30535 Skills Problem Set 4

4/28/2022

Front matter
This submission is my work alone and complies with the 30535 integrity policy.

Add your initials to indicate your agreement: **W.J.**

Late coins used this pset: 1. Late coins left: 0.

Submission Notes: Total page of the file is 23 and I've submitted on github

Clear Global Environment

```
knitr::opts_chunk$set(
  echo = TRUE,
  message = FALSE,
  warning = FALSE
)
rm(list = ls())
```

Working Directory and Loading Packages

```
# Setting the Working Directory
setwd("~/Desktop/Spring Quarter 2022/DPPP R/Week 5/skills-problem-set-4-weiluj")
# Load packages
library(ggplot2)
```

1 Tidy data with pivot_wider() and pivot_longer()

1.1

```
# Loading Packages
library(tidyverse)
?table1

# table 1
str(table1)
```

```
## tibble [6 x 4] (S3: tbl_df/tbl/data.frame)
##  $ country   : chr [1:6] "Afghanistan" "Afghanistan" "Brazil" "Brazil" ...
##  $ year      : int [1:6] 1999 2000 1999 2000 1999 2000
##  $ cases     : int [1:6] 745 2666 37737 80488 212258 213766
##  $ population: int [1:6] 19987071 20595360 172006362 174504898 1272915272 1280428583
```

```r
table1 %>%
  pivot_wider(
    names_from = year,
    values_from = c(cases, population),
    values_fn = ~ as.numeric(.x)
  ) %>%
  mutate(
    rate_1999 = cases_1999 / population_1999 * 10000,
    rate_2000 = cases_2000 / population_2000 * 10000,
    rate_all = (cases_1999 + cases_2000) /
      (population_1999 + population_2000) * 10000
  ) %>%
  select(country, starts_with("rate"), everything())
```

```
## # A tibble: 3 x 8
##   country     rate_1999 rate_2000 rate_all cases_1999 cases_2000 population_1999
##   <chr>           <dbl>     <dbl>    <dbl>      <dbl>      <dbl>           <dbl>
## 1 Afghanistan     0.373      1.29    0.841        745       2666        19987071
## 2 Brazil          2.19       4.61    3.41       37737      80488       172006362
## 3 China           1.67       1.67    1.67      212258     213766      1272915272
## # ... with 1 more variable: population_2000 <dbl>
```

```r
# table 2
str(table2)
```

```
## tibble [12 x 4] (S3: tbl_df/tbl/data.frame)
##  $ country: chr [1:12] "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
##  $ year   : int [1:12] 1999 1999 2000 2000 1999 1999 2000 2000 1999 1999 ...
##  $ type   : chr [1:12] "cases" "population" "cases" "population" ...
##  $ count  : int [1:12] 745 19987071 2666 20595360 37737 172006362 80488 174504898 212258 1272915272
```

```r
table2 %>%
  pivot_wider(
    names_from = c(year, type),
    values_from = count,
    values_fn = ~ as.numeric(.x)
  ) %>%
  mutate(
    rate_1999 = `1999_cases` / `1999_population` * 10000,
    rate_2000 = `2000_cases` / `2000_population` * 10000,
    rate_all = (`1999_cases` + `2000_cases`) /
      (`1999_population` + `2000_population`) * 10000
  ) %>%
  select(country, starts_with("rate"), everything())
```

```
## # A tibble: 3 x 8
##   country     rate_1999 rate_2000 rate_all `1999_cases` `1999_population`
##   <chr>           <dbl>     <dbl>    <dbl>        <dbl>             <dbl>
## 1 Afghanistan     0.373      1.29    0.841          745          19987071
## 2 Brazil          2.19       4.61    3.41         37737         172006362
## 3 China           1.67       1.67    1.67        212258        1272915272
## # ... with 2 more variables: `2000_cases` <dbl>, `2000_population` <dbl>
```

```r
# table 3
str(table3)
```

```
## tibble [6 x 3] (S3: tbl_df/tbl/data.frame)
##  $ country: chr [1:6] "Afghanistan" "Afghanistan" "Brazil" "Brazil" ...
##  $ year   : int [1:6] 1999 2000 1999 2000 1999 2000
##  $ rate   : chr [1:6] "745/19987071" "2666/20595360" "37737/172006362" "80488/174504898" ...
```

```r
table3 %>%
  separate(rate,
    into = c("cases", "population"),
    convert = TRUE
  ) %>%
  pivot_wider(
    names_from = year,
    values_from = c(cases, population),
    values_fn = ~ as.numeric(.x)
  ) %>%
  mutate(
    rate_1999 = cases_1999 / population_1999 * 10000,
    rate_2000 = cases_2000 / population_2000 * 10000,
    rate_all = (cases_1999 + cases_2000) /
      (population_1999 + population_2000) * 10000
  ) %>%
  select(country, starts_with("rate"), everything())
```

```
## # A tibble: 3 x 8
##   country     rate_1999 rate_2000 rate_all cases_1999 cases_2000 population_1999
##   <chr>           <dbl>     <dbl>    <dbl>      <dbl>      <dbl>           <dbl>
## 1 Afghanistan     0.373      1.29    0.841        745       2666        19987071
## 2 Brazil          2.19       4.61    3.41       37737      80488       172006362
## 3 China           1.67       1.67    1.67      212258     213766      1272915272
## # ... with 1 more variable: population_2000 <dbl>
```

```r
# table 4a and table 4b
str(table4a)
```

```
## tibble [3 x 3] (S3: tbl_df/tbl/data.frame)
##  $ country: chr [1:3] "Afghanistan" "Brazil" "China"
##  $ 1999   : int [1:3] 745 37737 212258
##  $ 2000   : int [1:3] 2666 80488 213766
```

```r
table4 <-
  left_join(table4a, table4b, by = "country") %>%
  mutate(
    across(.cols = where(is.integer), .fns = as.numeric)
  )
table4 %>%
  mutate(
    rate_1999 = `1999.x` / `1999.y` * 10000,
    rate_2000 = `2000.x` / `2000.y` * 10000,
    rate_all = (`1999.x` + `2000.x`) / (`1999.y` + `2000.y`) * 10000
```

```
  ) %>%
  select(country, starts_with("rate"), everything())
```

```
## # A tibble: 3 x 8
##   country     rate_1999 rate_2000 rate_all `1999.x` `2000.x`   `1999.y` `2000.y`
##   <chr>           <dbl>     <dbl>    <dbl>    <dbl>    <dbl>      <dbl>    <dbl>
## 1 Afghanistan     0.373      1.29    0.841      745     2666   19987071   2.06e7
## 2 Brazil          2.19       4.61    3.41     37737    80488  172006362   1.75e8
## 3 China           1.67       1.67    1.67    212258   213766 1272915272   1.28e9
```

```
# table 5
str(table5)
```

```
## tibble [6 x 4] (S3: tbl_df/tbl/data.frame)
##  $ country: chr [1:6] "Afghanistan" "Afghanistan" "Brazil" "Brazil" ...
##  $ century: chr [1:6] "19" "20" "19" "20" ...
##  $ year   : chr [1:6] "99" "00" "99" "00" ...
##  $ rate   : chr [1:6] "745/19987071" "2666/20595360" "37737/172006362" "80488/174504898" ...
```

```
table5 %>%
  select(-century) %>%
  separate(rate,
    into = c("cases", "population"),
    convert = TRUE
  ) %>%
  pivot_wider(
    names_from = year,
    values_from = c(cases, population),
    values_fn = ~ as.numeric(.x)
  ) %>%
  mutate(
    rate_99 = cases_99 / population_99 * 10000,
    rate_00 = cases_00 / population_00 * 10000,
    rate_all = (cases_99 + cases_00) /
      (population_99 + population_00) * 10000
  ) %>%
  mutate(
    across(.cols = where(is.integer), .fns = as.numeric)
  ) %>%
  select(country, starts_with("rate"), everything())
```

```
## # A tibble: 3 x 8
##   country rate_99 rate_00 rate_all cases_99 cases_00 population_99 population_00
##   <chr>     <dbl>   <dbl>    <dbl>    <dbl>    <dbl>         <dbl>         <dbl>
## 1 Afghan~   0.373    1.29    0.841      745     2666      19987071      20595360
## 2 Brazil    2.19     4.61    3.41     37737    80488     172006362     174504898
## 3 China     1.67     1.67    1.67    212258   213766    1272915272    1280428583
```

