# Parallel Looping & Variants

Daniel Anderson

Week 5

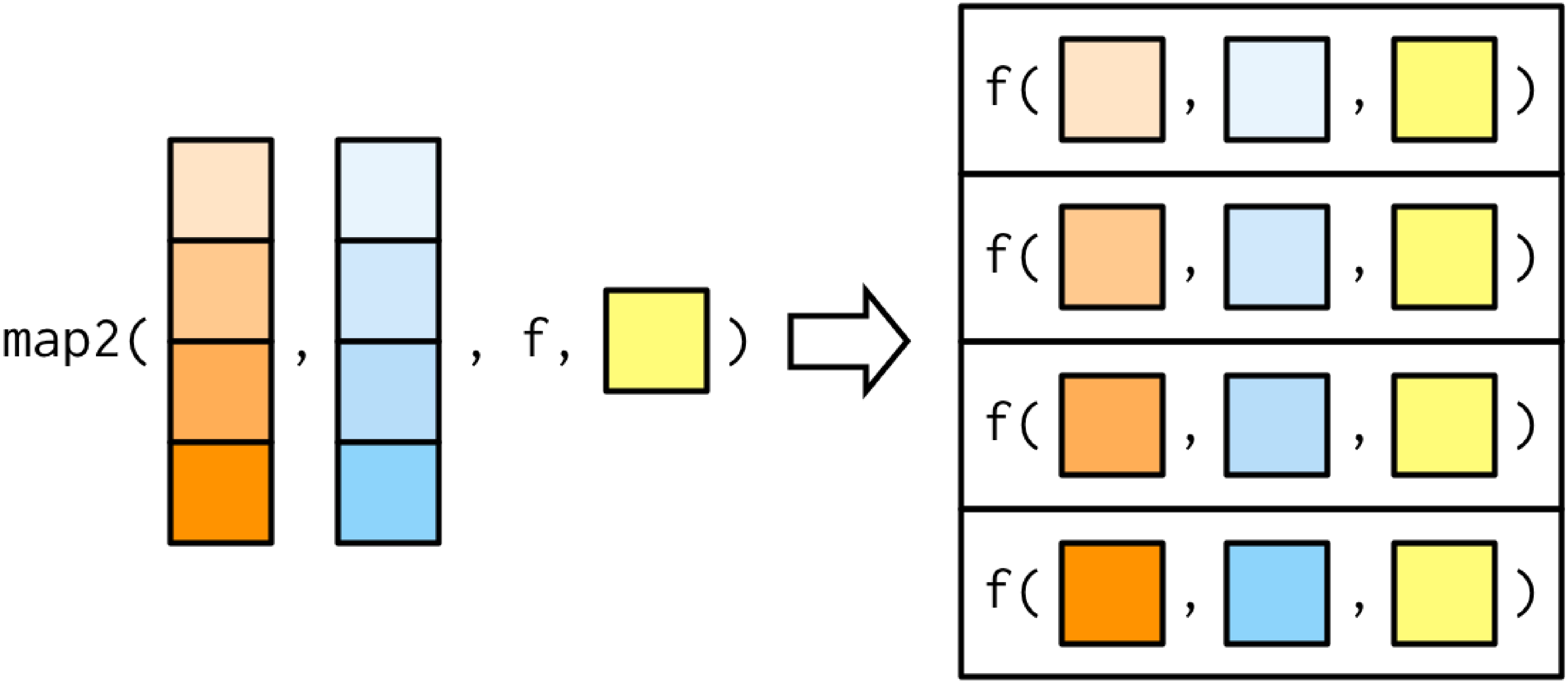
# Agenda

- Discuss `map2_*` and `pmap_*` (parallel iterations)
- `walk()` and friends
- `modify()`
- `safely()`
- `reduce()`

# Learning objectives

- Understand the differences between `map`, `map2`, and `pmap`
- Know when to apply `walk` instead of `map`, and why it may be useful
- Understand the similarities and differences between `map` and `modify`
- Diagnose errors with `safely` and understand other situations where it may be helpful
- Collapsing/reducing lists with `purrr::reduce()` or `base::Reduce()`

