Data Science for Everyone – Iteration

Dr. Ab Mosca (they/them)

Slides based off slides courtesy of Jordan Crouser (https://jcrouser.github.io/)

Plan for Today

- XX

Iteration

- Suppose you have `f(val)`, which returns a value, and you want to use this function on values `1 - 100`

Iteration

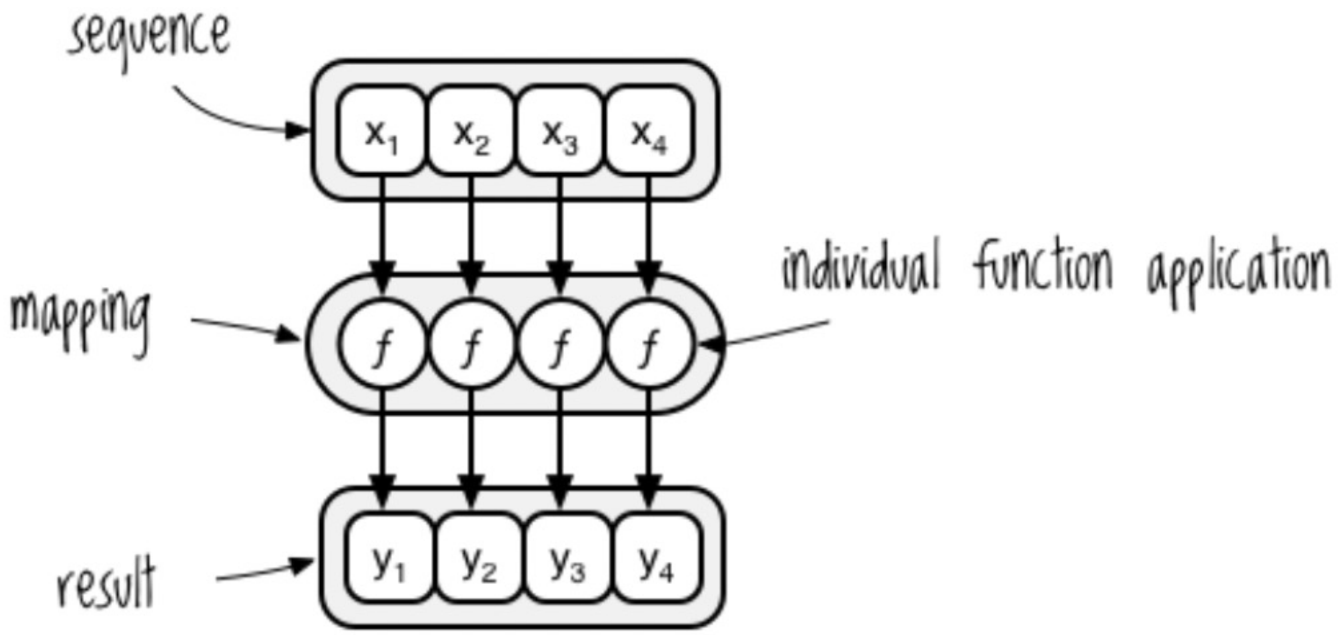
- Suppose you have `f(val)`, which returns a value, and you want to use this function on values `1 - 100`
- One option for doing this would be to call `f(val)` for `1 - 100`
  - `f(1)`
  - `f(2)`
  - `f(3)`
  - `f(4)`
  - `f(4)`
  - `f(6)`
  - `...okay, I’m already bored`

Iteration

- Suppose you have `f(val)`, which returns a value, and you want to use this function on a vector of values `1 – 100`
- One option for doing this would be to call `f(val)` for `1 – 100`
  - `f(1)`
  - `f(2)`
  - `f(3)`
  - `f(4)`
  - `f(4)`
  - `f(6)`
  - `...okay, I'm already bored`
- Uh-oh…not only am I bored, but
  - I accidentally called `f(4)` twice and skipped `f(5)`
  - and I'm going to get all my results separately, but I'd really like them in a vector