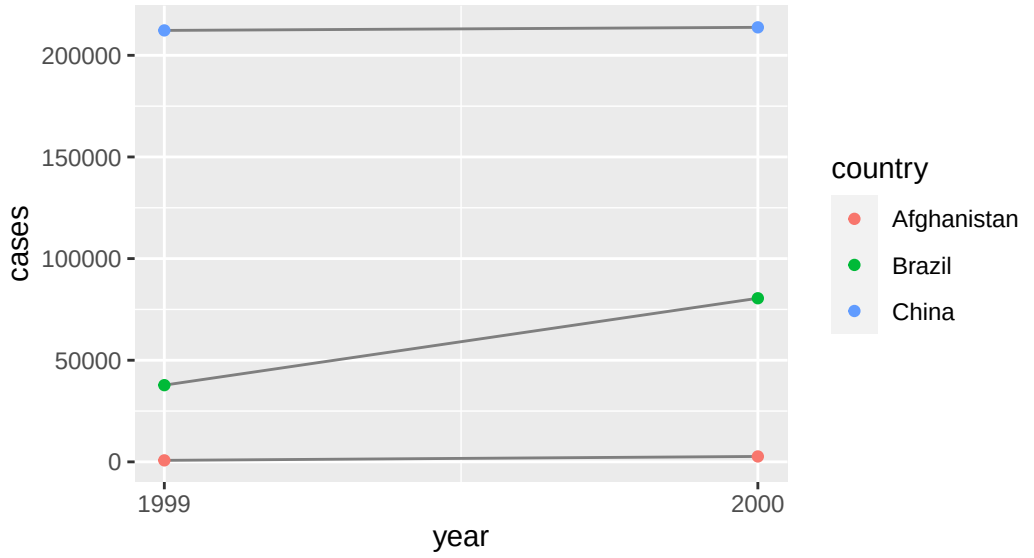
Answers

All tables are not quite straightforward. However, I think table4a and table4b are easier to work with, while table5 is harder. For table4a and table4b, although we need to first combine the two tibbles, the structure

of the tibble is pretty easy to work with. Each year's population and cases are listed in a separate column. We only need to use mutate() to calculate the ratio without pivoting data.
For table5, we first need to identify which column works better to represent the year of the data, then we also need to separate the values in column *rate* to convert the character type into numeric type. After those steps, we still need to perform all the procedures we applied to other tibbles(pivoting data, mutate etc.). Therefore, I think table5 is the most tedious one to work with.
The best data to work with would be the one with country, year, population by year, cases by year information listed in separate columns.

1.2

```
table2_plot <- table2 %>%
  filter(type == "cases") %>%
  group_by(country, year) %>%
  summarise(cases = count)
ggplot(table2_plot, aes(x = year, y = cases)) +
  geom_line(aes(group = country),
    color = "grey50"
  ) +
  geom_point(aes(color = country)) +
  scale_x_continuous(breaks = unique(table2_plot$year))
```



Answers

To recreate the plot from the textbook showing changes in cases over time using table2, we first need to filter the data to only keep rows with cases value and remove those with population values.

Reference

R for Data Science Book Section 12.2

1.3

```
# Code provided
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
```

```
## # A tibble: 4 x 3
##    year  half return
##   <dbl> <dbl>  <dbl>
## 1  2015     1   1.88
## 2  2015     2   0.59
## 3  2016     1   0.92
## 4  2016     2   0.17
```

```
stocks %>%
  pivot_wider(names_from = year, values_from = return) %>%
  pivot_longer(`2015`:`2016`, names_to = "year", values_to = "return")
```

```
## # A tibble: 4 x 3
##    half year  return
##   <dbl> <chr>  <dbl>
## 1     1 2015    1.88
## 2     1 2016    0.92
## 3     2 2015    0.59
## 4     2 2016    0.17
```

```
# Correction
stocks %>%
  pivot_wider(names_from = year, values_from = return) %>%
  pivot_longer(`2015`:`2016`,
    names_to = "year", values_to = "return",
    names_transform = list(year = as.numeric)
  )
```

```
## # A tibble: 4 x 3
##    half  year return
##   <dbl> <dbl>  <dbl>
## 1     1  2015   1.88
## 2     1  2016   0.92
## 3     2  2015   0.59
## 4     2  2016   0.17
```

Answers

- pivot_longer() and pivot_wider() are not perfectly symmetrical because we cannot keep the column type the same when converting data. pivot_longer() will combine columns with potential different types of data into the same column, which means the data type may be messed. We need to use names_to() to specify the name of the new column for information stored in the column names of data selected, and we use values_to() to specify the name of the column to create from the data stored in cell values. If we use names_from(), then it's to specify which column to get the name of the output column. values_from() is used to specify which column to get the output data. The *from* expression is used for pivoting data from long to wider, which does not meet our requirement here.

