

Finishing up with tidy data

*Daniel Anderson
Week 7, Class 2*



Agenda

- Discuss `gather` and `spread` more

Learning objectives for today

- Get more comfortable with moving from wide to long
- Understand when you may need to "overgather", and how to best approach this

Agenda

- Discuss `gather` and `spread` more

What questions do you have?

Revisiting git

Talk with neighbor. What do these terms mean?

- clone
- pull
- stage
- commit
- push
- repo
- remote

4 / 34

5 / 34

Read in the hiv data

Note, you'll need to add `header = TRUE` to `import` to get it to treat the first row as column names instead of data.

- Make some manipulations (just `{dplyr}` variety) to make the data look like the below.
- Are these data tidy?
- How would you add a new variable - say, percent aids?
- Discuss how you would transform these data, if at all.

```
## # A tibble: 3 x 5
##   country      `1979` `1989` `1999` `2009`
##   <chr>        <dbl>  <lgl>   <dbl>  <dbl>
## 1 France       NA     NA      0.3    0.4
## 2 South Africa NA     NA     14.8   17.2
## 3 United States 0.03176408 NA      0.5    0.6
```

6 / 34

Reasons to prefer tidy data

- More efficient storage and retrieval
- More scalable
- The *tidyverse* of tools will work much more efficiently

5 / 34

How do we tidy these data?

- Use the `gather()` function from `tidyverse`

```
hiv_tidy <- hiv %>%
  gather(year, percentage, -1)
head(hiv_tidy)

## # A tibble: 6 x 3
##   country      year  percentage
##   <chr>        <chr>     <dbl>
## 1 France      1979     NA
## 2 South Africa 1979     NA
## 3 United States 1979  0.03176408
## 4 France      1989     NA
## 5 South Africa 1989     NA
## 6 United States 1989     NA
```

7 / 34

How do we tidy these data?

- Use the `gather()` function from *tidyverse*

```
hiv_tidy <- hiv %>%
  gather(year, percentage, -1)
head(hiv_tidy)

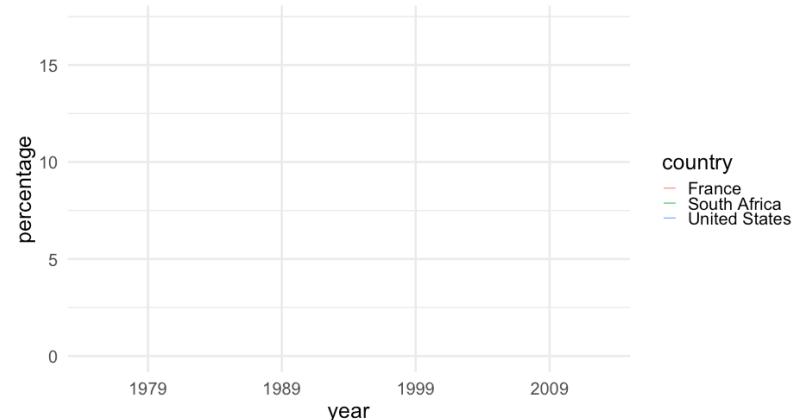
## # A tibble: 6 x 3
##   country     year   percentage
##   <chr>      <chr>     <dbl>
## 1 France     1979     NA
## 2 South Africa 1979     NA
## 3 United States 1979  0.03176408
## 4 France     1989     NA
## 5 South Africa 1989     NA
## 6 United States 1989     NA
```

We still have a problem here... What is it?

7 / 34

Trying to plot

```
ggplot(hiv_tidy, aes(year, percentage, color = country)) +
  geom_line()
```



8 / 34

Redefine year as numeric

```
library(magrittr)
hiv_tidy %>%
  mutate(year = parse_number(year))

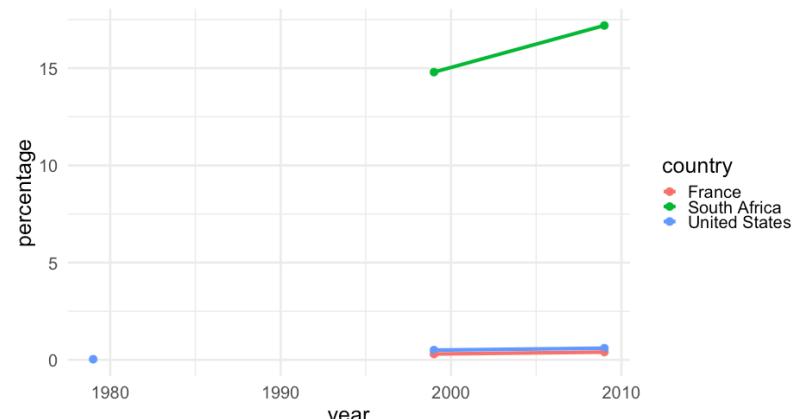
head(hiv_tidy)

## # A tibble: 6 x 3
##   country     year   percentage
##   <chr>      <dbl>     <dbl>
## 1 France     1979     NA
## 2 South Africa 1979     NA
## 3 United States 1979  0.03176408
## 4 France     1989     NA
## 5 South Africa 1989     NA
## 6 United States 1989     NA
```