# map2



# A few Examples

Basic simulations – iterating over two vectors

Plots by month, changing the title

# Simulation

- Simulate data from a normal distribution
  - Vary $n$ from 5 to 150 by increments of 5
  - For each $n$, vary $\mu$ from $-2$ to 2 by increments of 0.25

## How do we get all combinations

`expand.grid`

# Example `expand.grid`

Bonus: It turns it into a data frame!

```r
ints <- 1:3
lets <- c("a", "b", "c")
expand.grid(ints, lets)
```

```
##   Var1 Var2
## 1    1    a
## 2    2    a
## 3    3    a
## 4    1    b
## 5    2    b
## 6    3    b
## 7    1    c
## 8    2    c
## 9    3    c
```

# Set conditions

Please follow along

```
conditions <- expand.grid(
  n = seq(5, 150, 5),
  mu = seq(-2, 2, 0.25)
)
```

```
head(conditions)
```

```
##    n mu
## 1  5 -2
## 2 10 -2
## 3 15 -2
## 4 20 -2
## 5 25 -2
## 6 30 -2
```

```
tail(conditions)
```

```
##       n mu
## 505 125  2
## 506 130  2
## 507 135  2
## 508 140  2
## 509 145  2
## 510 150  2
```

# Simulate!

```r
sim1 <- map2(conditions$n, conditions$mu, ~{
    rnorm(n = .x, mean = .y, sd = 10)
})
str(sim1)
```

```
## List of 510
##  $ : num [1:5] -2.451 -4.568 13.281 -0.655 0.511
##  $ : num [1:10] 8.96 13.47 4.49 -11.02 -4.59 ...
##  $ : num [1:15] -2.004 -9.092 10.106 0.518 1.188 ...
##  $ : num [1:20] -1.68 -1.15 -2.93 -10.55 12.22 ...
##  $ : num [1:25] -4.44 -17.06 2.5 3.4 -7.92 ...
##  $ : num [1:30] 2.843 -8.808 -1.62 -3.26 0.751 ...
##  $ : num [1:35] -21.81 -11.1 4.57 1.63 -2.46 ...
##  $ : num [1:40] 11.49 -2.028 -20.209 0.831 -6.537 ...
##  $ : num [1:45] -5.159 0.236 -6.797 -3.766 -6.161 ...
##  $ : num [1:50] 3.38 -19.13 -6.7 3.11 1.54 ...
##  $ : num [1:55] -11.77 -1.45 12.5 -4.6 -1.92 ...
##  $ : num [1:60] -5.998 0.561 -6.62 10.99 3.965 ...
##  $ : num [1:65] -20.38 -13.52 -3.15 -6.58 7.44 ...
##  $ : num [1:70] 19.76 1.95 10.57 -12.95 -11.4 ...
##  $ : num [1:75] -2.29 -10.73 -11.27 -3.03 -7.03 ...
##  $ : num [1:80] 4.57 -2.82 4.18 -1.33 -8.75 ...
##  $ : num [1:85] 11.05 15.3 4.34 -14.35 -11.56 ...
##  $ : num [1:90] -3.62 3.73 6.69 -15.89 -7.76 ...
##  $ : num [1:95] 0.867 -5.777 -18.431 -2.325 11.104 ...
##  $ : num [1:100] 4.98 6.23 -4.84 11.01 8.29 ...
##  $ : num [1:105] -10.27 -10.91 12.07 -6.96 7.68 ...
##  $ : num [1:110] -0.417 5.606 -8.071 -10.544 12.469 ...
##  $ : num [1:115] 4.027 -4.422 -8.751 -9.958 -0.423 ...
##  $ : num [1:120] 12.238 -0.105 14.976 -5.154 -18.831 ...
##  $ : num [1:125] -13.682 -0.952 -8.613 -10.508 -8.27 ...
```

