Tidy exercises in class

Tianying Zhang

2/21/2018

Tidy Data

Introduction

```
library(tidyverse)
```

Tidy Data

1. Using prose, describe how the variables and observations are organized in each of the sample tables.

In `table1` each row is a (country, year) with variables `cases` and `population`.

```
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
```

In `table2`, each row is country, year , variable ("cases", "population") combination, and there is a `count` variable with the numeric value of the variable.

```
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
##  7 Brazil       2000 cases           80488
##  8 Brazil       2000 population  174504898
##  9 China        1999 cases          212258
## 10 China        1999 population 1272915272
## 11 China        2000 cases          213766
## 12 China        2000 population 1280428583
```

In `table3`, each row is a (country, year) combination with the column `rate` having the rate of cases to population as a character string in the format `"cases/rate"`.

```
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
```

Table 4 is split into two tables, one table for each variable: `table4a` is the table for cases, while `table4b` is the table for population. Within each table, each row is a country, each column is a year, and the cells are the value of the variable for the table.

```
table4a
```

```
## # A tibble: 3 x 3
##   country     `1999` `2000`
## * <chr>        <int>  <int>
## 1 Afghanistan    745   2666
## 2 Brazil       37737  80488
## 3 China       212258 213766
```

```
table4b
```

```
## # A tibble: 3 x 3
##   country         `1999`     `2000`
## * <chr>            <int>      <int>
## 1 Afghanistan   19987071   20595360
## 2 Brazil       172006362  174504898
## 3 China       1272915272 1280428583
```

2. Compute the `rate` for `table2`, and `table4a` + `table4b`. You will need to perform four operations:

- Extract the number of TB cases per country per year.
- Extract the matching population per country per year.
- Divide cases by population, and multiply by 10000.
- Store back in the appropriate place.

Which representation is easiest to work with? Which is hardest? Why?

Without using the join functions introduced in Ch 12:

```
tb2_cases <- filter(table2, type == "cases")[["count"]]
tb2_country <- filter(table2, type == "cases")[["country"]]
tb2_year <- filter(table2, type == "cases")[["year"]]
tb2_population <- filter(table2, type == "population")[["count"]]
table2_clean <- tibble(country = tb2_country,
       year = tb2_year,
       rate = tb2_cases / tb2_population)
table2_clean
```

```
## # A tibble: 6 x 3
##   country      year      rate
##   <chr>       <int>     <dbl>
## 1 Afghanistan  1999 0.0000373
## 2 Afghanistan  2000 0.000129
```

```
## 3 Brazil       1999 0.000219
## 4 Brazil       2000 0.000461
## 5 China        1999 0.000167
## 6 China        2000 0.000167
```

Note, that this assumes that all observations are sorted so that each country, year will have the observation for cases followed by population.

```
tibble(country = table4a[["country"]],
       `1999` = table4a[["1999"]] / table4b[["1999"]],
       `2000` = table4b[["2000"]] / table4b[["2000"]])
```

```
## # A tibble: 3 x 3
##   country        `1999` `2000`
##   <chr>           <dbl>  <dbl>
## 1 Afghanistan 0.0000373   1.00
## 2 Brazil      0.000219    1.00
## 3 China       0.000167    1.00
```

