Taming and Tidying your Data

Class #9 | GEOG 215

Intro To Spatial Data Science

Today's Class

The Fun Part

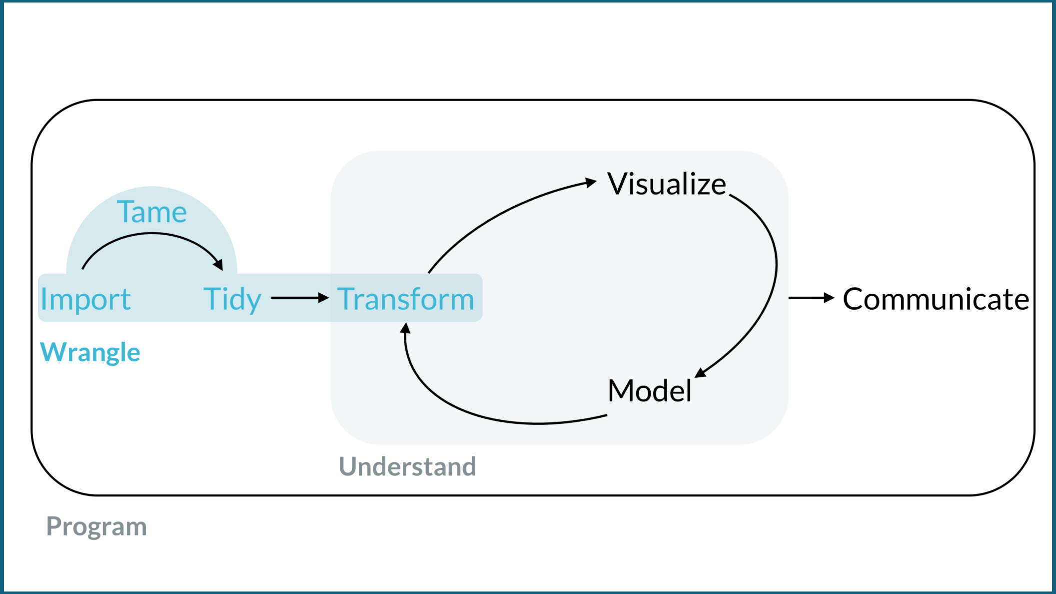
Today's Class

The Fun Part (Almost)

- ## Taming --> Tidying --> Transforming

Next class

- Visualize
- Explore
- Repeat


Tame
Import
Tidy
Transform
Visualize
Model
Communicate
Wrangle
Understand
Program

Taming Your Data

Parsing/Casting your columns

- **Making sure data is in the correct format**
    - Categories are factors/character
    - Quantititative variables are numeric
    - Dates are dates
- **Commands from `readr` package**
    - parse eg. `parse_number()`
    - casting eg. `col_number()`

Taming Your Data

Recoding Values

- **Making sure values in columns are correct**
  - eg. Yes = 1, No = 0
- **Switch from continuous to discrete**
  - eg. Changing Income values to high,medium,low
- **Useful to create dummy variables (0,1) (absence/presence)**
- **Commands from `dplyr` package**
  - parse eg. `recode` to factor using `recode_factor()`
  - frequently within `mutate`

