Romain François

@romain_francois

rladies montpellier - 2018/12/12

https://github.com/romainfrancois/rap

http://bit.ly/rap-rladies-mtp

Thinking inside the box: you can do that inside a data frame?!

DOWNLOAD MATERIALS

Abstract

The data frame is a crucial data structure in R and, especially, in the tidyverse. Working on a column or a variable is a very natural operation, which is great. But what about row-oriented work? That also comes up frequently and is more awkward. In this webinar I'll work through concrete code examples, exploring patterns that arise in data analysis. We'll discuss the general notion of "split-apply-combine", row-wise work in a data frame, splitting vs. nesting, and list-columns.

About the speaker

Jenny Bryan
Software Engineer, RStudio

Jenny is a recovering biostatistician who takes special delight in eliminating the small agonies of data analysis. She's part of Hadley's team, working on R packages and integrating them into fluid workflows. She's been working in R for over 20 years, serves in the leadership of rOpenSci and Forwards, and is an Ordinary Member of the R Foundation. Jenny is an Associate Professor of Statistics (on leave) at the University of British Columbia, where she created the course STAT 545.

Share:

Why so many ways to do **THING** for each row?

Because there is no way.

download materials: **rstd.io/row-work**

rstd.io/row-work

Why so many ways to do **THING** for each row?

Columns are very special in R.

This is fantastic for data analysis.

Tradeoff: row-oriented work is harder.

download materials: **rstd.io/row-work**

rstd.io/row-work

How to choose?

Speed and ease of:

