “NW-50-Chronology-Dspec649-2019-withdensity-SIMPLEVIEW with graphs v3.tab”

Homework


tidyr
dplyr
ggplot2
readr
Import
Tidy
Transform
Visualise
Model
Communicate
Wrangle
Understand
Program
stringr
forcats

Transformations

- Look in the `nycflights13::flights` tibble
- Many data types (`dttm`)
- Factor `fctr` is not there but is v useful

dplyr

- “grammar of data manipulation”
- `mutate()` adds new variables that are functions of existing variables
- `select()` picks variables based on their names
- `filter()` picks cases based on their values
- `summarise()` reduces multiple values down to a single summary
- `arrange()` changes the ordering of the rows
- `group_by()` allows you to perform any operation “by group”

Python: pandas, dplython, siuba

dplyr

- `filter(flights, month == 1, day == 1)`
  - data.frame/tibble, expression(s)
- `filter(flights, month = 1)`
  - Why the error?
- Try more complex

dplyr

- `filter(starwars, species == "Human")`

dplyr

- `filter(flights, month == 11 | month == 12)`
- Logical operators

dplyr

- `filter(flights, !(arr_delay > 120 | dep_delay > 120))`
- `filter(flights, arr_delay ⩽ 120, dep_delay ⩽ 120)`

dplyr

- Find all flights that
  - Flew to Houston (IAH or HOU)
  - Were operated by United, American, or Delta
  - From JFK to LAX on Christmas Day

Order rows

- `arrange()` orders the rows of a data frame by the values of selected columns
- `desc()` sorted in descending order
- `arrange(flights, desc(dep_delay))`

Select Columns

- `select()` columns by name or range
- `select(flights, year, month, day)`
- `select(flights, year:day)`
- `select(flights, 1:3)`
- `select(flights, -(month:minute))`

Select + Helpers

- `starts_with("abc")`
- `ends_with("xyz")`
- `contains("ijk")`
- `matches("(.)\\1")`
- `?select()`

Mutate

- ```
  flights_sml ← select(flights,
    year:day,
    ends_with("delay"),
    distance,
    air_time
  )
  ```
- ```
  mutate(flights_sml,
    gain = dep_delay - arr_delay,
    speed = distance / air_time * 60
  )
  ```

Transmute

- ```
  flights_sml ← select(flights,
    year:day,
    ends_with("delay"),
    distance,
    air_time
  )
  ```
- ```
  transmute(flights_sml,
    gain = dep_delay - arr_delay,
    speed = distance / air_time * 60
  )
  ```

