monday class notes

Tim Riffe

7/5/2021

Markdown basics

This here is normal text. I'm going to use it to take notes about demography stuff, very literally, just like I am right now.

You can start a code chunk just like this: `Ctrl + Alt + i`

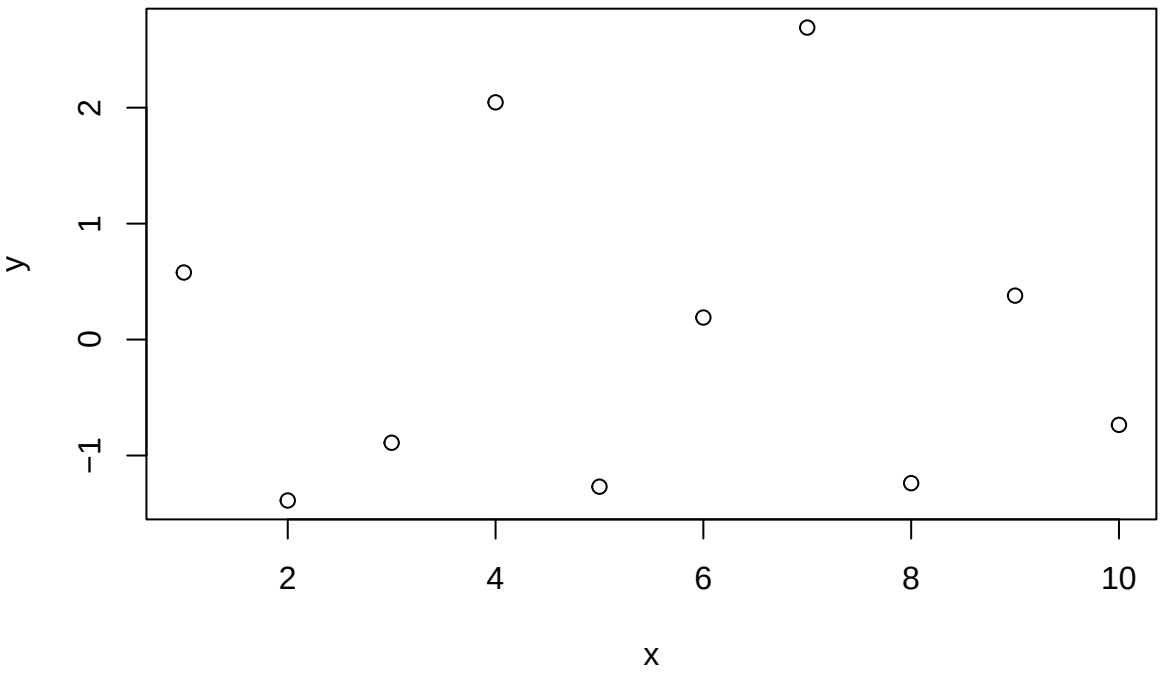
```
rnorm(10)
```

```
##  [1] -0.98207253  0.10556467 -0.76017753  1.05433457  1.49832921 -2.53356459
##  [7] -0.67643082  0.23844069 -0.07849565  0.58422142
```

```
# this is an R code note
```

You can also add plots right inside the document.

```
x <- 1:10
y <- rnorm(10)
plot(x, y)
```



Things to note: by default the code chunk will show up in your document including the code you wrote, the console output, and whatever plots were created.