9 / 34

Trying to plot again

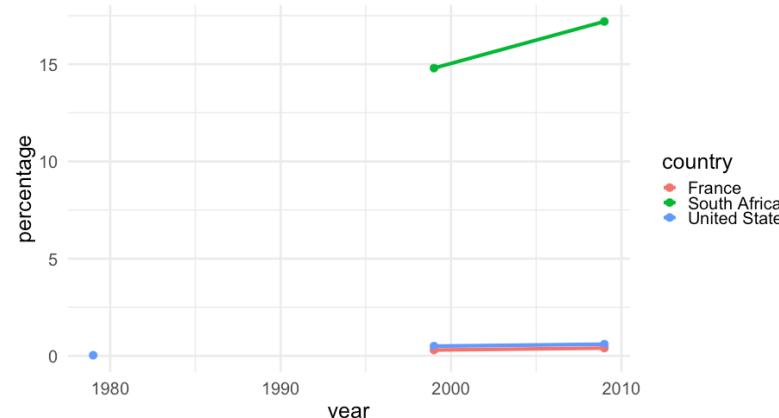
```
ggplot(hiv_tidy, aes(year, percentage, color = country)) +
  geom_point(size = 4) +
  geom_line(size = 2)
```



10 / 34

Trying to plot again

```
ggplot(hiv_tidy, aes(year, percentage, color = country)) +  
  geom_point(size = 4) +  
  geom_line(size = 2)
```



10 / 34

Missing data

```
hiv_tidy
```

```
## # A tibble: 12 x 3  
##   country      year  percentage  
##   <chr>        <dbl>     <dbl>  
## 1 France       1979    NA  
## 2 South Africa 1979    NA  
## 3 United States 1979  0.03176408  
## 4 France       1989    NA  
## 5 South Africa 1989    NA  
## 6 United States 1989    NA  
## 7 France       1999    0.3  
## 8 South Africa 1999   14.8  
## 9 United States 1999   0.5  
## 10 France      2009    0.4  
## 11 South Africa 2009  17.2  
## 12 United States 2009   0.6
```

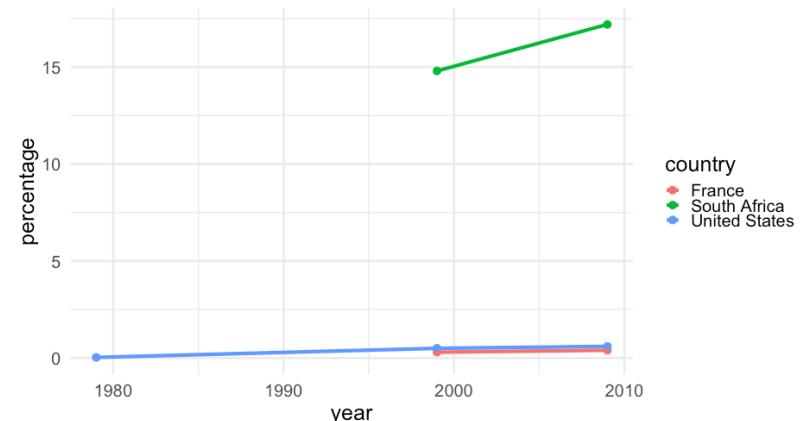
11 / 34

```
hiv_tidy %>%  
  filter(!is.na(percentage))
```

```
## # A tibble: 7 x 3  
##   country      year  percentage  
##   <chr>        <dbl>     <dbl>  
## 1 United States 1979  0.03176408  
## 2 France       1999   0.3  
## 3 South Africa 1999  14.8  
## 4 United States 1999   0.5  
## 5 France       2009   0.4  
## 6 South Africa 2009  17.2  
## 7 United States 2009   0.6
```

12 / 34

```
hiv_tidy %>%  
  filter(!is.na(percentage)) %>%  
  ggplot(aes(year, percentage, color = country)) +  
  geom_point(size = 4) +  
  geom_line(size = 2)
```