Mutate/Transmute

- Arithmetic
- Functions
- Vector-based
- Logical
- Rank
- Cumulative

Grouped Summaries

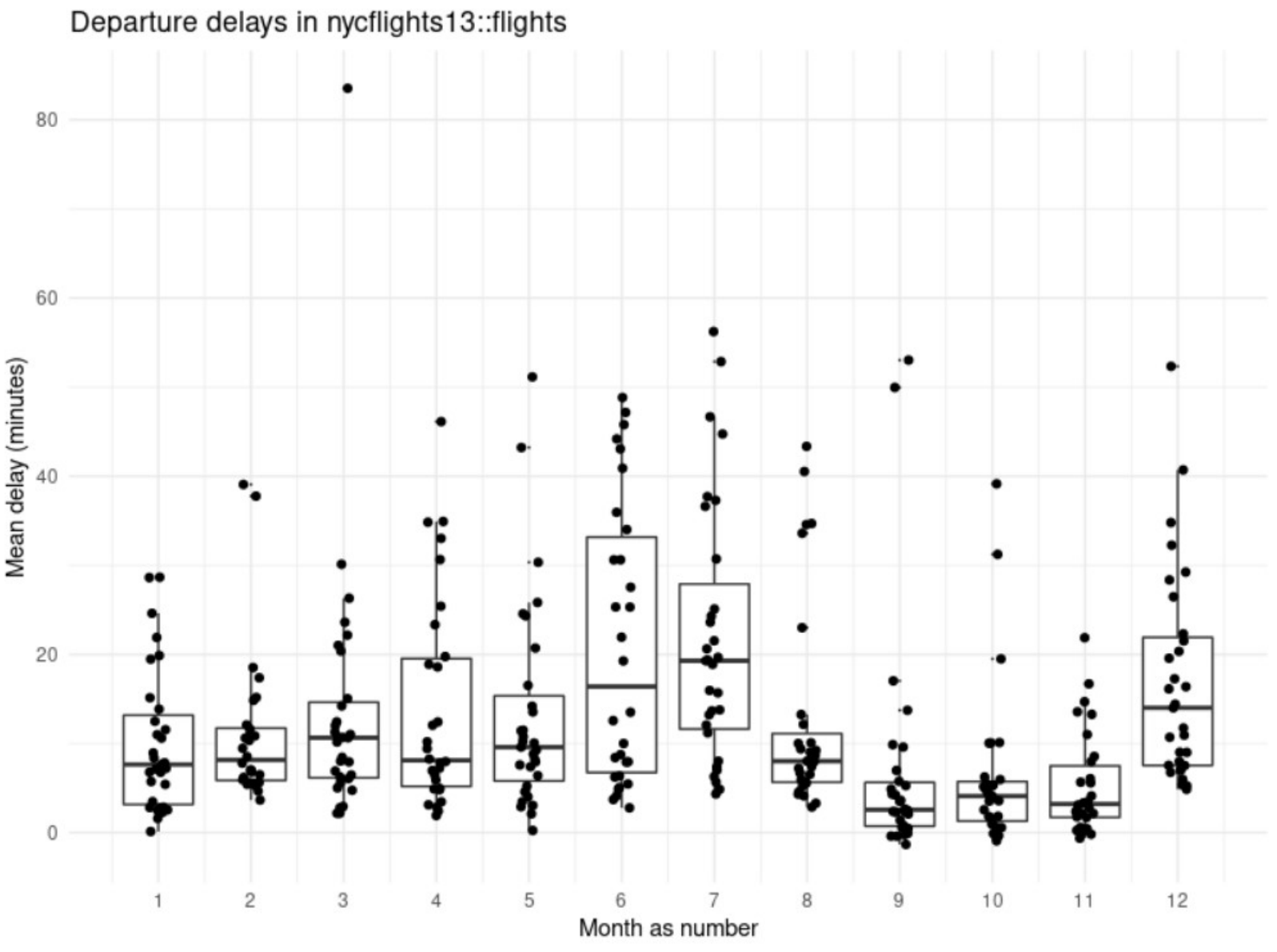
- Collapses data.frame to a single row
  - `summarise(flights, delay = mean(dep_delay, na.rm = TRUE))`
- More powerful when grouped

```
> summarise(flights, delay = mean(dep_delay, na.rm = TRUE))
# A tibble: 1 x 1
  delay
  <dbl>
1  12.6
```

Grouped Summaries

- `by_day ← group_by(flights, year, month, day)`
- `summarise(by_day, delay = mean(dep_delay, na.rm = TRUE))`

```
> by_day ← group_by(flights, year, month, day)
> summarise(by_day, delay = mean(dep_delay, na.rm = TRUE))
`summarise()` regrouping output by 'year', 'month' (override with `.groups` argument)
# A tibble: 365 x 4
# Groups:   year, month [12]
    year month   day delay
   <int> <int> <int> <dbl>
 1  2013     1     1 11.5
 2  2013     1     2 13.9
```


Departure delays in nycflights13::flights
Mean delay (minutes)
0
20
40
60
80
1
2
3
4
5
6
7
8
9
10
11
12
Month as number