Ready for some demography

Some basic R commands and reminders: 1. we assign with `<-` 2. you can coerce data types with `as.*()`

```
x <- 1:10
length(x)
```

```
## [1] 10
```

```
class(x)
```

```
## [1] "integer"
```

```
x
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
as.character(x)
```

```
##  [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10"
```

```
mydf <- data.frame(x, y = rnorm(10), abc = letters[1:10])
dim(mydf)
```

```
## [1] 10  3
```

```
mydf <- data.frame(x, y = rnorm(10))
as.matrix(mydf)
```

```
##         x          y
##  [1,]   1 -0.2298053
##  [2,]   2 -0.8075458
##  [3,]   3 -0.4130224
##  [4,]   4  0.8541509
##  [5,]   5  0.5596928
##  [6,]   6  0.3840209
##  [7,]   7 -0.1421624
##  [8,]   8 -0.8013893
##  [9,]   9 -1.5137267
## [10,]  10  0.2658860
```

What is demography

1. Octavio (demography): science of events across the lifecourse (individual and population levels). There was a Preston call-out (population processes)
2. Elizabeth (population): a group of people that changes based on births, deaths, and migration.

3) John-Jairo study of population changes and their determinants.

Age

Elapsed time since the moment of birth

Can you think of a demographic process for which age is NOT important? That is to say, where the probability of experiencing a transition does not depend on age? I can't

Q: we can define arbitrary age-like things. How do we know which are important? Octavio: hands on, no good priors.

Example of arbitrary time measure: duration between any two random life transitions. e.g. time married, birth interval durations, time since retirement. How do we know what's important variation?

I offered that anything that isn't a flat hazard (risk) is potentially interesting. But that might be a lazy approach.

BUT even a flat hazard might be interesting if studying group inequalities, wherein the shape of the hazard isn't so interesting as is the level.

Cohort

Unless otherwise qualified, *cohort* refers to the group of people born in a defined period. Example: the 2020 birth cohort. BUT more generally it refers to a group of people experiencing the same thing at the same time (HT Momoko). We imagined several kinds of cohorts related to the current pandemic.

Period

Period refers to calendar time, it's the moving dot on a timeline.

Age-period-cohort

These three time measures consist in an event date, in a moving date (period), and in a moving elapsed time between them, i.e. a duration (age). These three measures form an identity, meaning that if you know two of them you get the other one automatically. Insert brief note that it's hard or impossible to separate these three things in an empirically useful way. Go exploring there when you have more time.

Let's talk about denominators

Probabilities are a bit clearer than rates, and we ought to know how to think about them. Maybe even intuitively. Probabilities are bounded between and including zero and one.

Rates: put person years of exposure in the denominator. This is almost always approximated.

1. Take average of population before and after (multiplied by period interval)
2. Use the mid-interval population estimate (sometimes published as such). Examples: July 1 estimates, or UN WPP mid-interval estimates.
3. You can see other more sophisticated exposure approximations in the HMD methods protocol.

Let's look at some basic measures

`Ctrl + Alt + i` is the shortcut for pipes!

`read_delim()` reads files with any kind of separator. We have fixed width files using spaces as fillers, so we declare `delim = " "`, and we wipe out the other white space using `trim_ws = TRUE`. We ensure age is numeric all over with `Age = parse_number(Age)` (removes `+` and `-` signs in the data). `mutate()` creates new columns, potentially based on columns already in the data, and potentially many at once (comma separated). We ensure structural variables (`Age`, `Sex`) follow the same codes all over. For deaths and population, `Sex` was given spread over columns, and we stack these using `pivot_longer()`.

```
# install.packages("tidyverse")
# install.packages("readr")
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.2     v dplyr   1.0.7
## v tidyr   1.1.3     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(readr)

# First get births
B2014 <-
  read_delim("BirthsSpain2014.txt",
             trim_ws = TRUE,
             delim = " ") %>%
  mutate(Sex = "Total",
         Age = parse_number(Age)) %>%
  select(Year, Age, Sex, Births = Total)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Year = col_double(),
##   Age = col_character(),
##   Total = col_double()
## )
```

```
# now get deaths
D2014 <-
  read_delim("DeathsSpain2014.txt",
             trim_ws = TRUE,
             delim = " ") %>%
    pivot_longer(Female:Total,
                 names_to = "Sex",
                 values_to = "Deaths") %>%
    mutate(Age = parse_number(Age))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Year = col_double(),
##   Age = col_character(),
##   Female = col_double(),
##   Male = col_double(),
##   Total = col_double()
## )
```

```
# now get population
P2014 <-
  read_delim("PopulationSpain2014.txt",
             trim_ws = TRUE,
             delim = " ") %>%
    pivot_longer(Female:Total,
                 names_to = "Sex",
```

```
                values_to = "Population") %>%
    mutate(Age = parse_number(Age))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Year = col_double(),
##   Age = col_character(),
##   Female = col_double(),
##   Male = col_double(),
##   Total = col_double()
## )
```

```
P2015 <-
  read_delim("PopulationSpain2015.txt",
             trim_ws = TRUE,
             delim = " ") %>%
    pivot_longer(Female:Total,
                 names_to = "Sex",
                 values_to = "Population") %>%
    mutate(Age = parse_number(Age))
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Year = col_double(),
##   Age = col_character(),
##   Female = col_double(),
##   Male = col_double(),
##   Total = col_double()
## )
```