- However, pivot_wider() creates column names from values in column. In the example above, pivot_wider() pivots the tibble to create a tibble with year as column names, and the values would be stored as column values, while pivot_longer() will transform it into a tibble with half, year and return columns. The year column will be mistaken as character, therefore, we won't get the exact same tibble by using pivot_wider() and then pivot_longer() to manipulate data.
  To correct this, we need to use *names_transform = list(year = as.numeric)*.

1.4

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

1.5

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

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

Answers
The reason why the code fails is because we cannot treat *1999* and *2000* as integers in this case. By looking at the tibble, we know that both *1999* and *2000* columns are character type. If we do not include the backtick symbol, then R will automatically think we are asking R to select the 1999th to 2000th columns in table4a. That's why it tells us that *Locations 1999 and 2000 don't exist.* In order to correctly choose the columns, we need to add the backtick symbol.

1.6

```
# Original tibble
people <- tribble(
  ~name, ~key, ~value,
  #----------------/--------/------
  "Phillip Woods", "age", 45,
  "Phillip Woods", "height", 186,
  "Phillip Woods", "age", 50,
  "Phillip Woods", "height", 185,
  "Jessica Cordero", "age", 37,
```

```
  "Jessica Cordero", "height", 156
)

# Using pivot_wider()
people %>%
  pivot_wider(names_from = "name", values_from = "value")
```

```
## # A tibble: 2 x 3
##   key    `Phillip Woods` `Jessica Cordero`
##   <chr>  <list>          <list>
## 1 age    <dbl [2]>       <dbl [1]>
## 2 height <dbl [2]>       <dbl [1]>
```

```
# Update dataset
people_unique <-
  people %>%
  mutate(num = c(1, 1, 2, 2, 1, 1))
people_unique %>%
  pivot_wider(names_from = "name", values_from = "value")
```

```
## # A tibble: 4 x 4
##   key      num `Phillip Woods` `Jessica Cordero`
##   <chr>  <dbl>           <dbl>             <dbl>
## 1 age        1              45                37
## 2 height     1             186               156
## 3 age        2              50                NA
## 4 height     2             185                NA
```

Answers
Pivot_wider() will not be able to work on this tibble because both the *name* and *key* columns are not unique identifier for each row. We have two rows for *Phillips Woods* with regard to his age and height. To solve the problem, we need to add a unique identifier as shown in the code above.

1.7

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

```
## # A tibble: 2 x 3
##   pregnant  male female
##   <chr>    <dbl>  <dbl>
## 1 yes         NA     10
## 2 no          20     12
```

```r
# Using pivot_longer()
preg %>%
  pivot_longer(
    male:female,
    names_to = "gender",
    values_to = "count"
  ) %>%
  arrange(desc(pregnant), gender) %>%
  drop_na(count)
```

```
## # A tibble: 3 x 3
##   pregnant gender count
##   <chr>    <chr>  <dbl>
## 1 yes      female    10
## 2 no       female    12
## 3 no       male      20
```

Answers

We would like to use pivot_longer() here to make the tibble longer. Because we would like to have a more straightforward understanding of the distribution of people get pregnant or not by gender. Specifically, since we know no man would pregnant, it would be more important to know the distribution of females by their pregnancy status. pivot_longer() would make it more clear. By doing so, we're transforming an explicit missing value to implicit missing value. If we use pivot_wider(), missing values could only be explicit, which is not necessary in this case.

Note: *drop_na(count)* is optional. Since no man would pregnant, the row is not informative for us but we could also just keep it.

1.8

```r
# Tibble 1
tibble(x = c("a,b,c", "d,e,f,g", "h,i,j")) %>%
  separate(x, c("one", "two", "three"))
```

```
## # A tibble: 3 x 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 d     e     f
## 3 h     i     j
```

```r
tibble(x = c("a,b,c", "d,e,f,g", "h,i,j")) %>%
  separate(x, c("one", "two", "three"), extra = "drop")
```

```
## # A tibble: 3 x 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 d     e     f
## 3 h     i     j
```

```
tibble(x = c("a,b,c", "d,e,f,g", "h,i,j")) %>%
  separate(x, c("one", "two", "three"), extra = "merge")
```

```
## # A tibble: 3 x 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 d     e     f,g
## 3 h     i     j
```

```
# Tibble 2
tibble(x = c("a,b,c", "d,e", "f,g,i")) %>%
  separate(x, c("one", "two", "three"))
```

```
## # A tibble: 3 x 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 d     e     <NA>
## 3 f     g     i
```

```
tibble(x = c("a,b,c", "d,e", "f,g,i")) %>%
  separate(x, c("one", "two", "three"), fill = "right")
```

```
## # A tibble: 3 x 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 d     e     <NA>
## 3 f     g     i
```

```
tibble(x = c("a,b,c", "d,e", "f,g,i")) %>%
  separate(x, c("one", "two", "three"), fill = "left")
```

```
## # A tibble: 3 x 3
##   one   two   three
##   <chr> <chr> <chr>
## 1 a     b     c
## 2 <NA>  d     e
## 3 f     g     i
```

Answers
The extra argument is used when there are too many pieces then it should be to separate, and the fill argument is used when there are not enough pieces to work on.

extra options

- "warn": The default setting. R will send a warning message and drop extra values
- "drop": R will drop any extra values without a warning