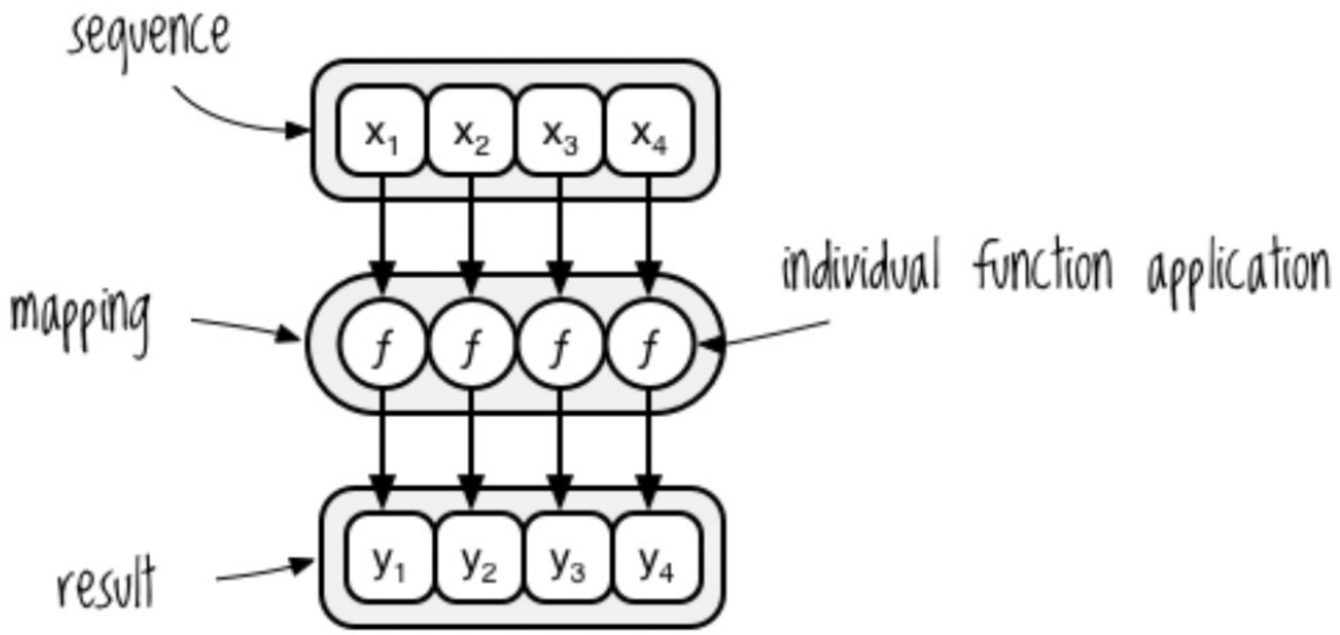
Iteration

- What if instead, we could apply `f()` to our vector of values, and get a vector of results?



Iteration

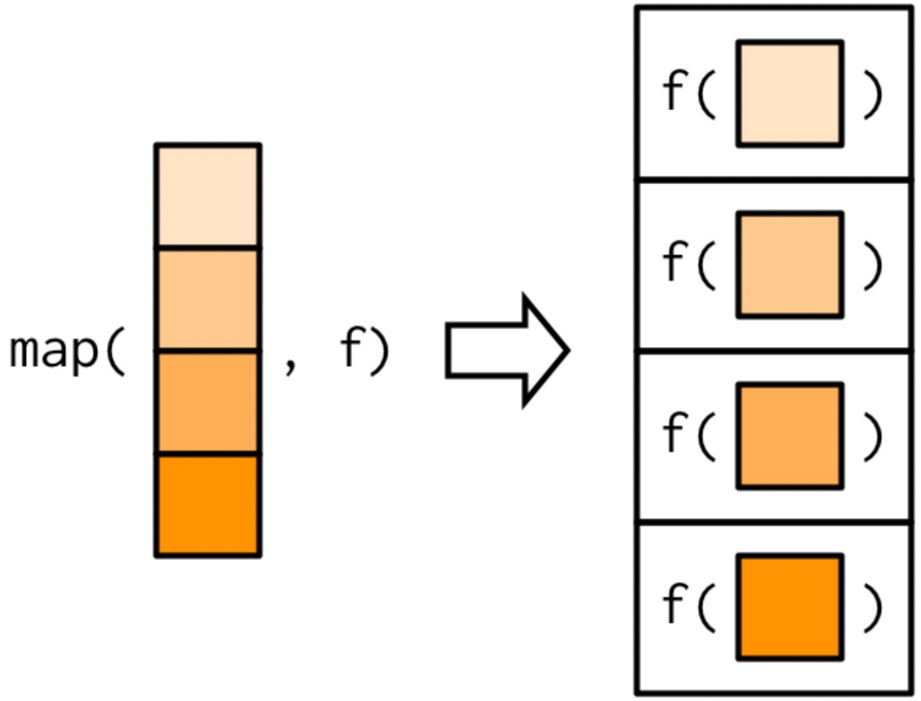
- What if instead, we could apply `f()` to our vector of values, and get a vector of results?