# More powerful

## Add it as a list column!

```
sim2 <- conditions %>%
  as_tibble() %>% # Not required, but definitely helpful
  mutate(sim = map2(n, mu, ~rnorm(n = .x, mean = .y, sd = 10)))
sim2
```

```
## # A tibble: 510 × 3
##         n    mu sim
##     <dbl> <dbl> <list>
##  1      5    -2 <dbl [5]>
##  2     10    -2 <dbl [10]>
##  3     15    -2 <dbl [15]>
##  4     20    -2 <dbl [20]>
##  5     25    -2 <dbl [25]>
##  6     30    -2 <dbl [30]>
##  7     35    -2 <dbl [35]>
##  8     40    -2 <dbl [40]>
##  9     45    -2 <dbl [45]>
## 10     50    -2 <dbl [50]>
## # … with 500 more rows
```

# Unnest

```
conditions %>%
  as_tibble() %>%
  mutate(sim = map2(n, mu, ~rnorm(.x, .y, sd = 10))) %>%
  unnest(sim)
```

```
## # A tibble: 39,525 × 3
##          n    mu        sim
##      <dbl> <dbl>      <dbl>
##  1       5    -2   6.028304
##  2       5    -2  15.13266
##  3       5    -2   4.924202
##  4       5    -2   1.260976
##  5       5    -2   1.369952
##  6      10    -2  14.44016
##  7      10    -2   8.041027
##  8      10    -2 -24.28194
##  9      10    -2 -28.28235
## 10      10    -2  -4.804035
## # … with 39,515 more rows
```

# Challenge

Can you replicate what we just did, but using a `rowwise()` approach?

```r
conditions %>%
  rowwise() %>%
  mutate(sim = list(rnorm(n, mu, sd = 10))) %>%
  unnest(sim)
```

```
## # A tibble: 39,525 × 3
##          n    mu        sim
##      <dbl> <dbl>      <dbl>
##  1       5    -2 -18.14905
##  2       5    -2   4.356298
##  3       5    -2  -6.149922
##  4       5    -2  -8.977729
##  5       5    -2 -31.17708
##  6      10    -2 -18.16732
##  7      10    -2  -5.176330
##  8      10    -2 -13.09294
##  9      10    -2  12.00952
## 10      10    -2 -15.62448
## # … with 39,515 more rows
```

03:00

# Vary the sd too?

`pmap`

Which we'll get to soon

# Varying the title of a plot

# The data

Please follow along

```r
library(fivethirtyeight)
pulitzer
```

```
## # A tibble: 50 × 7
##    newspaper             circ2004 circ2013 pctchg_circ num_finals1990_2003
##    <chr>                    <dbl>    <dbl>       <int>               <int>
##  1 USA Today              2192098  1674306         -24                   1
##  2 Wall Street Journal    2101017  2378827          13                  30
##  3 New York Times         1119027  1865318          67                  55
##  4 Los Angeles Times       983727   653868         -34                  44
##  5 Washington Post         760034   474767         -38                  52
##  6 New York Daily News     712671   516165         -28                   4
##  7 New York Post           642844   500521         -22                   0
##  8 Chicago Tribune         603315   414930         -31                  23
##  9 San Jose Mercury News   558874   583998           4                   4
## 10 Newsday                 553117   377744         -32                  12
## # … with 40 more rows, and 2 more variables: num_finals2004_2014 <int>,
## #   num_finals1990_2014 <int>
```