- "merge": R will only splits at most length times. In other words, if there is one extra value, R will keep it and add it together with the second last value to the last column.(See the codes above)

fill options

- "warn": The default setting. R will send a warning message and fill from the right
- "right": R will fill with missing values on the right
- "left": R will fill with missing values on the left

Reference
R help function

1.9

```
who %>%
  select(country, iso2, iso3) %>%
  distinct() %>%
  group_by(country) %>%
  filter(n() > 1)
```

```
## # A tibble: 0 x 3
## # Groups:   country [0]
## # ... with 3 variables: country <chr>, iso2 <chr>, iso3 <chr>
```

Answers
If iso2 and iso3 do not add any additional information to the data, then within each country there should only be 1 type of combination of the country name, iso2 and iso3. The above code confirms this.
By searching online, I figured out that iso2 is the Internationally agreed two-letter country codes used to represent countries, and iso3 is the three-letter one, so the combination should be unique.
Reference: Wikipedia

2 Tidying case study

2.1

```
# Dataset used in Chapter 12.6
who_pivot_longer <- who %>%
  pivot_longer(
    new_sp_m014:newrel_f65,
    names_to = "key",
    values_to = "cases",
    values_drop_na = TRUE
  )

# Check the number of implicit missing values
nrow(who %>% complete(country, year)) - nrow(who)
```

```
## [1] 206
```

```r
# Find implicit missing values
anti_join(complete(who, country, year),
  who,
  by = c("country", "year")
) %>%
  select(country, year, iso2) %>%
  group_by(country, iso2) %>%
  summarise(
    start_year = min(year),
    end_year = max(year)
  )
```

```
## # A tibble: 9 x 4
## # Groups:   country [9]
##   country                           iso2  start_year end_year
##   <chr>                             <chr>      <int>    <int>
## 1 Bonaire, Saint Eustatius and Saba <NA>        1980     2009
## 2 Curacao                           <NA>        1980     2009
## 3 Montenegro                        <NA>        1980     2004
## 4 Netherlands Antilles              <NA>        2010     2013
## 5 Serbia                            <NA>        1980     2004
## 6 Serbia & Montenegro               <NA>        2005     2013
## 7 Sint Maarten (Dutch part)         <NA>        1980     2009
## 8 South Sudan                       <NA>        1980     2010
## 9 Timor-Leste                       <NA>        1980     2001
```

```r
# Find case values equal to 0
who_pivot_longer %>%
  filter(cases == 0) %>%
  nrow()
```

```
## [1] 11080
```

```r
# Check country-year pairs which are explicitly missing TB data
country_year_pair <-
  who %>%
  pivot_longer(
    new_sp_m014:newrel_f65,
    names_to = "key",
    values_to = "cases"
  ) %>%
  group_by(country, year) %>%
  mutate(pect_missing = sum(is.na(cases)) / n())
n_groups(country_year_pair)
```

```
## [1] 7240
```

Answers

Explicit missing value means being shown as absent in the dataset. It will usually be represented by *NA*, while implicit missing value will simply not occur in the data. In other words, implicit missing value refers to the absence of presence in the dataset.

In this case, it make sense to use na.rm = TRUE to simply remove NA values. The reason is that we notice

there are 0 values in the case column, which means in this dataset if there is no case, it will be represented with 0 instead of NA. By further analysis with *anti_join()*, we notice that the implicit missing values are from countries with years before the founding the country. In sum, we know that NA values means there is no available data in this dataset, so it's ok to just remove them.
a. Yes, there are 206 rows of implicit missing values
b. There are 7240 country-year pairs are explicitly missing TB data.

Reference
1. R for Data Science Chapter 16.2 Explicit vs Implicit Missing Values
2. For anti_join(): https://www.statology.org/dplyr-anti-join/

2.2

By referring to the code in Q2.1, we know that:

- *zero* in the *cases* column means there is no TB case in the country-year pair for people belong to the *key* group(diagnosis method, gender and age group). For example, we get 0 case for people tagged as *new_sp_m014* in Afghanistan in 1997. It means there was no case in Afghanistan in 1997 for males aged between 0-14 years old who were diagnosed by positive pulmonary smear method.
- However, *NA* in this case study means there is no available data for this country-year pair, given that the country exists. Data before a country exist is categorized as implicit missing value.

2.3

```
# Neglect mutate()
who_pivot_longer %>%
  separate(key, c("new", "type", "sexage"), sep = "_") %>%
  arrange(desc(new))
```

```
## # A tibble: 76,046 x 8
##    country     iso2  iso3   year new    type  sexage cases
##    <chr>       <chr> <chr> <int> <chr>  <chr> <chr>  <int>
##  1 Afghanistan AF    AFG    2013 newrel m014  <NA>    1705
##  2 Afghanistan AF    AFG    2013 newrel f014  <NA>    1749
##  3 Albania     AL    ALB    2013 newrel m014  <NA>      14
##  4 Albania     AL    ALB    2013 newrel m1524 <NA>      60
##  5 Albania     AL    ALB    2013 newrel m2534 <NA>      61
##  6 Albania     AL    ALB    2013 newrel m3544 <NA>      32
##  7 Albania     AL    ALB    2013 newrel m4554 <NA>      44
##  8 Albania     AL    ALB    2013 newrel m5564 <NA>      50
##  9 Albania     AL    ALB    2013 newrel m65   <NA>      67
## 10 Albania     AL    ALB    2013 newrel f014  <NA>       5
## # ... with 76,036 more rows
```

```
# Include mutate()
who_clean_key <- who_pivot_longer %>%
  mutate(key = stringr::str_replace(key, "newrel", "new_rel"))
who_sep_keys <-
  who_clean_key %>%
  separate(key, c("new", "type", "sexage"), sep = "_")
```

Answers
If we neglect the mutate() step, first we will get a warning message from R saying that *Expected 3 pieces. Missing pieces filled with `NA` in 2580 row* . It means we are missing a large portion of information.
By checking the dataset, we find that we get wrong separation result due to the fact that the original *key* column does not follow consistent format. For example, TBs diagonised with *relapse* method are coded as *newrel* instead of *new_rel*, while the expected format should be *new_method*. The inconsistency leads to the error that those data have no *sexage* information while their *type* column is a combination of gender and age. Therefore, we must include the mutate() step to make sure *key* form is consistent.
Reference
Class material