13 / 34

How does `gather` work?

arg 3
Columns to Gather

```
# A tibble: 3 x 5
#>   Country `1979` `1989` `1999` `2009`
#>   <chr>     <dbl>    <dbl>    <dbl>    <dbl>
#> 1 France      NA      NA      0.3      0.4
#> 2 South Africa NA      NA     14.8     17.2
#> 3 United States 0.0318  NA      0.5      0.6
```

gather(year, percentage, -1)

```
# A tibble: 12 x 3
#>   Country year  percentage
#>   <chr>    <chr>    <dbl>
#> 1 France   1979      NA
#> 2 South Africa 1979      NA
#> 3 United States 1979  0.0318
#> 4 France   1989      NA
#> 5 South Africa 1989      NA
#> 6 United States 1989      NA
#> 7 France   1999  0.3000
#> 8 South Africa 1999 14.8000
#> 9 United States 1999  0.5000
#> 10 France  2009  0.4000
#> 11 South Africa 2009 17.2000
#> 12 United States 2009  0.6000
```

14 / 34

Pop quiz

- What would have happened if I didn't use `-1` to define the columns to gather?

Country is part of the gather

```
hiv %>%
  gather(year, percentage)
```

```
## # A tibble: 15 x 2
##   year  percentage
##   <chr>    <dbl>
## 1 country France
## 2 country South Africa
## 3 country United States
## 4 1979      <NA>
## 5 1979      <NA>
## 6 1979      0.031764078
## 7 1989      <NA>
## 8 1989      <NA>
## 9 1989      <NA>
## 10 1999     0.3
## 11 1999     14.8
## 12 1999     0.5
## 13 2009     0.4
## 14 2009     17.2
## 15 2009     0.6
```

16 / 34

Declaring columns to gather

- I could have declared the columns to gather other ways. The important part is just being clear which columns should be part of the gather. All of the below are equivalent

```
hiv %>%
  gather(year, percentage, -1)
```

```
hiv %>%
  gather(year, percentage, `1979`, `1989`, `1999`, `2009`)
```

```
hiv %>%
  gather(year, percentage, `1979`:`2009`)
```

```
hiv %>%
  gather(year, percentage, -country)
```

15 / 34

17 / 34

A few more examples (follow along)

- Load the *votes_repub.csv* dataset.

```
votes <- import(here("data", "votes_repub.csv"),
                setclass = "tbl_df")
head(votes)

## # A tibble: 6 x 32
##   V1      X1856 X1860 X1864 X1868 X1872 X1876     X1880 X1884 X1888     X1892
##   <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Alabama NA     NA     NA     51.44 53.19 40.02 36.98000 38.44 32.28  3.95
## 2 Alaska  NA     NA     NA     NA     NA     NA     NA     NA     NA     NA
## 3 Arizona NA     NA     NA     NA     NA     NA     NA     NA     NA     NA
## 4 Arkans.. NA     NA     NA     53.73 52.17 39.88 39.550 40.5   38.07 32.0100
## 5 Califor.. 18.77 32.96 58.63 50.24 56.38 50.88 48.92 52.08 49.95 43.76
## 6 Colora... NA     NA     NA     NA     NA     51.28 54.39 55.31 41.13
## # ... with 21 more variables: X1896 <dbl>, X1900 <dbl>, X1904 <dbl>,
## #   X1908 <dbl>, X1912 <dbl>, X1916 <dbl>, X1920 <dbl>, X1924 <dbl>,
## #   X1928 <dbl>, X1932 <dbl>, X1936 <dbl>, X1940 <dbl>, X1944 <dbl>,
## #   X1948 <dbl>, X1952 <dbl>, X1956 <dbl>, X1960 <dbl>, X1964 <dbl>,
## #   X1968 <dbl>, X1972 <dbl>, X1976 <dbl>
```

18 / 34

Discuss with neighbor

- What's the first step to tidy these data?
- Second?

