Tidying data: extras

Packages

```
library(tidyverse)
```

The pig feed data again

```
my_url <- "http://ritsokiguess.site/datafiles/pigs1.txt"
pigs <- read_table(my_url)
pigs
```

```
# A tibble: 5 x 5
    pig feed1 feed2 feed3 feed4
  <dbl> <dbl> <dbl> <dbl> <dbl>
1     1  60.8  68.7  92.6  87.9
2     2  57    67.7  92.1  84.2
3     3  65    74    90.2  83.1
4     4  58.6  66.3  96.5  85.7
5     5  61.7  69.8  99.1  90.3
```

Make longer (as before)

```
pigs %>% pivot_longer(-pig, names_to="feed",
                             values_to="weight") -> pigs_longer
pigs_longer
```

```
# A tibble: 20 x 3
     pig feed  weight
   <dbl> <chr>  <dbl>
 1     1 feed1   60.8
 2     1 feed2   68.7
 3     1 feed3   92.6
 4     1 feed4   87.9
 5     2 feed1   57
 6     2 feed2   67.7
 7     2 feed3   92.1
 8     2 feed4   84.2
 9     3 feed1   65
10     3 feed2   74
```

Make wider two ways 1/2

`pivot_wider` is inverse of `pivot_longer`:

```
pigs_longer %>%
  pivot_wider(names_from=feed, values_from=weight)
```

```
# A tibble: 5 x 5
    pig feed1 feed2 feed3 feed4
  <dbl> <dbl> <dbl> <dbl> <dbl>
1     1  60.8  68.7  92.6  87.9
2     2  57    67.7  92.1  84.2
3     3  65    74    90.2  83.1
4     4  58.6  66.3  96.5  85.7
5     5  61.7  69.8  99.1  90.3
```

we are back where we started.

Make wider 2/2

Or

```
pigs_longer %>%
  pivot_wider(names_from=pig, values_from=weight)
```

```
# A tibble: 4 x 6
  feed    `1`   `2`   `3`   `4`   `5`
  <chr> <dbl> <dbl> <dbl> <dbl> <dbl>
1 feed1  60.8  57    65    58.6  61.7
2 feed2  68.7  67.7  74    66.3  69.8
3 feed3  92.6  92.1  90.2  96.5  99.1
4 feed4  87.9  84.2  83.1  85.7  90.3
```

but:

```
pigs_longer %>%
  pivot_wider(names_from=pig, values_from=weight) %>%
  select(2)
```

Disease presence and absence at two locations

Frequencies of plants observed with and without disease at two locations:

```
Species     Disease present         Disease absent
       Location X Location Y  Location X Location Y
A            44         12          38        10
B            28         22          20        18
```

This has two rows of headers, so I rewrote the data file:

```
Species  present_x present_y    absent_x  absent_y
A            44         12          38        10
B            28         22          20        18
```

Read in

… into data frame called `prevalence`:

```
my_url <- "http://ritsokiguess.site/STAC32/disease.txt"
prevalence <- read_table(my_url)
prevalence
```

```
# A tibble: 2 x 5
  Species present_x present_y absent_x absent_y
  <chr>       <dbl>     <dbl>    <dbl>    <dbl>
1 A              44        12       38       10
2 B              28        22       20       18
```

Lengthen and separate

```
prevalence %>%
  pivot_longer(-Species, names_to = "column",
               values_to = "freq") %>%
  separate_wider_delim(column, "_",
                       names = c("disease", "location"))
```

```
# A tibble: 8 x 4
  Species disease location  freq
  <chr>   <chr>   <chr>    <dbl>
1 A       present x           44
2 A       present y           12
3 A       absent  x           38
4 A       absent  y           10
5 B       present x           28
6 B       present y           22
7 B       absent  x           20
8 B       absent  y           18
```

Making longer, the better way

```
prevalence %>%
  pivot_longer(-Species, names_to=c("disease", "location"),
               names_sep="_",
               values_to="frequency") -> prevalence_longer
prevalence_longer
```

```
# A tibble: 8 x 4
  Species disease location frequency
  <chr>   <chr>   <chr>        <dbl>
1 A       present x               44
2 A       present y               12
3 A       absent  x               38
4 A       absent  y               10
5 B       present x               28
6 B       present y               22
7 B       absent  x               20
8 B       absent  y               18
```

Making wider, different ways 1/2

```
prevalence_longer %>%
  pivot_wider(names_from=c(Species, location),
              values_from=frequency)
```

```
# A tibble: 2 x 5
  disease    A_x   A_y   B_x   B_y
  <chr>    <dbl> <dbl> <dbl> <dbl>
1 present     44    12    28    22
2 absent      38    10    20    18
```