- **Writing** the code
- **Reading** the code
- **Executing** the code

download materials: **rstd.io/row-work**

rstd.io/row-work

Tips for row-oriented workflows

embrace the **data frame**

esp. the **tibble** = tidyverse data frame

embrace **lists**

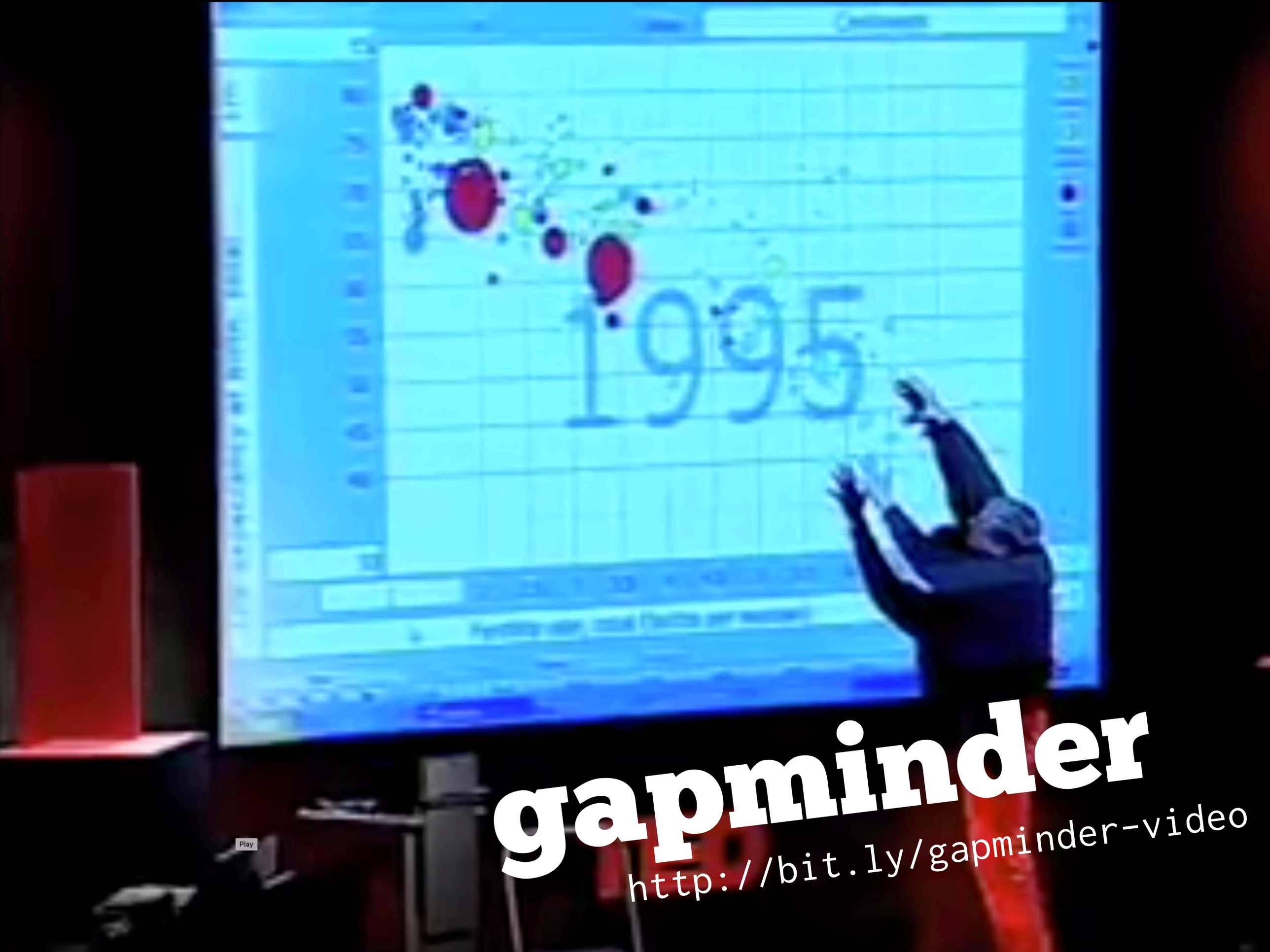
embrace lists as variables in a tibble

"**list-columns**", may come from nesting

embrace **purrr::map()** & friends

download materials: **rstd.io/row-work**

rstd.io/row-work


1995
gapminder
http://bit.ly/gapminder-video
Play

```
library(gapminder)
gapminder
#> # A tibble: 1,704 x 6
#>    country     continent  year lifeExp      pop gdpPercap
#>    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
#>  1 Afghanistan Asia       1952    28.8  8425333      779.
#>  2 Afghanistan Asia       1957    30.3  9240934      821.
#>  3 Afghanistan Asia       1962    32.0 10267083      853.
#>  4 Afghanistan Asia       1967    34.0 11537966      836.
#>  5 Afghanistan Asia       1972    36.1 13079460      740.
#>  6 Afghanistan Asia       1977    38.4 14880372      786.
#>  7 Afghanistan Asia       1982    39.9 12881816      978.
#>  8 Afghanistan Asia       1987    40.8 13867957      852.
#>  9 Afghanistan Asia       1992    41.7 16317921      649.
#> 10 Afghanistan Asia       1997    41.8 22227415      635.
#> # … with 1,694 more rows
```

```r
library(gapminder)
library(tidyverse)

# setup
gap_nested <- gapminder %>%
  filter(continent == "Asia") %>%
  mutate(
    yr1952 = year - 1952,
    country = fct_drop(country)
  ) %>%
  group_nest(country) %>%
  print()
#> # A tibble: 33 x 2
#>    country     data
#>    <fct>       <list>
#>  1 Afghanistan <tibble [12 × 6]>
#>  2 Bahrain     <tibble [12 × 6]>
#>  3 Bangladesh  <tibble [12 × 6]>
#>  4 Cambodia    <tibble [12 × 6]>
#>  5 China       <tibble [12 × 6]>
#> # … with 28 more rows
```

```
# A tibble: 12 x 6
   continent  year lifeExp      pop gdpPercap yr1952
   <fct>     <int>   <dbl>    <int>     <dbl>  <dbl>
 1 Asia       1952    28.8  8425333       779.      0
 2 Asia       1957    30.3  9240934       821.      5
 3 Asia       1962    32.0 10267083       853.     10
 4 Asia       1967    34.0 11537966       836.     15
 5 Asia       1972    36.1 13079460       740.     20
 6 Asia       1977    38.4 14880372       786.     25
 7 Asia       1982    39.9 12881816       978.     30
 8 Asia       1987    40.8 13867957       852.     35
 9 Asia       1992    41.7 16317921       649.     40
10 Asia       1997    41.8 22227415       635.     45
11 Asia       2002    42.1 25268405       727.     50
12 Asia       2007    43.8 31889923       975.     55
```