Now let's merge these data files into a single file that we can do tidy demography on.

`bind_rows()` does the same as `rbind()`, we stack the population data, then spread out the years over columns using `pivot_wider()`, creating columns `P2014` and `P2015`, cuz I felt like it. We then join deaths and births. Note `left_join()` respects the data being joined *to*, so births (`B2014`) just matches to the ages in the data and to `Sex = "Total"`.

```
ES2014 <-
  P2014 %>%
  bind_rows(P2015) %>%
  pivot_wider(names_from = Year,
              values_from = "Population",
              names_prefix = "P") %>%
  left_join(D2014) %>%
  left_join(B2014) %>%
  arrange(Sex, Age)
```

```
## Joining, by = c("Age", "Sex")
```

```
## Joining, by = c("Age", "Sex", "Year")
```

looking at the population structure

A population pyramid, by request. See more details for nicer labels in the handout.

`Ctrl + Shift + m` for pipe shortcut

```
ES2014 %>%
  select(Sex, Age, Population = P2015) %>%
  filter(Sex != "Total") %>%
  mutate(Population = ifelse(Sex == "Male", -Population, Population)) %>%   ggplot(aes(x = Age, y = Popu
  geom_bar(stat = "identity") +
  coord_flip()
```

On with rates

For rates, we calculate exposures by averaging populations. This is why we wanted 2014 and 2015 populations side by side. `summarize()` is an aggregation function, so we calculate the marginal sums of deaths and exposures, which reduces the data in this case to a single row. Finally, we calculate a new variable for the crude death rate, `CDR`. We talked about whether crude death rates are comparable at all. Answer: not much!

```
ES2014 %>%
  filter(Sex == "Total") %>%
  mutate(Exposure = (P2014 + P2015) / 2) %>%
  summarize(Deaths = sum(Deaths),
            Exposure = sum(Exposure)) %>%
  mutate(CDR = Deaths / Exposure)
```

```
## # A tibble: 1 x 3
##   Deaths Exposure     CDR
##    <dbl>    <dbl>   <dbl>
## 1 393734 46480971 0.00847
```