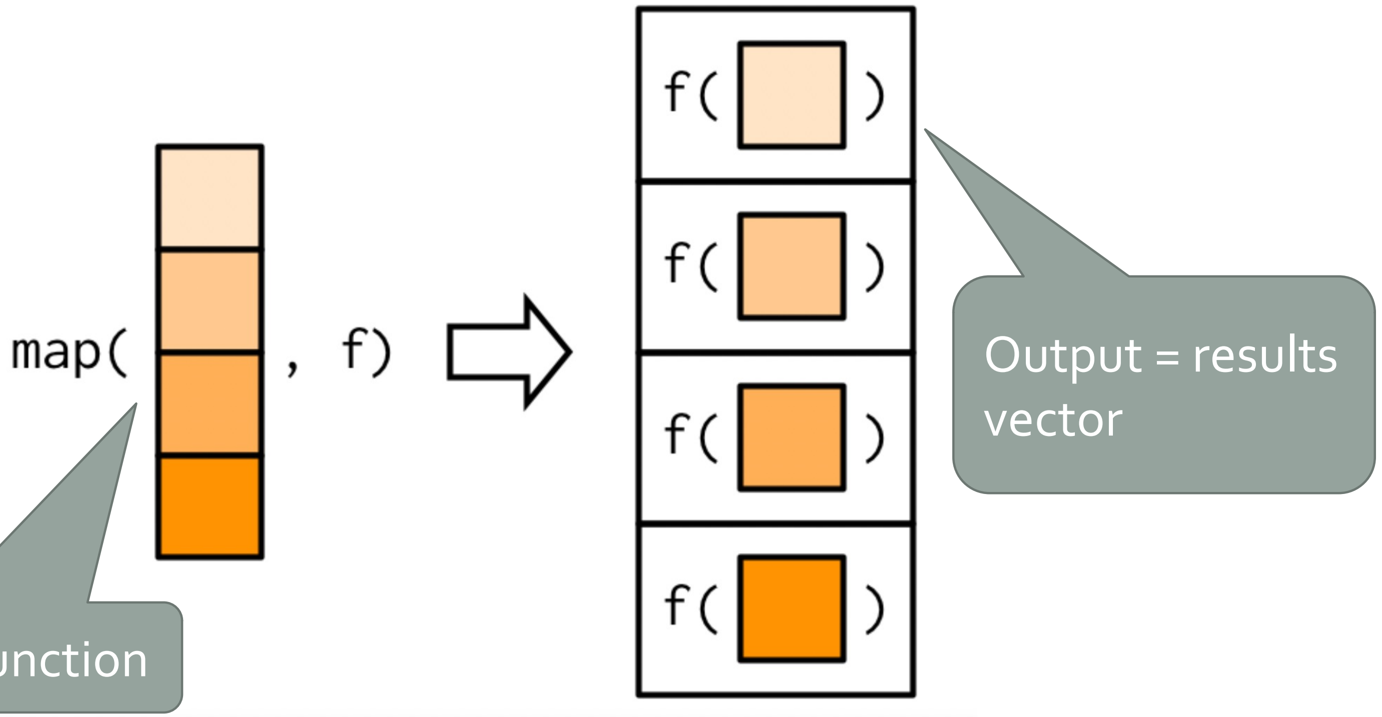
- We call this mapping, and it falls into the category of something called functional programming

- The library `purrr` makes functional programming with R easier
- Find the `purr` cheatsheet here: https://www.rstudio.com/resources/cheatsheets/

- The library `purrr` makes functional programming with R easier
- Find the `purr` cheatsheet here: https://www.rstudio.com/resources/cheatsheets/
- We will mostly use the `map()` function



- The library `purrr` makes functional programming with R easier
- Find the `purr` cheatsheet here: https://www.rstudio.com/resources/cheatsheets/
- We will mostly use the `map()` function



map()

- `map()` example

```r
# vector of values
words <- c("alphabet", "bunny", "cathedral")

# iterate the function over values
# return a list of number of characters per word
map(words, nchar)
```