Taming Your Data

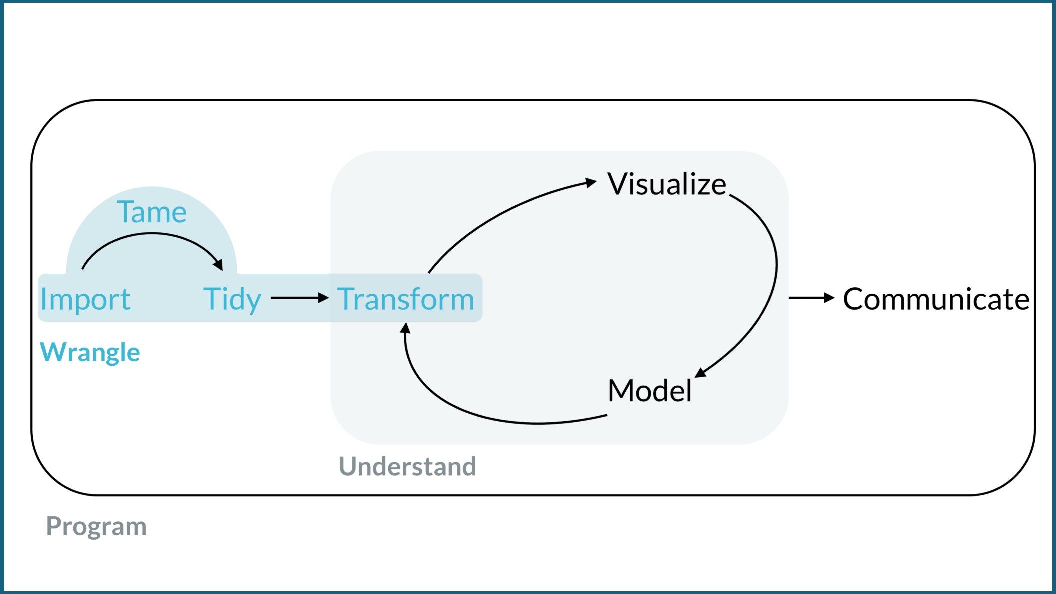
Selecting columns

- **Making sure only relevant columns are included in dataset**
  - eg. drop irrelevant/intermediate columns
- **Make sure columns are in correct order**
  - eg. Eg - all grouping columns together, all thematic columns together
- **Useful to create dummy variables (0,1) (absence/presence)**
- **Commands from `dplyr` package**
  - select eg. `select` function
  - reorder variables using `select` and helper functions

Taming Your Data

Reformatting and Renaming Variable Names

- **Makes sure variable names make sense**
  - eg. Total cases of disease vs percent of population with disease is reflected in column names
- **variable names are consistent**
  - eg. No foreign characters, consistent cases, no spaces etc
- **Commands from `dplyr` and `janitor` package**
  - clean variable names using `clean_names` from `janitor` package
  - rename using `rename` from `dplyr`. often used with `select` for reordering and keeping new variables



tame data ≠ **tidy data**

Tidy Data

> “Happy families are all alike; every unhappy family is unhappy in its own way.”
>
> -- Leo Tolstoy
>
> “Tidy datasets are all alike, but every messy dataset is messy in its own way.”
>
> -- Hadley Wickham (inventor of Tidyverse)
>
> ## Three Cardinal Rules of a tidy dataset
>
> - Each variable must be its own column
> - Each observation must have its own row
> - Each value must have its own cell

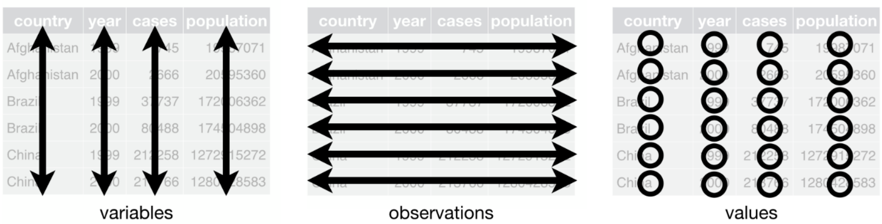


Figure 12.1: Following three rules makes a dataset tidy: variables are in columns, observations are in rows, and values are in cells.

Put each dataset in a tibble (or data frame)

Put each variable in a column

Which one out of these is tidy:

```
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
```

```
table3
#> # A tibble: 6 x 3
#>   country      year rate
#> * <chr>       <int> <chr>
#> 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
```

```
table4a  # cases
#> # A tibble: 3 x 3
#>   country     `1999` `2000`
#> * <chr>        <int>  <int>
#> 1 Afghanistan    745   2666
#> 2 Brazil       37737  80488
#> 3 China       212258 213766
table4b  # population
#> # A tibble: 3 x 3
#>   country         `1999`     `2000`
#> * <chr>            <int>      <int>
#> 1 Afghanistan   19987071   20595360
#> 2 Brazil       172006362  174504898
#> 3 China       1272915272 1280428583
```

Making data Tidy

`pivot_longer()`

- Wide format to long format
- succeeds `gather()`

```r
table4a %>%
  pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "cases")
```

Figure 12.2: Pivoting `table4` into a longer, tidy form.

Making data Tidy

`pivot_wider()`

- long format to wide format
- succeeds `spread()`

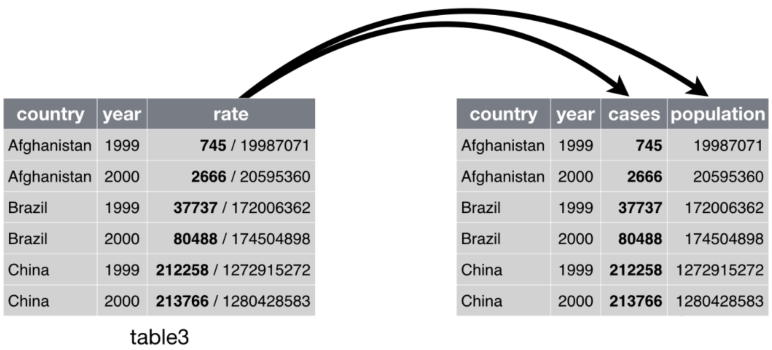
```
table4a %>%
  pivot_longer(c(`1999`, `2000`), names_to = "year", values_to = "cases")
```