# Prep data

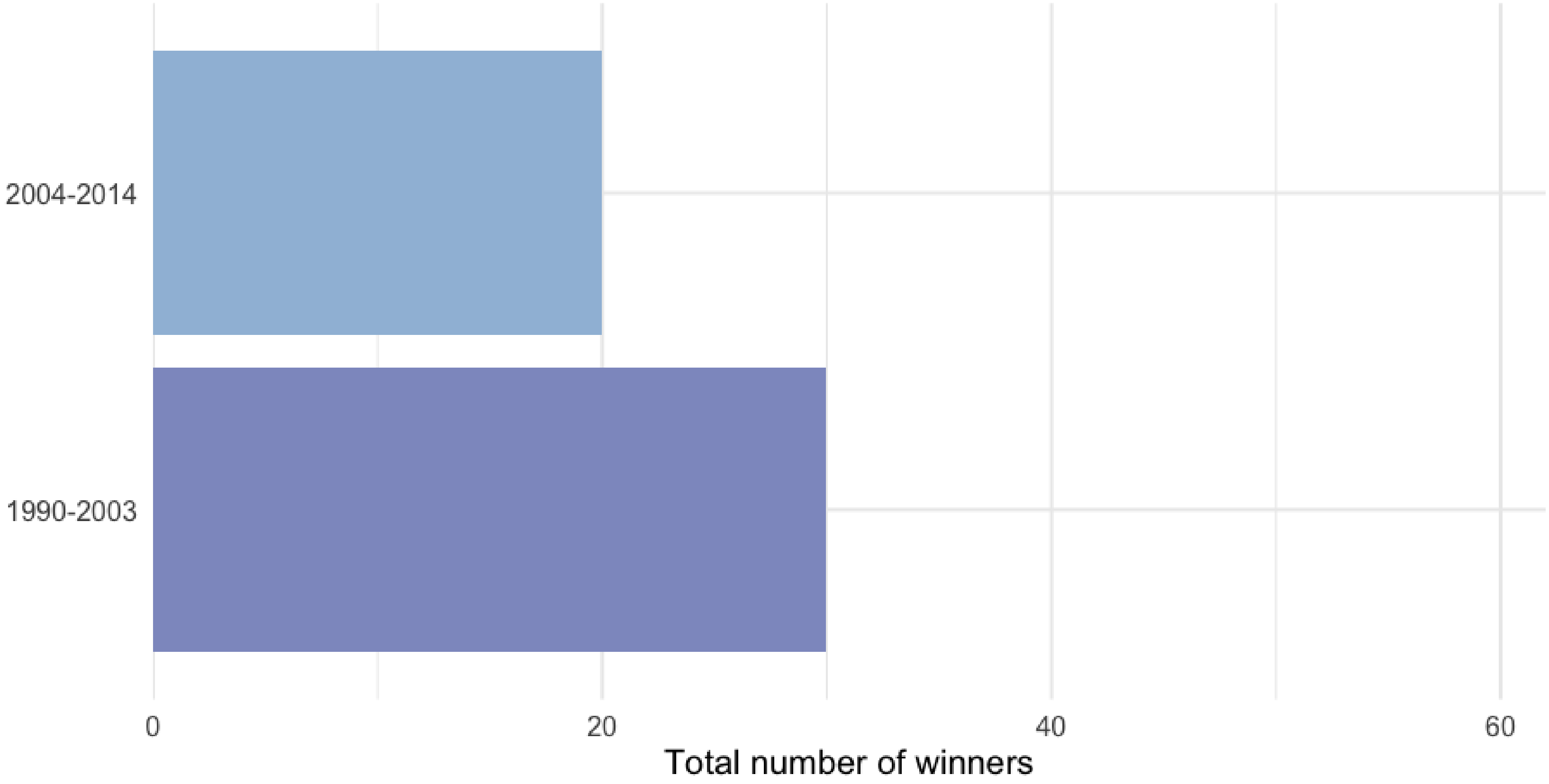
```
pulitzer<- pulitzer %>%
  select(newspaper, starts_with("num")) %>%
  pivot_longer(
    -newspaper,
    names_to = "year_range",
    values_to = "n",
    names_prefix = "num_finals"
  ) %>%
  mutate(year_range = str_replace_all(year_range, "_", "-")) %>%
  filter(year_range != "1990-2014")

head(pulitzer)
```

```
## # A tibble: 6 × 3
##   newspaper           year_range     n
##   <chr>               <chr>      <int>
## 1 USA Today           1990-2003      1
## 2 USA Today           2004-2014      1
## 3 Wall Street Journal 1990-2003     30
## 4 Wall Street Journal 2004-2014     20
## 5 New York Times      1990-2003     55
## 6 New York Times      2004-2014     62
```