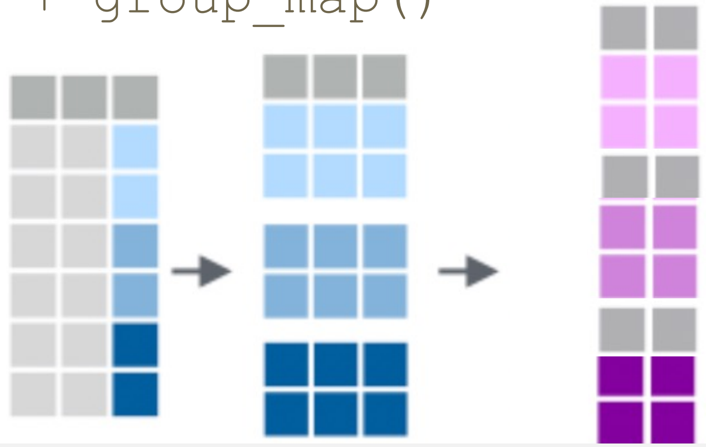
```
## [[1]]
## [1] 8
##
## [[2]]
## [1] 5
##
## [[3]]
## [1] 9
```

```r
# iterate the function over values
# return a vector of number of characters per word
map_int(words, nchar)
```

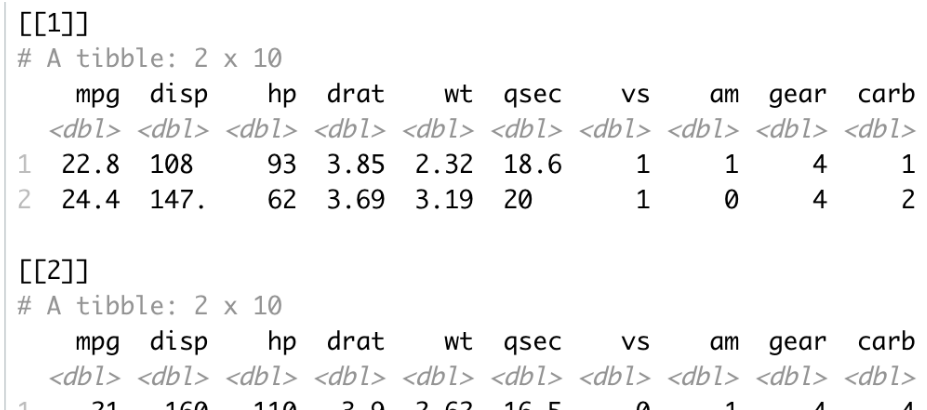
```
## [1] 8 5 9
```

Mapping with and grouping

- We can use `group_by()` and apply different mapping functions to groups within our datasets
- Ex. `group_by() + group_map()`



```
mtcars %>%
    group_by(cyl) %>% # creates a *list* of data frames!
    group_map(head, n = 2)
```

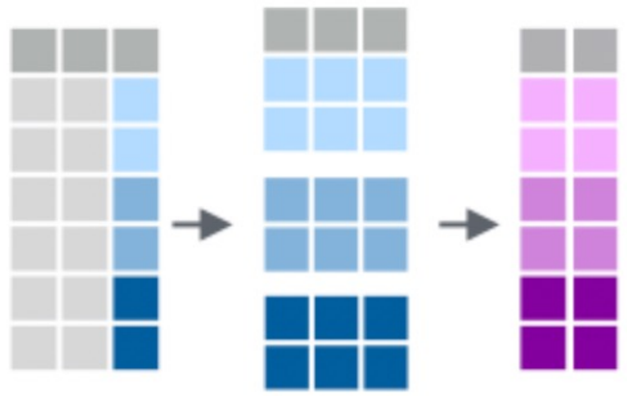


```
[[1]]
# A tibble: 2 x 10
    mpg  disp    hp  drat    wt  qsec    vs    am  gear  carb
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  22.8  108     93  3.85  2.32  18.6     1     1     4     1
2  24.4  147.    62  3.69  3.19  20       1     0     4     2

[[2]]
# A tibble: 2 x 10
    mpg  disp    hp  drat    wt  qsec    vs    am  gear  carb
  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
1  21    160    110  3.9   2.62  16.5     0     1     4     4
```

Mapping with and grouping

- We can use `group_by()` and apply different mapping functions to groups within our datasets
- Ex. `group_split() + map_dfr()`



```
mtcars %>%
    group_split(cyl) %>% # creates a grouped data frame
    map_dfr(head, n = 2)
```

```
## # A tibble: 6 x 11
##     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
## 2  24.4     4  147.    62  3.69  3.19  20       1     0     4     2
## 3  21       6  160    110  3.9   2.62  16.5     0     1     4     4
## 4  21       6  160    110  3.9   2.88  17.0     0     1     4     4
## 5  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2
## 6  14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
```

Practice

- Work with 1-2 other people on the functions_and_iteration lab