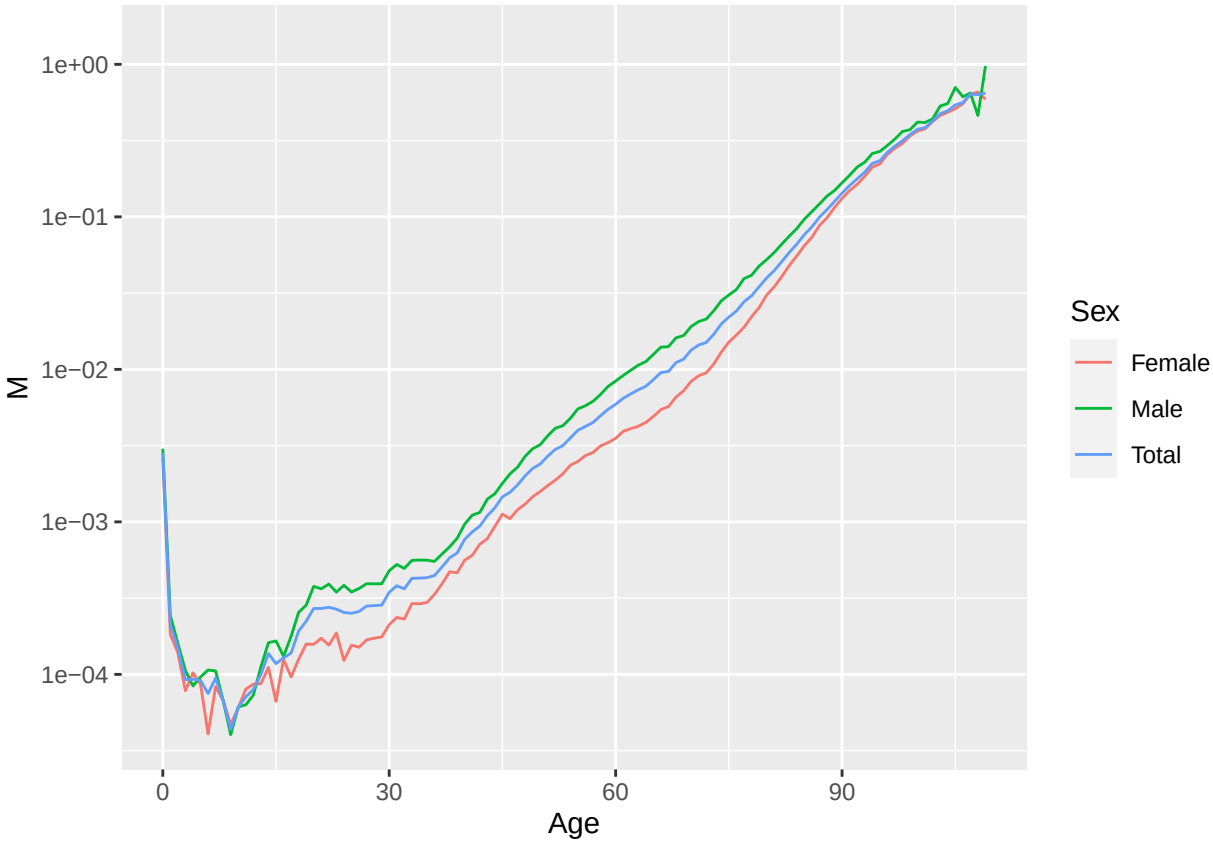
Calculate death rates:

Finally, we calculate death rates, same approach as before, except everythnig can happen within age, which means it can happen all in the same `mutate()` call. Then we plot all three age patterns in log scale.

```
ES2014 %>%
  mutate(Exposure = (P2014 + P2015) / 2,
         M = Deaths / Exposure) %>%
  ggplot(aes(x = Age, y = M, color = Sex)) +
  geom_line() +
  scale_y_log10() +
  xlim(0,109)
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 3 row(s) containing missing values (geom_path).
```



Exercises (optional)

1. Load the data `PopulationSpain.txt` and `BirthsSpain.txt` (it looks like the data we used before except you'll need to `skip` a couple header rows! See `?read_delim`)
2. Calculate age-specific fertility rates from 1950 until 2014 (annually) from age 12 to 55. Skip the year 1975! (The population step will require some thinking. Try making two copies, indicating left and right sides before joining.)
3. Plot the age-specific rates in 1950 and 2014. (in `aes()` try adding `group = Year` or something like that?)
4. Calculate the TFR at each year. (`group_by(Year) %>% summarize(TFR = sum(ASFR))`)

5. Plot TFR over time.
6. Calculate the mean age at birth over time (see example in handout).
7. Plot the mean age over time.