Making wider, different ways 2/2

```
prevalence_longer %>%
  pivot_wider(names_from=location, values_from=frequency)
```

```
# A tibble: 4 x 4
  Species disease     x     y
  <chr>   <chr>   <dbl> <dbl>
1 A       present    44    12
2 A       absent     38    10
3 B       present    28    22
4 B       absent     20    18
```

Interlude

Pigs data again:

```
pigs_longer %>%
  group_by(feed) %>%
  summarize(weight_mean=mean(weight))
```

```
# A tibble: 4 x 2
  feed  weight_mean
  <chr>       <dbl>
1 feed1        60.6
2 feed2        69.3
3 feed3        94.1
4 feed4        86.2
```

What if summary is more than one number?

eg. quartiles:

```
pigs_longer %>%
  group_by(feed) %>%
  summarize(r = quantile(weight, c(0.25, 0.75)))
```

```
Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated
dplyr 1.1.0.
i Please use `reframe()` instead.
i When switching from `summarise()` to `reframe()`, remember that `reframe()`
  always returns an ungrouped data frame and adjust accordingly.

`summarise()` has grouped output by 'feed'. You can override using the
`.groups` argument.

# A tibble: 8 x 2
# Groups:   feed [4]
  feed      r
  <chr> <dbl>
1 feed1  58.6
2 feed1  61.7
3 feed2  67.7
4 feed2  69.8
5 feed3  92.1
6 feed3  96.5
```

Following the hint (gives no warning)

```
pigs_longer %>%
  group_by(feed) %>%
  reframe(r=quantile(weight, c(0.25, 0.75)))
```

```
# A tibble: 8 x 2
  feed      r
  <chr> <dbl>
1 feed1  58.6
2 feed1  61.7
3 feed2  67.7
4 feed2  69.8
5 feed3  92.1
6 feed3  96.5
7 feed4  84.2
8 feed4  87.9
```

this also works

```
pigs_longer %>%
  group_by(feed) %>%
  summarize(r=list(quantile(weight, c(0.25, 0.75)))) %>%
  unnest(r)
```

```
# A tibble: 8 x 2
  feed      r
  <chr> <dbl>
1 feed1  58.6
2 feed1  61.7
3 feed2  67.7
4 feed2  69.8
5 feed3  92.1
6 feed3  96.5
7 feed4  84.2
8 feed4  87.9
```

Or, even better, use `enframe`:

```
quantile(pigs_longer$weight, c(0.25, 0.75))
```

```
   25%    75% 
65.975 90.225 
```

```
enframe(quantile(pigs_longer$weight, c(0.25, 0.75)))
```

```
# A tibble: 2 x 2
  name  value
  <chr> <dbl>
1 25%    66.0
2 75%    90.2
```

A nice look

Run this one line at a time to see how it works:

```
pigs_longer %>%
  group_by(feed) %>%
  summarize(r=list(enframe(quantile(weight, c(0.25, 0.75))))) %>%
  unnest(r) %>%
  pivot_wider(names_from=name, values_from=value) -> d
d
```

```
# A tibble: 4 x 3
  feed  `25%` `75%`
  <chr> <dbl> <dbl>
1 feed1  58.6  61.7
2 feed2  67.7  69.8
3 feed3  92.1  96.5
4 feed4  84.2  87.9
```

A hairy one

18 people receive one of three treatments. At 3 different times (pre, post, followup) two variables `y` and `z` are measured on each person:

```
my_url <- "http://ritsokiguess.site/STAC32/repmes.txt"
repmes0 <- read_table(my_url)
repmes0
```

```
# A tibble: 18 x 8
   treatment   rep pre_y post_y  fu_y pre_z post_z  fu_z
   <chr>     <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl>
 1 A             1     3     13     9     0      0     9
 2 A             2     0     14    10     6      6     3
 3 A             3     4      6    17     8      2     6
 4 A             4     7      7    13     7      6     4
 5 A             5     3     12    11     6     12     6
 6 A             6    10     14     8    13      3     8
 7 B             1     9     11    17     8     11    27
 8 B             2     4     16    13     9      3    26
```

Create unique ids

```
repmes0 %>% mutate(id=str_c(treatment, ".", rep)) %>%
  select(-rep) %>%
  select(id, everything()) -> repmes
repmes
```

```
# A tibble: 18 x 8
   id      treatment pre_y post_y  fu_y pre_z post_z  fu_z
   <chr>   <chr>     <dbl>  <dbl> <dbl> <dbl>  <dbl> <dbl>
 1 A.1     A             3     13     9     0      0     9
 2 A.2     A             0     14    10     6      6     3
 3 A.3     A             4      6    17     8      2     6
 4 A.4     A             7      7    13     7      6     4
 5 A.5     A             3     12    11     6     12     6
 6 A.6     A            10     14     8    13      3     8
 7 B.1     B             9     11    17     8     11    27
 8 B.2     B             4     16    13     9      3    26
 9 B.3     B             8     10     9    12      0    18
```