# One plot

```r
wsj <- pulitzer %>%
    filter(newspaper == "Wall Street Journal")

ggplot(wsj, aes(n, year_range)) +
  geom_col(aes(fill = n)) +
  scale_fill_distiller(
    type = "seq",
    limits = c(0, max(pulitzer$n)),
    palette = "BuPu",
    direction = 1
  ) +
  scale_x_continuous(
    limits = c(0, max(pulitzer$n)),
    expand = c(0, 0)
  ) +
  guides(fill = "none") +
  labs(
    title = "Pulitzer Prize winners: Wall Street Journal",
    x = "Total number of winners",
    y = ""
  )
```

Pulitzer Prize winners: Wall Street Journal



# Nest data

```
by_newspaper <- pulitzer %>%
    group_by(newspaper) %>%
    nest()

by_newspaper
```

```
## # A tibble: 50 × 2
## # Groups:   newspaper [50]
##    newspaper             data
##    <chr>                 <list>
##  1 USA Today             <tibble [2 × 2]>
##  2 Wall Street Journal   <tibble [2 × 2]>
##  3 New York Times        <tibble [2 × 2]>
##  4 Los Angeles Times     <tibble [2 × 2]>
##  5 Washington Post       <tibble [2 × 2]>
##  6 New York Daily News   <tibble [2 × 2]>
##  7 New York Post         <tibble [2 × 2]>
##  8 Chicago Tribune       <tibble [2 × 2]>
##  9 San Jose Mercury News <tibble [2 × 2]>
## 10 Newsday               <tibble [2 × 2]>
## # … with 40 more rows
```

# Produce all plots

## You try first!

Don't worry about the correct title yet, if you don't want

03:00

```
by_newspaper %>%
    mutate(
      plot = map(
        data, ~{
          ggplot(aes(n, year_range)) +
            geom_col(aes(fill = n)) +
          scale_fill_distiller(
            type = "seq",
            limits = c(0, max(pulitzer$n)),
            palette = "BuPu",
            direction = 1
          ) +
          scale_x_continuous(
            limits = c(0, max(pulitzer$n)),
            expand = c(0, 0)
          ) +
        guides(fill = "none") +
        labs(
          title = "Pulitzer Prize winners",
          x = "Total number of winners",
          y = ""
        )
      }
    )
  )
```

# Add title

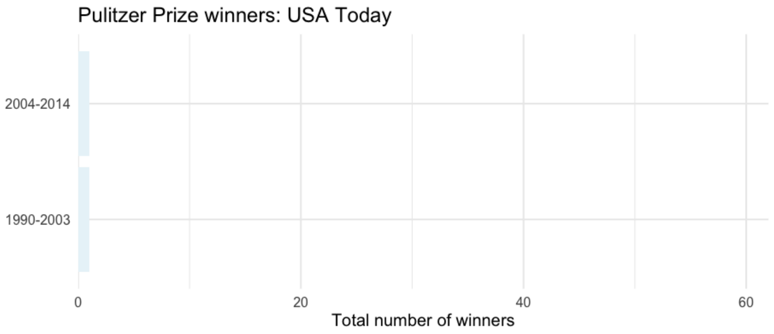
```r
library(glue)

p <- by_newspaper %>%
    mutate(
      plot = map2(
     data, newspaper, ~{
          ggplot(.x, aes(n, year_range)) +
            geom_col(aes(fill = n)) +
          scale_fill_distiller(
            type = "seq",
            limits = c(0, max(pulitzer$n)),
            palette = "BuPu",
            direction = 1
          ) +
          scale_x_continuous(
            limits = c(0, max(pulitzer$n)),
            expand = c(0, 0)
          ) +
          guides(fill = "none") +
          labs(
            title = glue("Pulitzer Prize winners: {.y}"),
            x = "Total number of winners",
            y = ""
          )
        }
        )
    )
```

```
p
```