2.4

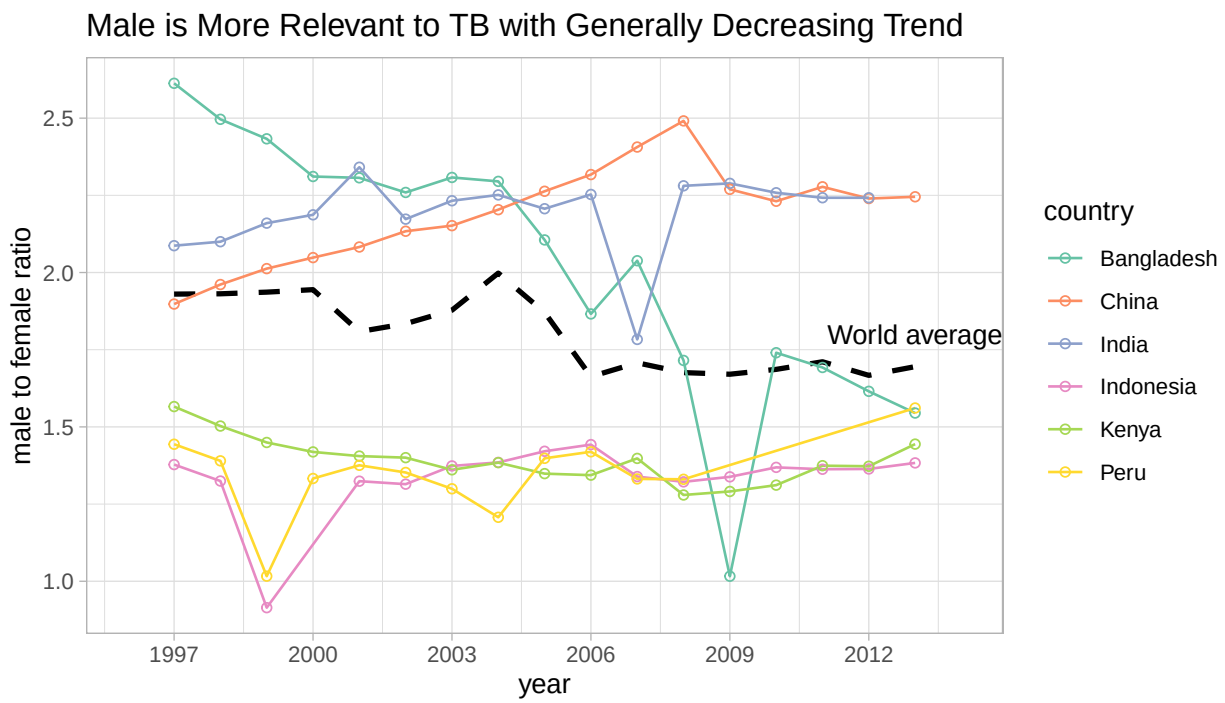
```
# Tidied WHO data
who_tidy_data <-
  who_sep_keys %>%
  select(-new, -iso2, -iso3) %>%
  separate(sexage, c("sex", "age"), sep = 1)
# a.
who_plot <-
  who_tidy_data %>%
  group_by(country, year, sex) %>%
  summarise(cases = sum(cases)) %>%
  arrange(desc(cases))
who_plot
```

```
## # A tibble: 6,921 x 4
## # Groups:   country, year [3,484]
##    country  year sex    cases
##    <chr>   <int> <chr>  <int>
##  1 India    2007 m     767767
##  2 China    2009 m     613947
##  3 China    2011 m     601126
##  4 China    2010 m     600094
##  5 China    2012 m     593751
##  6 China    2013 m     586127
##  7 India    2011 m     444202
##  8 India    2010 m     436764
##  9 India    2012 m     435396
## 10 India    2009 m     434690
## # ... with 6,911 more rows
```

```
# c.
who_plot <-
  who_plot %>%
  pivot_wider(
    names_from = "sex",
    values_from = "cases"
    ) %>%
  filter(year >= 1997) %>%
  mutate(
    cases = sum(f, m),
```

```
    m_f_ratio = m / f
    ) %>%
  drop_na() %>%
  filter(m_f_ratio != Inf) %>%
  arrange(
    desc(cases),
    desc(m_f_ratio)
    )
# e.
# World average by Year
who_plot_world <-
  who_plot %>%
  ungroup() %>%
  group_by(year) %>%
  summarise(m_f_ratio_avg = mean(m_f_ratio))
# Male to female ratio among countries with generally most cases
who_country_most <-
  who_plot %>%
  filter(country %in% c(
    "China", "India", "Indonesia",
    "Bangladesh", "Kenya", "Peru"
  )) %>%
  left_join(who_plot_world, by = "year")
# Result 1
ggplot(who_country_most) +
  geom_line(aes(x = year, y = m_f_ratio_avg),
    color = "black",
    linetype = 2,
    size = 1
  ) +
  geom_line(aes(
    x = year, y = m_f_ratio,
    color = country
  )) +
  geom_point(aes(
    x = year, y = m_f_ratio,
    color = country
  ),
  shape = 1
  ) +
  scale_color_brewer(palette = "Set2") +
  scale_x_continuous(
    breaks = seq(1997, 2013, 3),
    limits = c(1996, 2014)
  ) +
  annotate(
    "text",
    x = 2013, y = 1.8,
    label = "World average",
    color = "black"
  ) +
  labs(
    title = "Male is More Relevant to TB with Generally Decreasing Trend",
```