Making data Tidy

`separate()`

- *break up single* column to *multiple* columns
- To ensure that each value is its *own* cell

```
table3 %>%
  separate(rate, into = c("cases", "population"), convert = TRUE)
```



country	year	rate
Afghanistan	1999	**745** / 19987071
Afghanistan	2000	**2666** / 20595360
Brazil	1999	**37737** / 172006362
Brazil	2000	**80488** / 174504898
China	1999	**212258** / 1272915272
China	2000	**213766** / 1280428583

table3

country	year	cases	population
Afghanistan	1999	**745**	19987071
Afghanistan	2000	**2666**	20595360
Brazil	1999	**37737**	172006362
Brazil	2000	**80488**	174504898
China	1999	**212258**	1272915272
China	2000	**213766**	1280428583

Figure 12.4: Separating `table3` makes it tidy

Another untidy table

```
#> # A tibble: 6 x 4
#>   country     century year  rate
#>   <chr>       <chr>   <chr> <chr>
#> 1 Afghanistan 19      99    745/19987071
#> 2 Afghanistan 20      00    2666/20595360
#> 3 Brazil      19      99    37737/172006362
#> 4 Brazil      20      00    80488/174504898
#> 5 China       19      99    212258/1272915272
#> 6 China       20      00    213766/1280428583
```

Making data Tidy

`unite()`

- *Combines multiple* columns into a *single* column
- To ensure that each value is its *own* cell

```
table5 %>%
  unite(new, century, year, sep = "")
```

country	year	rate
Afghanistan	1999	745 / 19987071
Afghanistan	2000	2666 / 20595360
Brazil	1999	37737 / 172006362
Brazil	2000	80488 / 174504898
China	1999	212258 / 1272915272
China	2000	213766 / 1280428583

country	century	year	rate
Afghanistan	19	99	745 / 19987071
Afghanistan	20	0	2666 / 20595360
Brazil	19	99	37737 / 172006362
Brazil	20	0	80488 / 174504898
China	19	99	212258 / 1272915272
China	20	0	213766 / 1280428583

table6

Figure 12.5: Uniting `table5` makes it tidy

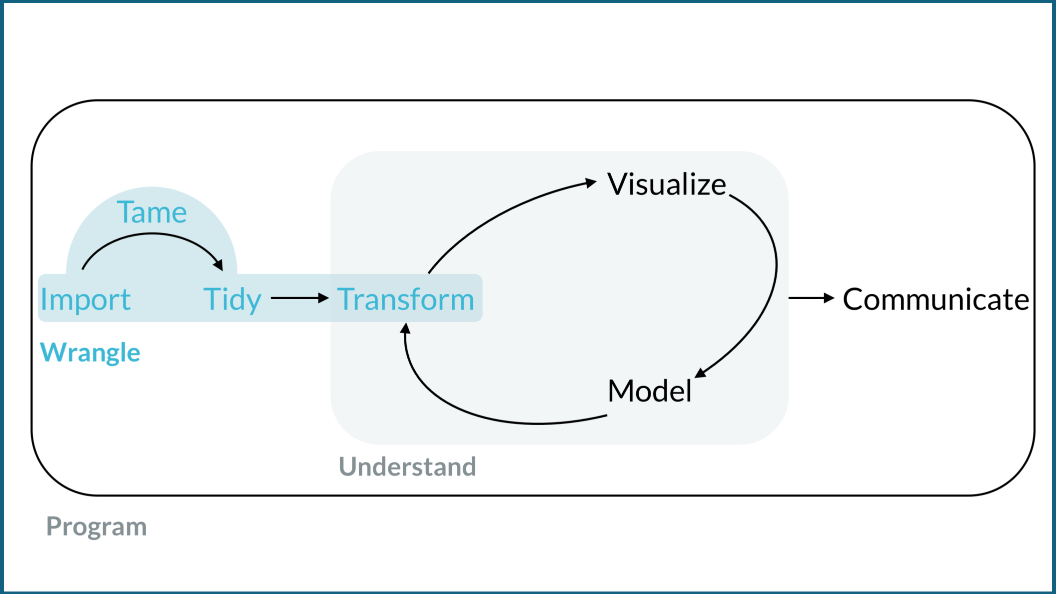
Tidy Data Tips

Tidy data is the start of your data wrangling journey, not the end

- There is not a single “tidy” version of a dataset

Not all non-tidy is incorrect, bad, or not-useful

- May have better space or performance advantages
  - Eg. Big issue with spatial data (sometimes)
- Some fields/data have their own useful conventions
- All data can be fit in rectangular structures
  - genomic data
  - Corpus of texts
    - Network/graph datasets



TRANSFORMING DATA

The `famous 5` verbs of `dplyr`

- `arrange`
- `select`
- `filter`
- `mutate`
- `summarize`

Other important transformation variables

- **group_by() , ungroup**
    - often used with the famous 5
- **join commands**
    - combining multiple datasets/tables

Use cheatsheets often

https://rstudio.com/resources/cheatsheets/

Next Class

- **Data Visualization (spatial and non-spatial)**
  - email a few visualizations, we will scrutinize them
- **Lab 3/HW 1 doubts**
- **Fill in polleverywhere Area of interest survey (LAST CHANCE)**