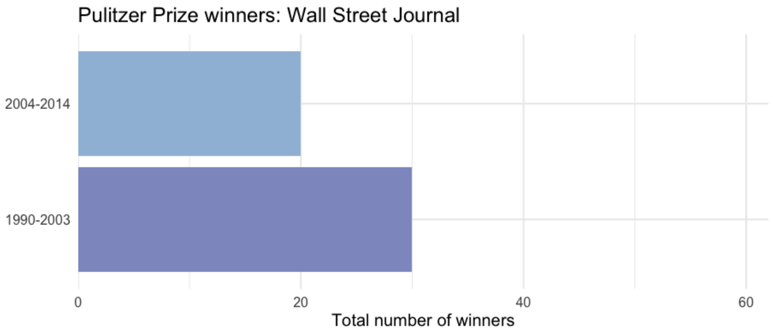
```
## # A tibble: 50 × 3
## # Groups:   newspaper [50]
##    newspaper              data             plot  
##    <chr>                  <list>           <list>
##  1 USA Today              <tibble [2 × 2]> <gg>  
##  2 Wall Street Journal    <tibble [2 × 2]> <gg>  
##  3 New York Times         <tibble [2 × 2]> <gg>  
##  4 Los Angeles Times      <tibble [2 × 2]> <gg>  
##  5 Washington Post        <tibble [2 × 2]> <gg>  
##  6 New York Daily News    <tibble [2 × 2]> <gg>  
##  7 New York Post          <tibble [2 × 2]> <gg>  
##  8 Chicago Tribune        <tibble [2 × 2]> <gg>  
##  9 San Jose Mercury News <tibble [2 × 2]> <gg>  
## 10 Newsday                <tibble [2 × 2]> <gg>  
## # … with 40 more rows
```

# Look at a couple plots

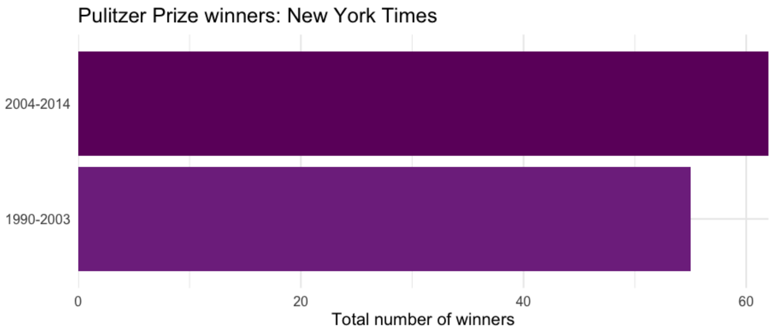
```
p$plot[[1]]
```



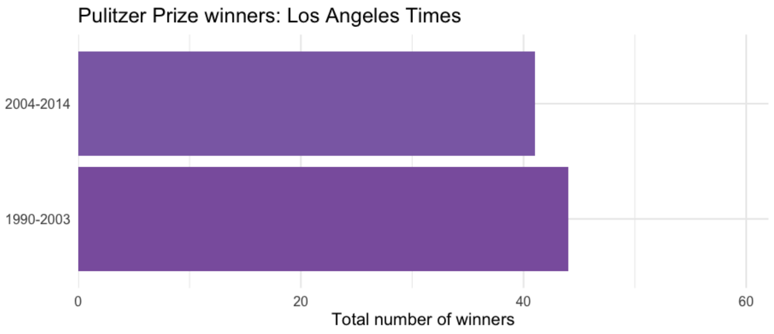
```
p$plot[[2]]
```



```
p$plot[[3]]
```



```
p$plot[[4]]
```



# Challenge

(You can probably guess where this is going)

Can you reproduce the prior plots using a `rowwise()` approach?