or

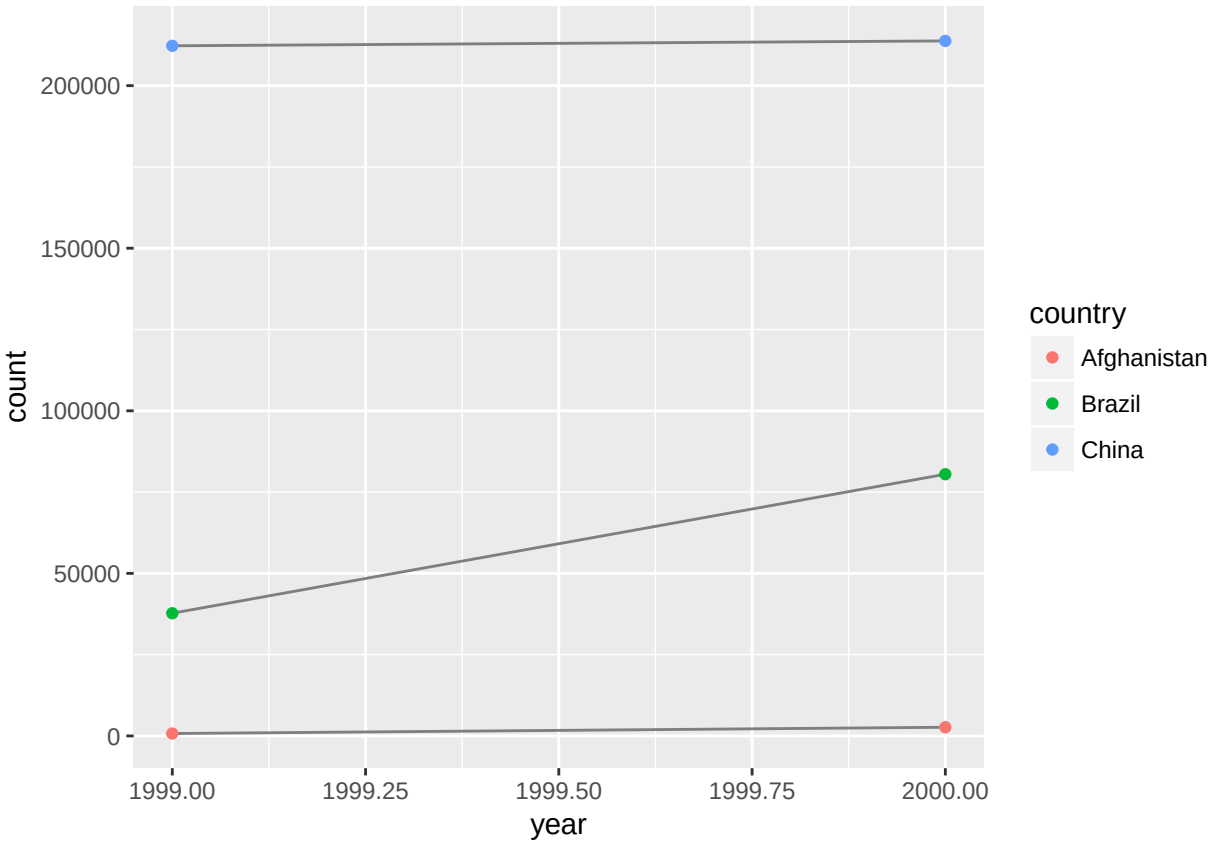
```
tibble(country = rep(table4a[["country"]], 2),
       year = rep(c(1999, 2000), each = nrow(table4a)),
       `rate` = c(table4a[["1999"]] / table4b[["1999"]],
                  table4b[["2000"]] / table4b[["2000"]]))
```

```
## # A tibble: 6 x 3
##   country      year      rate
##   <chr>       <dbl>     <dbl>
## 1 Afghanistan  1999 0.0000373
## 2 Brazil       1999 0.000219
## 3 China        1999 0.000167
## 4 Afghanistan  2000 1.00
## 5 Brazil       2000 1.00
## 6 China        2000 1.00
```

3. Recreate the plot showing change in cases over time using `table2` instead of `table1`. What do you need to do first?

First, I needed to filter the tibble to only include those rows that represented the "cases" variable.

```
table2 %>%
  filter(type == "cases") %>%
  ggplot(aes(year, count)) +
  geom_line(aes(group = country), colour = "grey50") +
  geom_point(aes(colour = country))
```



Spreading and Gathering

This code is reproduced from the chapter because it is needed by the exercises:

```
tidy4a <- table4a %>%
  gather(`1999`, `2000`, key = "year", value = "cases")
tidy4b <- table4b %>%
  gather(`1999`, `2000`, key = "year", value = "cases")
```

1. Why are `gather()` and `spread()` not perfectly symmetrical? Carefully consider the following example:

```
stocks <- tibble(
  year   = c(2015, 2015, 2016, 2016),
  half   = c(   1,    2,     1,    2),
  return = c(1.88, 0.59, 0.92, 0.17)
)
stocks %>%
  spread(year, return) %>%
  gather("year", "return", `2015`:`2016`)
```

```
## # A tibble: 4 x 3
##    half year  return
##   <dbl> <chr>  <dbl>
## 1  1.00 2015   1.88
## 2  2.00 2015   0.590
## 3  1.00 2016   0.920
## 4  2.00 2016   0.170
```

The functions `spread` and `gather` are not perfectly symmetrical because column type information is not transferred between them. In the original table the column `year` was numeric, but after running `spread()` and `gather()` it is a character vector. This is because variable names are always converted to a character vector by `gather()`.