Multiple Operations

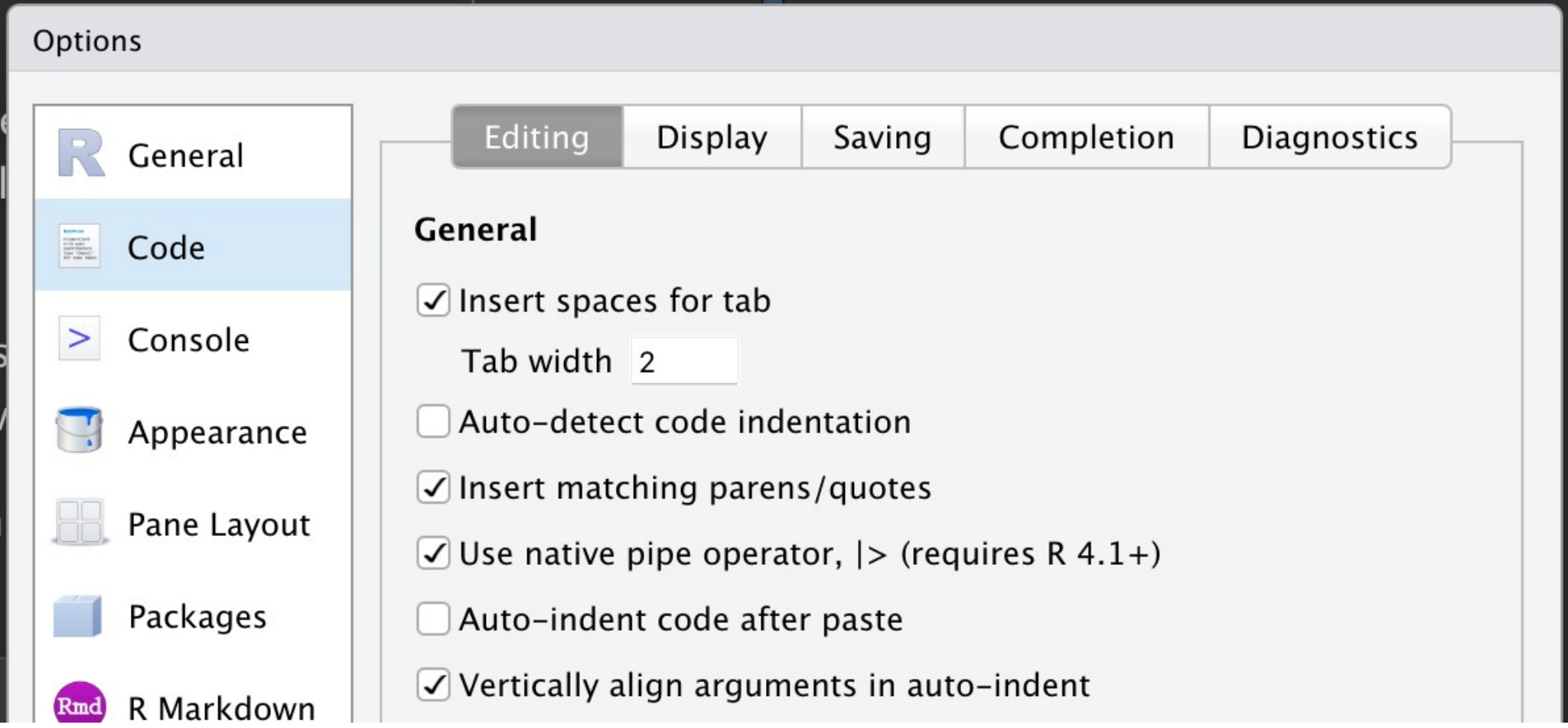
- The `magrittr` pipe to chain operation (improved semantics) %>%

5.3 |> vs. %>%

While |> and %>% behave identically for simple cases, there are a few crucial differences. These are most likely to affect you if you're a long-term user of %>% who has taken advantage of some of the more advanced features. But they're still good to know about even if you've never used %>% because you're likely to encounter some of them when reading wild-caught code.

- By default, the pipe passes the object on its left-hand side to the first argument of the function on the right-hand side. %>% allows you to change the placement with a . placeholder. For example, x %>% f(1) is equivalent to f(x, 1) but x %>% f(1, .) is equivalent to f(1, x). R 4.2.0 added a _ placeholder to the base pipe, with one additional restriction: the argument has to be named. For example, x |> f(1, y = _) is equivalent to f(1, y = x).

New text:
https://r4ds.hadley.nz/data-transform.html#sec-the-pipe



New text:
https://r4ds.hadley.nz/data-transform.html#sec-the-pipe

magrittr

If you've been using the tidyverse for a while, you might be familiar with the `%>%` pipe provided by the **magrittr** package. The magrittr package is included in the core tidyverse, so you can use `%>%` whenever you load the tidyverse:

```
library(tidyverse)

mtcars %>%
  group_by(cyl) %>%
  summarize(n = n())
```

For simple cases, `|>` and `%>%` behave identically. So why do we recommend the base pipe? Firstly, because it's part of base R, it's always available for you to use, even when you're not using the tidyverse. Secondly, `|>` is quite a bit simpler than `%>%`: in the time between the invention of `%>%` in 2014 and the inclusion of `|>` in R 4.1.0 in 2021, we gained a better understanding of the pipe. This allowed the base implementation to jettison infrequently used and less important features.