For each country

- Fit linear model:

```
fit <- lm(lifeExp ~ yr1952, data = .)
```

- Grab intercept and slope as columns

```
coef(fit)["(Intercept)"]
coef(fit)["(slope)"]
```

```
gap_nested %>%
  mutate(
    fit       = map(data, ~ lm(lifeExp ~ yr1952, data = .x)),
    intercept = map_dbl(fit, ~ coef(.x)["(Intercept)"]),
    slope     = map_dbl(fit, ~ coef(.x)["yr1952"])
  )
```

What happens

What is being iterated on

Types of the new columns (empty means list)

Names of the new columns

```
# mutate + *map*
gap_nested %>%
  mutate(
    fit       = map    (data, ~ lm(lifeExp ~ yr1952, data = .x)),
    intercept = map_dbl(fit, ~ coef(.x)["(Intercept)"]),
    slope     = map_dbl(fit, ~ coef(.x)["yr1952"])
  )
#> # A tibble: 33 x 5
#>    country           data               fit      intercept slope
#>    <fct>             <list>             <list>       <dbl> <dbl>
#>  1 Afghanistan       <tibble [12 × 6]> <S3: lm>      29.9 0.275
#>  2 Bahrain           <tibble [12 × 6]> <S3: lm>      52.7 0.468
#>  3 Bangladesh        <tibble [12 × 6]> <S3: lm>      36.1 0.498
#>  4 Cambodia          <tibble [12 × 6]> <S3: lm>      37.0 0.396
#>  5 China             <tibble [12 × 6]> <S3: lm>      47.2 0.531
#>  6 Hong Kong, China <tibble [12 × 6]> <S3: lm>      63.4 0.366
#>  7 India             <tibble [12 × 6]> <S3: lm>      39.3 0.505
#>  8 Indonesia         <tibble [12 × 6]> <S3: lm>      36.9 0.635
#>  9 Iran              <tibble [12 × 6]> <S3: lm>      45.0 0.497
#> 10 Iraq              <tibble [12 × 6]> <S3: lm>      50.1 0.235
#> # … with 23 more rows
```

map()

map_lgl(), map_int(), map_dbl(), map_chr()

map_if(), map_at()

map_dfr(), map_dfc()

map2()

map2_lgl(), map2_int(), map2_dbl(), map2_chr()

map2_dfr(), map2_dfc()

pmap()

pmap_lgl(), pmap_int(), pmap_dbl(), pmap_chr()

pmap_dfr(), pmap_dfc()

imap()

imap_lgl(), imap_chr(), imap_int(), imap_dbl()

imap_dfr(), imap_dfc()

map()

- Iterate on 1 thing:

```
map[_type](<thing>, ~fun(.x))
```

- Iterate on 2 things:

```
map2[_type](<thing1>, <thing2>, ~fun(.x, .y))
```

- Iterate on more than 2 things

```
pmap[_type](
  list(<thing1>, <thing2>, <...>, <thing_n>),
  function(<name1>, <name2>, <...>, <name_n>) {
     fun(<name1>, <name2>, <...>, <name_n>)
  }
)
```

PURRR ENTAL

ADVISORY

IMPLICIT MAPPING

```
# devtools::install_github("romainfrancois/rap")
library(rap)
```

rap() iterates on rows of gap_nested


```
gap_fitted <- gap_nested %>%
  rap(
    fit       =          ~ lm(lifeExp ~ yr1952, data = data),
    intercept = double() ~ coef(fit)[["(Intercept)"]],
    slope     = double() ~ coef(fit)[["yr1952"]]
  )
```