The `convert` argument tries to convert character vectors to the appropriate type. In the background this uses the `type.convert` function.

```
stocks %>%
  spread(year, return) %>%
  gather("year", "return", `2015`:`2016`, convert = TRUE)
```

```
## # A tibble: 4 x 3
##    half  year return
##   <dbl> <int>  <dbl>
## 1  1.00  2015  1.88
## 2  2.00  2015  0.590
## 3  1.00  2016  0.920
## 4  2.00  2016  0.170
```

2. Why does this code fail?

```
table4a %>%
  gather(1999, 2000, key = "year", value = "cases")
```

```
## Error in inds_combine(.vars, ind_list): Position must be between 0 and n
```

The code fails because the column names `1999` and `2000` are not standard and thus needs to be quoted. The tidyverse functions will interpret `1999` and `2000` without quotes as looking for the 1999th and 2000th column of the data frame. This will work:

```
table4a %>%
  gather(`1999`, `2000`, key = "year", value = "cases")
```

```
## # A tibble: 6 x 3
##   country     year   cases
##   <chr>       <chr>  <int>
## 1 Afghanistan 1999     745
## 2 Brazil      1999   37737
## 3 China       1999  212258
## 4 Afghanistan 2000    2666
## 5 Brazil      2000   80488
## 6 China       2000  213766
```

3. Why does spreading this tibble fail? How could you add a new column to fix the problem?

```
people <- tribble(
  ~name,             ~key,    ~value,
  #-----------------/--------/------
  "Phillip Woods",   "age",       45,
  "Phillip Woods",   "height",   186,
  "Phillip Woods",   "age",       50,
  "Jessica Cordero", "age",       37,
  "Jessica Cordero", "height",   156
)
glimpse(people)
```

```
## Observations: 5
## Variables: 3
```

```
## $ name  <chr> "Phillip Woods", "Phillip Woods", "Phillip Woods", "Jess...
## $ key   <chr> "age", "height", "age", "age", "height"
## $ value <dbl> 45, 186, 50, 37, 156
```

```
spread(people, key, value)
```

```
## Error: Duplicate identifiers for rows (1, 3)
```

Spreading the data frame fails because there are two rows with "age" for "Phillip Woods". We would need to add another column with an indicator for the number observation it is,

```
people <- tribble(
  ~name,             ~key,    ~value, ~obs,
  #-----------------/--------/------/------
  "Phillip Woods",   "age",       45, 1,
  "Phillip Woods",   "height",   186, 1,
  "Phillip Woods",   "age",       50, 2,
  "Jessica Cordero", "age",       37, 1,
  "Jessica Cordero", "height",   156, 1
)
spread(people, key, value)
```

```
## # A tibble: 3 x 4
##   name              obs   age height
##   <chr>           <dbl> <dbl>  <dbl>
## 1 Jessica Cordero  1.00  37.0    156
## 2 Phillip Woods    1.00  45.0    186
## 3 Phillip Woods    2.00  50.0     NA
```

4. Tidy the simple tibble below. Do you need to spread or gather it? What are the variables?

```
preg <- tribble(
  ~pregnant, ~male, ~female,
  "yes",     NA,    10,
  "no",      20,    12
)
```

You need to gather it. The variables are:

- pregnant: logical ("yes", "no")
- female: logical
- count: integer

```
gather(preg, sex, count, male, female) %>%
  mutate(pregnant = pregnant == "yes",
         female = sex == "female") %>%
  select(-sex)
```

```
## # A tibble: 4 x 3
##   pregnant count female
##   <lgl>    <dbl> <lgl>
## 1 T         NA   F
## 2 F         20.0 F
## 3 T         10.0 T
## 4 F         12.0 T
```

Converting `pregnant` and `female` from character vectors to logical was not necessary to tidy it, but it makes it easier to work with.