Step 1: Name state

- dplyr syntax

```
votes <- votes %>%
  rename(state = V1)
```

20 / 34

Step 2: gather()

```
votes %>%
  gather(year, approval_rating, -1)

## # A tibble: 1,550 x 3
##   state      year approval_rating
##   <chr>      <chr>          <dbl>
## 1 Alabama    X1856           NA
## 2 Alaska     X1856           NA
## 3 Arizona    X1856           NA
## 4 Arkansas   X1856           NA
## 5 California X1856          18.77
## 6 Colorado   X1856           NA
## 7 Connecticut X1856          53.18
## 8 Delaware   X1856           2.11
## 9 Florida    X1856           NA
## 10 Georgia   X1856           NA
## # ... with 1,540 more rows
```

19 / 34

21 / 34

Clean up some

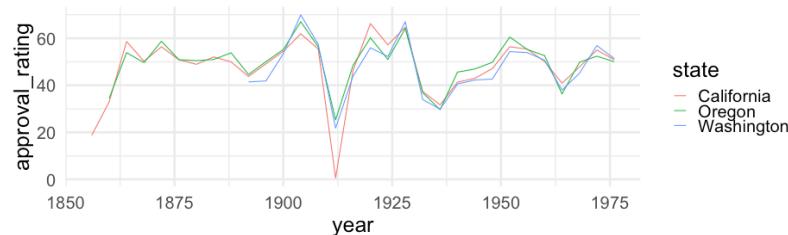
```
votes_tidy <- votes %>%
  gather(year, approval_rating, -1) %>%
  mutate(year = parse_number(year))
votes_tidy
```

```
## # A tibble: 1,550 x 3
##   state      year approval_rating
##   <chr>     <dbl>        <dbl>
## 1 Alabama    1856        NA
## 2 Alaska     1856        NA
## 3 Arizona    1856        NA
## 4 Arkansas   1856        NA
## 5 California 1856     18.77
## 6 Colorado   1856        NA
## 7 Connecticut 1856     53.18
## 8 Delaware   1856      2.11
## 9 Florida    1856        NA
## 10 Georgia   1856        NA
## # ... with 1,540 more rows
```

22 / 34

Plot

```
pd <- votes_tidy %>%
  filter(state == "California" |
        state == "Oregon" |
        state == "Washington")
ggplot(pd, aes(year, approval_rating, color = state)) +
  geom_line()
```



24 / 34

Exploratory plot

- Can you plot approval ratings by year for the west coast (California, Oregon, and Washington)?
- What do you observe?

23 / 34

More complex

- Load the affairs data
- Discuss with a partner/your table
 - What are the variables?
 - What needs to happen to make it tidy?

```
affairs <- import(here("data", "affairs.csv"),
                  setclass = "tbl_df")
head(affairs, n = 3)
```

```
## # A tibble: 3 x 19
##   V1 naffairs kids vryunhap unhappy avgmarr hapavg vryhyp antirel notrel
##   <int>    <int> <int>    <int> <int>    <int> <int>    <int> <int>
## 1     1        0     0        0     0        0     1     0     0     0
## 2     2        0     0        0     0        0     1     0     0     0
## 3     3        3     0        0     0        0     1     0     0     0
## # ... with 9 more variables: slghtrel <int>, smerel <int>, vryrel <int>,
## #   yrsmarri1 <int>, yrsmarri2 <int>, yrsmarri3 <int>, yrsmarri4 <int>,
## #   yrsmarri5 <int>, yrsmarri6 <int>
```

25 / 34

Variables

- Number of affairs: `naffairs`
- Have kids or not: `kids`
- Marriage happiness: `vryunhap:vryhap`
- Religious: `antirel:vryrel`
- Years married: `yrsmarried1:yrsmarried6`

26 / 34

gather

```
affairs %>%
  gather(var, val, -id, -naffairs, -kids)

## # A tibble: 9,616 x 5
##       id naffairs kids var           val
##   <int>    <int> <int> <chr>     <int>
## 1     1        0     0 marhap_vryunhap     0
## 2     2        0     0 marhap_vryunhap     0
## 3     3        3     0 marhap_vryunhap     0
## 4     4        0     1 marhap_vryunhap     0
## 5     5        3     1 marhap_vryunhap     0
## 6     6        0     1 marhap_vryunhap     0
## 7     7        0     0 marhap_vryunhap     0
## 8     8        0     0 marhap_vryunhap     0
## 9     9        7     1 marhap_vryunhap     0
## 10   10       0     0 marhap_vryunhap     0
## # ... with 9,606 more rows
```