Expression for a single row of gap_nested

Types of the new columns (empty means list)

Names of the new columns

mutate() + *map*()

```
gap_nested %>%
  mutate(
    fit       = map     (data, ~ lm(lifeExp ~ yr1952, data = .x)),
    intercept = map_dbl(fit,  ~ coef(.x)[["(Intercept)"]]),
    slope     = map_dbl(fit,  ~ coef(.x)[["yr1952"]])
  )
```

rap()

```
gap_fitted <- gap_nested %>%
  rap(
    fit       =          ~ lm(lifeExp ~ yr1952, data = data),
    intercept = double() ~ coef(fit)[["(Intercept)"]],
    slope     = double() ~ coef(fit)[["yr1952"]]
  )
```


splice !!!

```
fit1 <- gap_fitted$fit[[1]]
coef <- coef(fit1)
coef
#> (Intercept)      yr1952
#>   29.9072949   0.2753287
tibble(
   `(Intercept)` = coef[["(Intercept)"]],
   yr1952        = coef[["yr1952"]]
)
#> # A tibble: 1 x 2
#>   `(Intercept)` yr1952
#>           <dbl>  <dbl>
#> 1          29.9  0.275
```

```
tibble( !!!coef(fit1) )
#> # A tibble: 1 x 2
#>   `(Intercept)` yr1952
#>           <dbl>  <dbl>
#> 1          29.9  0.275
```

```
gap_fitted <- gap_nested %>%
  rap(
    fit       =              ~ lm(lifeExp ~ yr1952, data = data),
    coefs     = data.frame() ~ tibble(!!!coef(fit))
  )
gap_fitted
#> # A tibble: 33 x 4
#>    country            data                fit      coefs$`(Intercept)` $yr1952
#>    <fct>              <list>              <list>                 <dbl>   <dbl>
#>  1 Afghanistan        <tibble [12 × 6]> <S3: lm>                29.9   0.275
#>  2 Bahrain            <tibble [12 × 6]> <S3: lm>                52.7   0.468
#>  3 Bangladesh         <tibble [12 × 6]> <S3: lm>                36.1   0.498
#>  4 Cambodia           <tibble [12 × 6]> <S3: lm>                37.0   0.396
#>  5 China              <tibble [12 × 6]> <S3: lm>                47.2   0.531
#>  6 Hong Kong, China   <tibble [12 × 6]> <S3: lm>                63.4   0.366
#>  7 India              <tibble [12 × 6]> <S3: lm>                39.3   0.505
#>  8 Indonesia          <tibble [12 × 6]> <S3: lm>                36.9   0.635
#>  9 Iran               <tibble [12 × 6]> <S3: lm>                45.0   0.497
#> 10 Iraq               <tibble [12 × 6]> <S3: lm>                50.1   0.235
#> # … with 23 more rows
```

```r
tbl <- tibble(
  fun     = list(rnorm, runif),
  n       = c(5, 3),
  params  = list(list(mean = 10, sd = 2), list(min = 0, max = 2))
)
tbl
#> # A tibble: 2 x 3
#>   fun         n params
#>   <list> <dbl> <list>
#> 1 <fn>        5 <list [2]>
#> 2 <fn>        3 <list [2]>

tbl %>%
  rap(y = ~ fun(n, !!!params))
#> # A tibble: 2 x 4
#>   fun         n params     y
#>   <list> <dbl> <list>     <list>
#> 1 <fn>        5 <list [2]> <dbl [5]>
#> 2 <fn>        3 <list [2]> <dbl [3]>

tbl %>%
  wap(~ fun(n, !!!params))
#> [[1]]
#> [1]  6.132587 10.464439 12.266975 12.890391 11.039346
#>
#> [[2]]
#> [1] 0.3873068 1.8211390 1.1559047
```

Romain François

@romain_francois

rladies montpellier - 2018/12/12

https://github.com/romainfrancois/rap

http://bit.ly/rap-rladies-mtp