New text:
https://r4ds.hadley.nz/data-transform.html#sec-the-pipe

Pipe Example

- Q: relationship btwn distance and avg delay for each location
  - Group flights by destination.
  - Summarise to compute distance, average delay, and number of flights.
  - Filter to remove noisy points and Honolulu airport, which is almost twice as far away as the next closest airport.

Pipe Example

```
delays ← flights %>%
  group_by(dest) %>%
  summarise(
    count = n(),
    dist = mean(distance, na.rm = TRUE),
    delay = mean(arr_delay, na.rm = TRUE)
  ) %>%
  filter(count > 20, dest ≠ "HNL")
```

Tidy

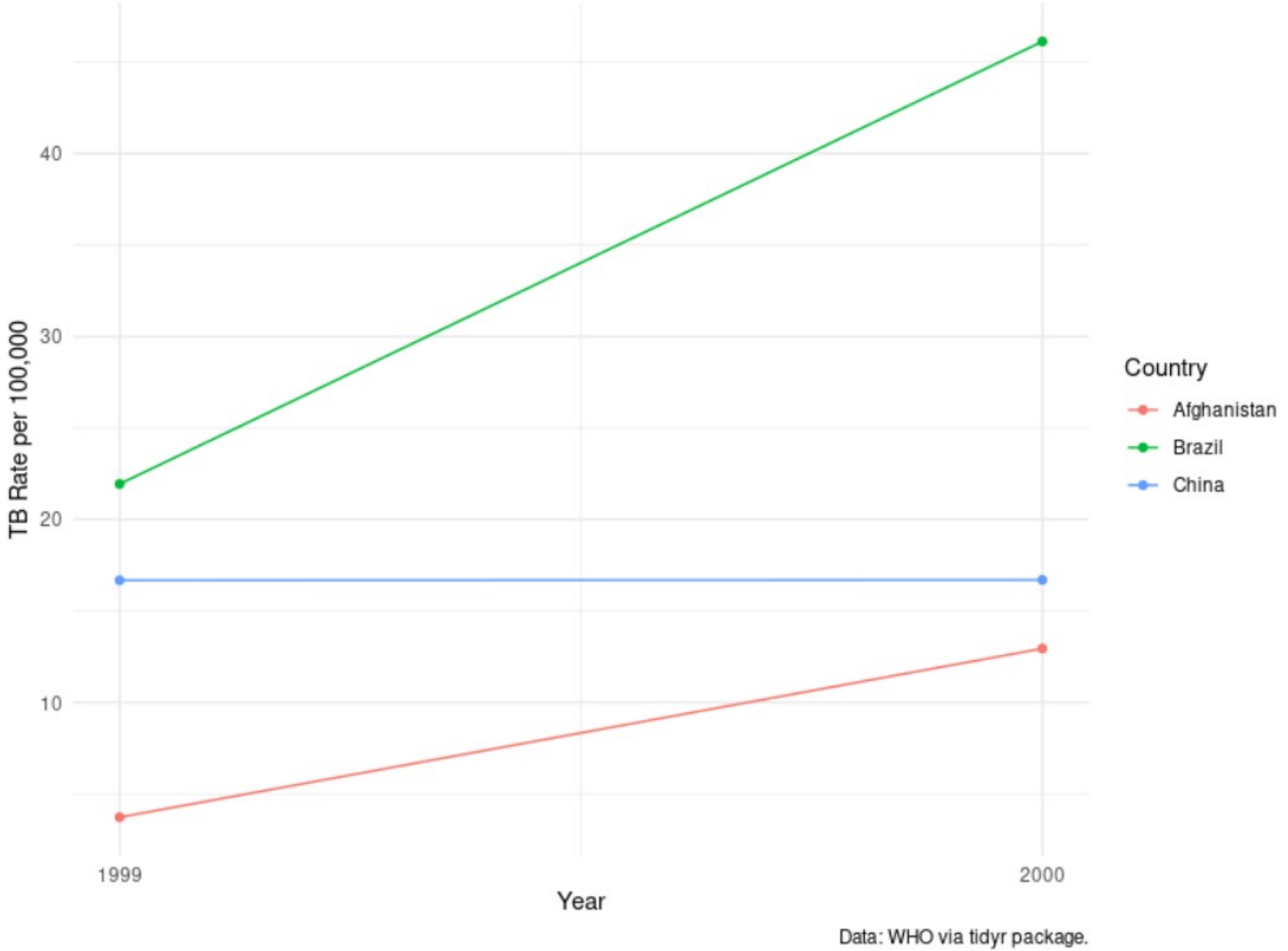
- Examples of tidy data: TB cases
- `table1`
- `table2`
- `table3`
- `table4a`
- `table4b`

```
table1
#> # A tibble: 6 x 4
#>   country      year  cases population
#>   <chr>       <int>  <int>      <int>
#> 1 Afghanistan  1999    745   19987071
#> 2 Afghanistan  2000   2666   20595360
#> 3 Brazil       1999  37737  172006362
#> 4 Brazil       2000  80488  174504898
#> 5 China        1999 212258 1272915272
#> 6 China        2000 213766 1280428583
table2
#> # A tibble: 12 x 4
#>   country      year type           count
#>   <chr>       <int> <chr>          <int>
#> 1 Afghanistan  1999 cases            745
#> 2 Afghanistan  1999 population  19987071
#> 3 Afghanistan  2000 cases           2666
#> 4 Afghanistan  2000 population  20595360
#> 5 Brazil       1999 cases          37737
#> 6 Brazil       1999 population 172006362
#> # … with 6 more rows
```