Attempt 1

```
repmes %>% pivot_longer(contains("_"),
                        names_to=c("time", "var"),
                        names_sep="_",
                        values_to = "vvv"
                         )
```

```
# A tibble: 108 x 5
   id    treatment time  var     vvv
   <chr> <chr>     <chr> <chr> <dbl>
 1 A.1   A         pre   y         3
 2 A.1   A         post  y        13
 3 A.1   A         fu    y         9
 4 A.1   A         pre   z         0
 5 A.1   A         post  z         0
 6 A.1   A         fu    z         9
 7 A.2   A         pre   y         0
 8 A.2   A         post  y        14
```

Comment

This is *too* long! We wanted a column called `y` and a column called `z`, but they have been pivoted-longer too.

Attempt 2

```
repmes %>% pivot_longer(contains("_"),
                        names_to=c("time", ".value"),
                        names_sep="_"
                        ) -> repmes3
repmes3
```

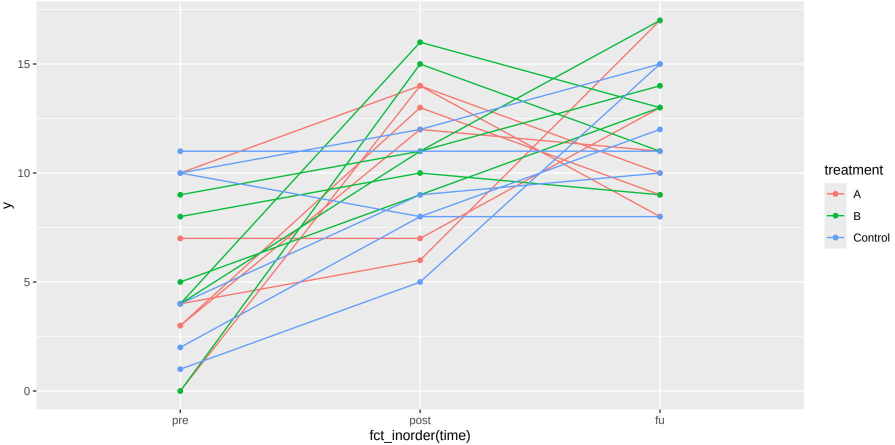
```
# A tibble: 54 x 5
   id    treatment time      y     z
   <chr> <chr>     <chr> <dbl> <dbl>
 1 A.1   A         pre       3     0
 2 A.1   A         post     13     0
 3 A.1   A         fu        9     9
 4 A.2   A         pre       0     6
 5 A.2   A         post     14     6
 6 A.2   A         fu       10     3
 7 A.3   A         pre       4     8
 8 A.3   A         post      6     2
```

Comment

This has done what we wanted.

Make a graph

```
ggplot(repmes3, aes(x=fct_inorder(time), y=y,
                    colour=treatment, group = id)) +
  geom_point() + geom_line()
```

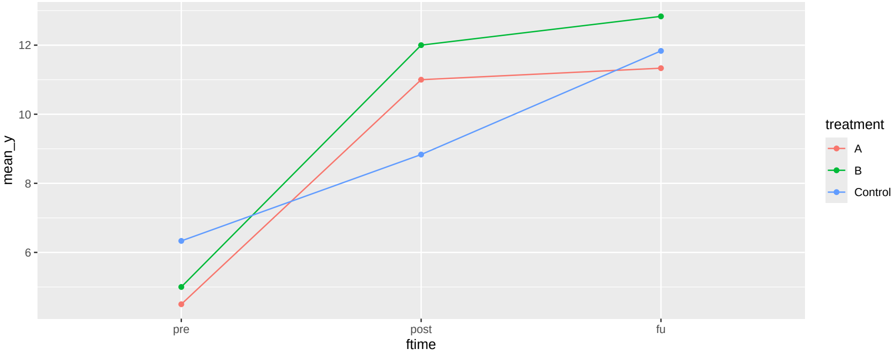


Comment

- A so-called “spaghetti plot”:
    - The three measurements for each person are joined by lines
    - The lines are coloured by treatment.

Or do the plot with means

```
repmes3 %>% group_by(treatment, ftime=fct_inorder(time)) %>%
  summarize(mean_y=mean(y)) %>%
  ggplot(aes(x=ftime, y=mean_y, colour=treatment, group=treatm
    geom_point() + geom_line()
```



Comment

- On average, the two real treatments go up and level off
- but the control group is very different.