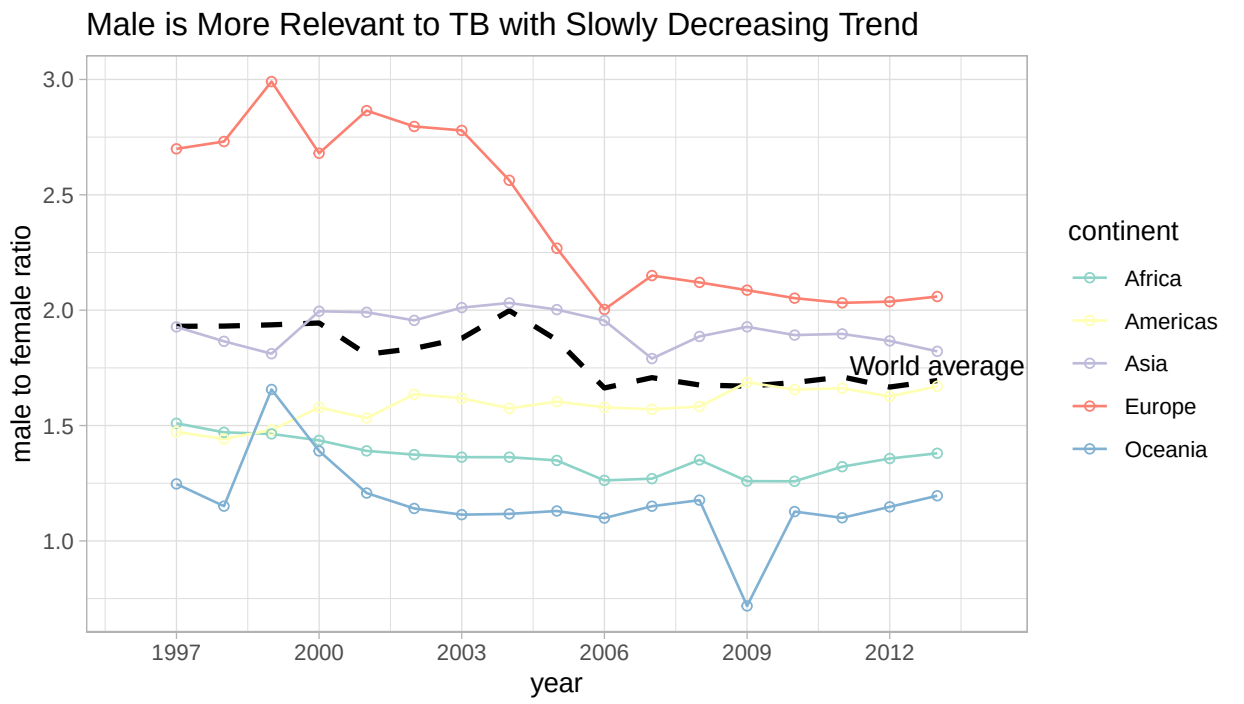
```
    y = "male to female ratio"
  ) +
  theme_light()
```




```
# Result 2
library(countrycode)
continent <- as.data.frame(countrycode(who_plot[["country"]],
                                       origin = "country.name",
                                       destination = "continent"
                                       )
                           )
who_continent_plot <- cbind(who_plot, continent)
colnames(who_continent_plot)[7] <- c("continent")

who_continent_plot <-
  who_continent_plot %>%
  ungroup() %>%
  select(-`country`) %>%
  group_by(continent, year) %>%
  mutate(
    female = sum(f),
    male = sum(m),
    cases = female + male,
    m_f_ratio = male / female
  ) %>%
  drop_na() %>%
  select(-m, -f) %>%
  arrange(
    year,
    desc(m_f_ratio)
```

```
  ) %>%
  distinct(m_f_ratio) %>%
  left_join(who_plot_world, by = "year") %>%
  select(continent, m_f_ratio, everything())
ggplot(who_continent_plot) +
  geom_line(aes(x = year, y = m_f_ratio_avg),
    color = "black",
    linetype = 2,
    size = 1
  ) +
  geom_line(aes(
    x = year, y = m_f_ratio,
    color = continent)
    ) +
  geom_point(aes(
    x = year, y = m_f_ratio,
    color = continent
    ),
  shape = 1
  ) +
  scale_color_brewer(palette = "Set3") +
  scale_x_continuous(
    breaks = seq(1997, 2013, 3),
    limits = c(1996, 2014)
  ) +
  annotate(
    "text",
    x = 2013, y = 1.76,
    label = "World average",
    color = "black"
  ) +
  labs(
    title = "Male is More Relevant to TB with Slowly Decreasing Trend",
    y = "male to female ratio"
  ) +
  theme_light()
```



Answers

b. Using the raw value, i.e., the absolute number of cases in each country, year and sex group, is probably not useful because the number of total population has been increasing between 1997 and 2013. Specifically, the growth rate of male and female could be different. Besides, the total population of male and female is also different greatly in most country. Usually male population is larger than female population. Therefore, more male cases could not necessarily lead to the conclusion that TB is more associated with male. Data from World Bank also confirms the assumption.

d. Producing male to female TB case ratios by year while ignoring country would also be a bad idea. The reason is that male to female ratio differs greatly among different countries during different year. For example, female accounts for 50.7% of the total population in the US while the percentage is 48.7 in Afghanistan. Also, in the US the percentage of female is decreasing during the past 2 decades while the percentage fluctuates greatly in Afghanistan. If we calculate the ratio by year without taking countries into consideration, we're averaging the total effect and may miss important findings with regard to whether TB is associated with a specific sex. TB may be relevant to male more in some regions while correlates with females in other regions. *Data Source* World Bank

f.

- Based on the plot with the world's average percentage of males in TB cases by year, we can notice that Tuberculosis is highly associated with male since the ratio is stably around 1.75. Besides, the time trend is that the ratio of male TB cases to female cases is declining with time. We can notice a common increase in the ration around 2008 in both plots.
- However, if we plot it by country or by continent, we have some new findings. Since there are more than 200 countries in total, I selected those with most cases and have data available throughout the 13 years. Data form countries with the least cases may not be informative since the sample size is too small. Based on that, we can notice all those countries with the heaviest TB burden have more male suffering form TB nearly throughout the 13 years except Indonesia in 1999. This could further confirm our assumption that Tuberculosis may associate with male much more than female. Nonetheless, we notice different trends of the ratio among those countries. More Men in China and India have suffered form TB relatively to women, while in Bangladesh females are relatively more and more more affected