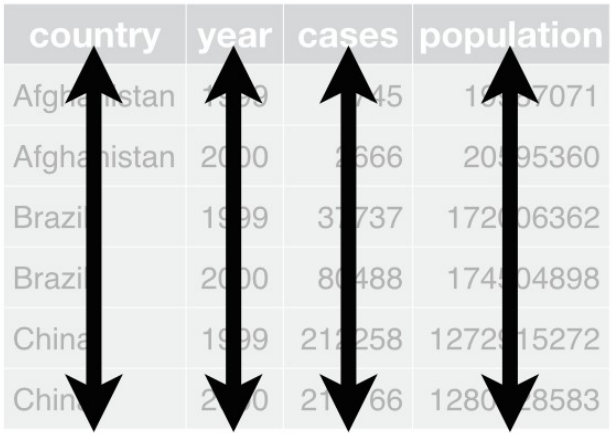
https://r4ds.hadley.nz/data-tidy.html

Plot Rate per 100,000

- Start with `table1`
- Calc rate
- Make plot


40
30
20
10
TB Rate per 100,000
1999
2000
Year
Country
Afghanistan
Brazil
China
Data: WHO via tidyr package.

Tidying



variables

observations

values

Pivoting

- Not PivotTable
- Data often organised to facilitate entry or quick visualisations

Longer

- Take wide, make long
- ```
  table4a %>%
    pivot_longer(c(`1999`, `2000`),
    names_to = "year", values_to = "cases")
  ```


  ```

Joining

- `tidy4a ← table4a %>% pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "cases")`
- `tidy4b ← table4b %>% pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "population")`
- `left_join(tidy4a, tidy4b)`

```
#> Joining, by = c("country", "year")
#> # A tibble: 6 x 4
#>   country     year   cases population
#>   <chr>       <chr>  <int>      <int>
#> 1 Afghanistan 1999     745   19987071
#> 2 Afghanistan 2000    2666   20595360
#> 3 Brazil      1999   37737  172006362
#> 4 Brazil      2000   80488  174504898
#> 5 China       1999  212258 1272915272
#> 6 China       2000  213766 1280428583
```

Separate & Unite

- Take long, make wide
- ```
  table3 %>%
    separate()
  ```
- ```
  table5 %>%
    unite()
  ```

Homework

- Combine `nycflights13::flights` & `nycflights13::weather`
  - Identify primary keys
  - Put R-Script and figure in discord