28 / 34

Lots of ways to do this

- One way: `rename, gather, separate, filter, spread` For renaming I'm using the base function and you can probably see why I prefer it in this case.

```
names(affairs)[1] <- "id"
names(affairs)[4:8] <- paste0("marhap_", names(affairs)[4:8])
names(affairs)[9:13] <- paste0("rel_", names(affairs)[9:13])
names(affairs)[14:19] <- paste0("yrs_", names(affairs)[14:19])
head(affairs, n = 3)

## # A tibble: 3 x 19
##       id naffairs kids marhap_vryunhap marhap_unhap marhap_avgmarr
##   <int>    <int> <int>          <int>      <int>          <int>
## 1     1        0     0              0          0          0
## 2     2        2     0              0          0          0
## 3     3        3     3              0          0          0
## # ... with 13 more variables: marhap_hapavg <int>, marhap_vryhap <int>,
## #   rel_antirel <int>, rel_notrel <int>, rel_slgthrel <int>,
## #   rel_smerel <int>, rel_vryrel <int>, yrs_yrsmarr1 <int>,
## #   yrs_yrsmarr2 <int>, yrs_yrsmarr3 <int>, yrs_yrsmarr4 <int>,
## #   yrs_yrsmarr5 <int>, yrs_yrsmarr6 <int>
```

27 / 34

separate

```
affairs %>%
  gather(var, val, -id, -naffairs, -kids) %>%
  separate(var, c("var", "char_val"))

## # A tibble: 9,616 x 6
##       id naffairs kids var   char_val  val
##   <int>    <int> <int> <chr> <chr> <int>
## 1     1        0     0 marhap vryunhap     0
## 2     2        0     0 marhap vryunhap     0
## 3     3        3     0 marhap vryunhap     0
## 4     4        0     1 marhap vryunhap     0
## 5     5        3     1 marhap vryunhap     0
## 6     6        0     1 marhap vryunhap     0
## 7     7        0     0 marhap vryunhap     0
## 8     8        0     0 marhap vryunhap     0
## 9     9        7     1 marhap vryunhap     0
## 10   10       0     0 marhap vryunhap     0
## # ... with 9,606 more rows
```

29 / 34

Filter

```
affairs %>%  
  gather(var, val, -id, -naffairs, -kids) %>%  
  separate(var, c("var", "char_val")) %>%  
  filter(val == 1)  
  
## # A tibble: 1,803 x 6  
##   id naffairs kids var   char_val   val  
##   <int>     <int> <int> <chr> <chr>   <int>  
## 1    49      12     1 marhap vryunhap     1  
## 2    64       0     1 marhap vryunhap     1  
## 3    82       0     1 marhap vryunhap     1  
## 4   138       0     1 marhap vryunhap     1  
## 5   150      12     1 marhap vryunhap     1  
## 6   206       7     1 marhap vryunhap     1  
## 7   215       0     1 marhap vryunhap     1  
## 8   275       3     1 marhap vryunhap     1  
## 9   358       0     1 marhap vryunhap     1  
## 10  371      12     1 marhap vryunhap     1  
## # ... with 1,793 more rows
```

30 / 34

Drop val, spread

```
affairs %>%  
  gather(var, val, -id, -naffairs, -kids) %>%  
  separate(var, c("var", "char_val")) %>%  
  filter(val == 1) %>%  
  select(-val) %>%  
  spread(var, char_val) %>%  
  head()  
  
## # A tibble: 6 x 6  
##   id naffairs kids marhap rel      yrs  
##   <int>     <int> <int> <chr> <chr>   <chr>  
## 1    1       0     0 hapavg slghtrel yrsmarr5  
## 2    2       0     0 hapavg smerel  yrsmarr3  
## 3    3       3     0 hapavg slghtrel yrsmarr2  
## 4    4       0     1 hapavg antirel yrsmarr6  
## 5    5       3     1 vryhap slghtrel yrsmarr3  
## 6    6       0     1 vryhap vryrel  yrsmarr6
```

31 / 34

Cleanup

```
affairs_tidy <- affairs %>%  
  gather(var, val, -id, -naffairs, -kids) %>%  
  separate(var, c("var", "char_val")) %>%  
  filter(val == 1) %>%  
  select(-val) %>%  
  spread(var, char_val) %>%  
  mutate(yrs = parse_number(yrs))  
  
head(affairs_tidy)
```

```
## # A tibble: 6 x 6  
##   id naffairs kids marhap rel      yrs  
##   <int>     <int> <int> <chr> <chr>   <dbl>  
## 1    1       0     0 hapavg slghtrel     5  
## 2    2       0     0 hapavg smerel      3  
## 3    3       3     0 hapavg slghtrel     2  
## 4    4       0     1 hapavg antirel     6  
## 5    5       3     1 vryhap slghtrel     3  
## 6    6       0     1 vryhap vryrel      6
```

32 / 34

Notice what we've done?

- Essentially, in this case, we've removed the dummy-coding
 - One of the most common ways data are not tidy

33 / 34

Lab