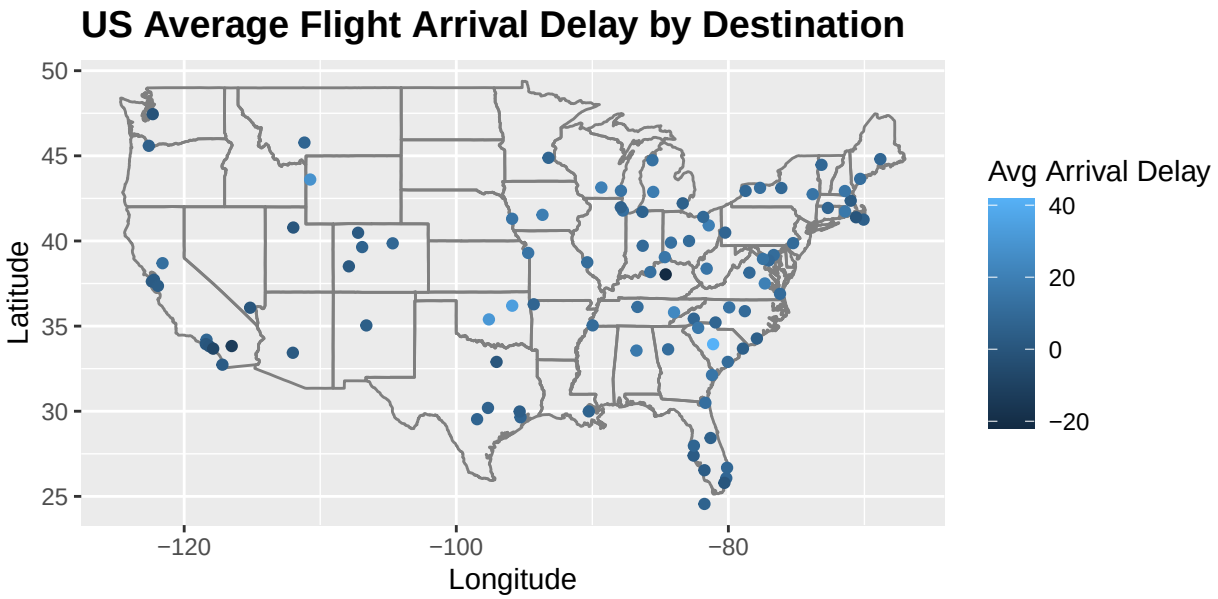
by TB compared to men. Although I noted earlier that the gender ratio in each country could differ, it's generally around 1.2, while we notice the ratio here is much more than 1.2. Therefore, we may say that based on the datasest we have, TB seems to be more associated with male, but the trend is slightly changing.

- To get a more solid observation, we could group the data by continent instead. It could reveal some differences within different countries while also keep the plot to be readable. By looking at the 2nd plot, we notice that in all the 5 continents mentioned here, male is more associated with TB with regard to the infectious cases compared to females in the same continent. Also, we can notice that the ratio is slowly declining, meaning that more female may be more likely to be affected by TB compared to the past. Among all those continents, males in Europe are significantly more related to TB, given that female accounts for percentage of the population in Europe. The ratio in Oceania (countries like Australia etc) have a more balanced ratio. Our general conclusion is probably not applicable there.

3 Joins

3.1

```
# Load dataset
flights <- nycflights13::flights
airports <- nycflights13::airports
# Combine data
avg_delays_by_dest <- flights %>%
  group_by(dest) %>%
  summarize(avg_delay = mean(arr_delay, na.rm = TRUE)) %>%
  inner_join(airports, by = c(dest = "faa")) %>%
  filter(dest != "HNL" & dest != "ANC")
# Plot
ggplot(avg_delays_by_dest,
  aes(
    x = lon, y = lat,
    color = avg_delay
  )
) +
  borders("state") +
  geom_point() +
  coord_quickmap() +
  labs(
    title = "US Average Flight Arrival Delay by Destination",
    x = "Longitude",
    y = "Latitude",
    color = "Avg Arrival Delay"
  ) +
  theme(plot.title = element_text(size = 14, face = "bold"))
```



Answers

We can observe in the map that flights to the east usually have longer arrival delay and available data in this dataset is kind of being converged to the east side since we see most points in the East in the US Map. We can also notice that flights to Columbia Metropolitan Airport, which is located in South Carolina, has the longest arrival delay in the dataset.

3.2

```
flights <-
  flights %>%
  left_join(
    airports,
    by = c("origin" = "faa")
  ) %>%
  left_join(
    airports,
    by = c("dest" = "faa"),
    suffix = c("_origin", "_dest")
  ) %>%
  select(
    year, month, day, starts_with(c("lat", "lon")),
    everything()
  )
head(flights)
```

```
## # A tibble: 6 x 33
##    year month   day lat_origin lat_dest lon_origin lon_dest dep_time
##   <int> <int> <int>      <dbl>    <dbl>      <dbl>    <dbl>    <int>
## 1  2013     1     1       40.7     30.0      -74.2    -95.3      517
## 2  2013     1     1       40.8     30.0      -73.9    -95.3      533
```

```
## 3  2013     1     1       40.6     25.8      -73.8    -80.3      542
## 4  2013     1     1       40.6     NA        -73.8     NA        544
## 5  2013     1     1       40.8     33.6      -73.9    -84.4      554
## 6  2013     1     1       40.7     42.0      -74.2    -87.9      554
## # ... with 25 more variables: sched_dep_time <int>, dep_delay <dbl>,
## #   arr_time <int>, sched_arr_time <int>, arr_delay <dbl>, carrier <chr>,
## #   flight <int>, tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
## #   name_origin <chr>, alt_origin <dbl>, tz_origin <dbl>, dst_origin <chr>,
## #   tzone_origin <chr>, name_dest <chr>, alt_dest <dbl>, tz_dest <dbl>,
## #   dst_dest <chr>, tzone_dest <chr>
```

3.3

```
plane_airline <-
  flights %>%
  filter(!is.na(tailnum)) %>%
  distinct(tailnum, carrier) %>%
  group_by(tailnum) %>%
  summarise(count_airline = n()) %>%
  filter(count_airline > 1)
plane_airline
```

```
## # A tibble: 17 x 2
##    tailnum count_airline
##    <chr>           <int>
##  1 N146PQ              2
##  2 N153PQ              2
##  3 N176PQ              2
##  4 N181PQ              2
##  5 N197PQ              2
##  6 N200PQ              2
##  7 N228PQ              2
##  8 N232PQ              2
##  9 N933AT              2
## 10 N935AT              2
## 11 N977AT              2
## 12 N978AT              2
## 13 N979AT              2
## 14 N981AT              2
## 15 N989AT              2
## 16 N990AT              2
## 17 N994AT              2
```

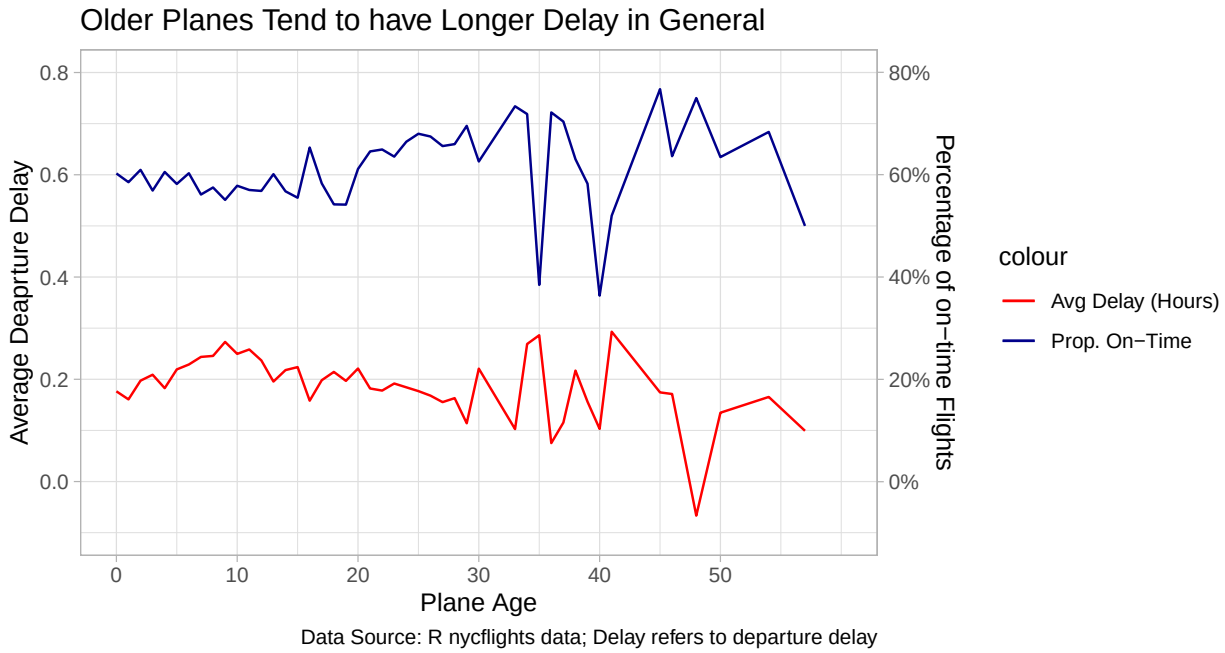
```
nrow(plane_airline)
```

```
## [1] 17
```

Answers
Not all planes only flew with a single airline. There are 17 planes which changed ownership within the dataset.

3.4

```
# Query
# Laod dataset planes to access planes' manufacturing information
planes <- nycflights13::planes
colnames(planes)[2] <- "plane_year"
# Combine planes' manufacturing year information with flights dataset
plane_age_dep_delay <-
  flights %>%
  left_join(planes,
    by = "tailnum"
  ) %>%
  mutate(plane_age = year - plane_year) %>%
  filter(!is.na(plane_age)) %>%
  group_by(plane_age) %>%
  summarise(
    mean_dep_delay = mean(dep_delay, na.rm = TRUE),
    good_flights = (sum(dep_delay <= 0, na.rm = TRUE) / n())
  )
# Result: plot relationship between departure delay and age of plane
ggplot(plane_age_dep_delay) +
  geom_line(aes(x = plane_age, y = mean_dep_delay / 60, color = "red")) +
  geom_line(aes(x = plane_age, y = good_flights, color = "darkblue")) +
  scale_x_continuous(
    breaks = seq(0, 50, 10),
    limits = c(0, 60)
  ) +
  scale_y_continuous(
    breaks = seq(0, 0.8, 0.2),
    limits = c(-0.1, 0.8),
    sec.axis = sec_axis(~.,
      name = "Percentage of on-time Flights",
      breaks = seq(0, 0.8, 0.2),
      labels = function(x) paste0(100*x, "%")
    )
  ) +
  scale_color_manual(
    values = c(
      "red" = "red",
      "darkblue" = "darkblue"
    ),
    labels = c("Avg Delay (Hours)", "Prop. On-Time")
  ) +
  labs(
    title = "Older Planes Tend to have Longer Delay in General",
    x = "Plane Age",
    y = "Average Deaprture Delay",
    caption = "Data Source: R nycflights data; Delay refers to departure delay"
  ) +
  theme(plot.title = element_text(size = 13,face = "bold"),
        plot.caption = element_text(hjust = 0, face = "italic")) +
  theme_light()
```



Answers

- Based on the graph above, we notice that in general, with increasing plane age, the average delay time will be longer. However, when the plane reaches to its 30ies, there will be a huge decline in delay time and great increase in the percentage of flights departed on time. The trend fluctuates greatly(a huge increase again round 35 years use) until the plane has been used for 40 years.
- Reasons could be when the plane has been used for around 30 years, people usually tend to believe it needs repair and extra attention to maintain good condition and elongate its expected expectancy. Otherwise passengers may not be willing to take those planes. Therefore, departure delay of those planes will decrease due to people's extra attention to reduce that.
- However, when planes have been used for over 40 years, it's hard to maintain good performance and they may be ready to *retire*. Those planes may have multiple problems to depart on time. Under this circumstance, the departure delay will rise again and their performance is not quite under control by their plane age. 30-40 may be the most important period for planes to be decided whether it could further be used a lot, so when there is a declined in underperformance around 35 years-use it's delay performance will be better again, while that's hard to achieve when the expected expentacy of planes has arrived a very high level(40 years)
- The decline of delay time after 50 years of use could be caused by the fact that those planes are seldom used due to safety concern etc, so the delay time could be shorter.