03:00

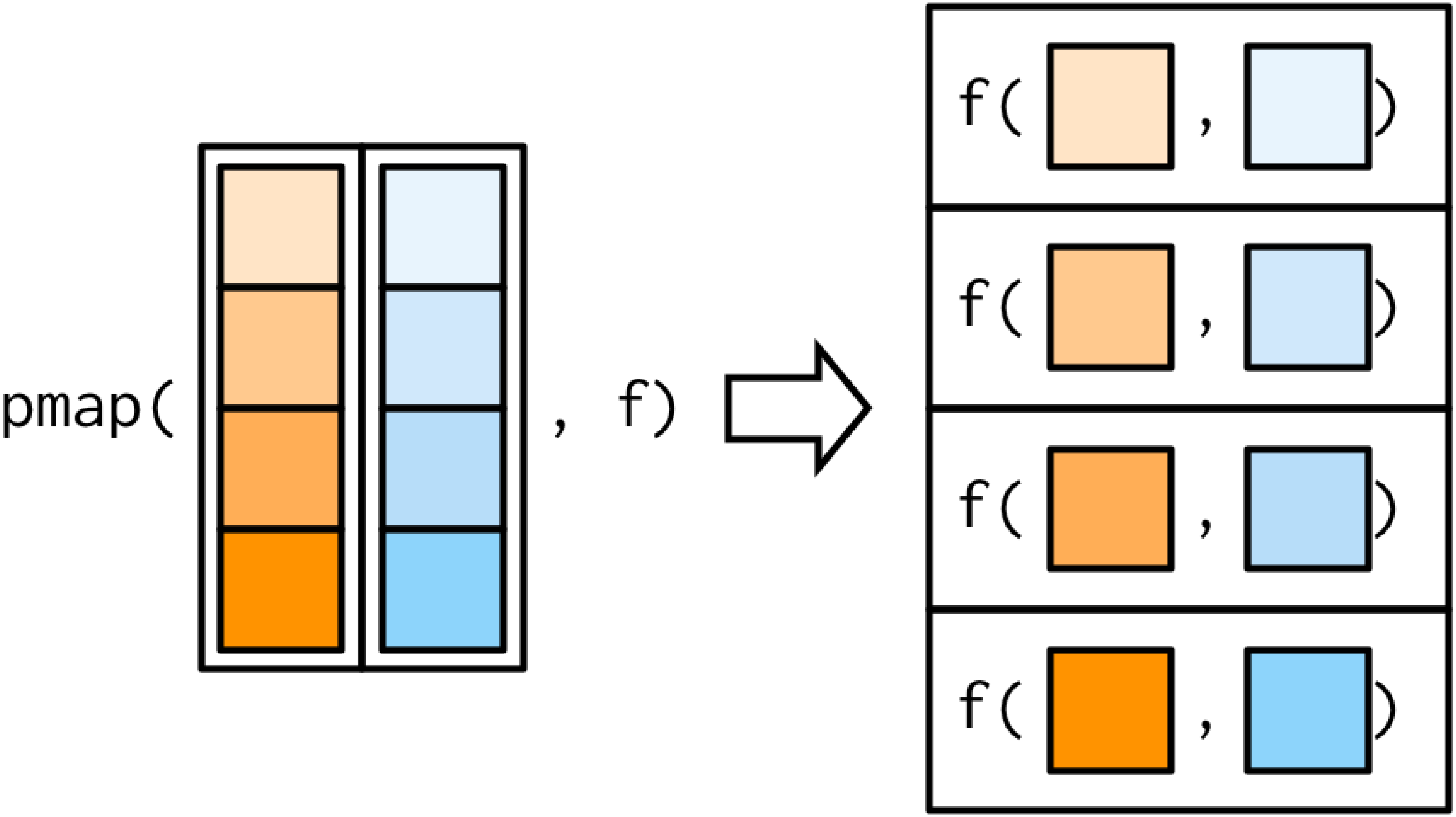
```
pulitzer %>%
nest_by(newspaper) %>%
    mutate(
      plot = list(
     ggplot(data, aes(n, year_range)) +
        geom_col(aes(fill = n)) +
        scale_fill_distiller(
          type = "seq",
          limits = c(0, max(pulitzer$n)),
          palette = "BuPu",
          direction = 1
        ) +
        scale_x_continuous(
          limits = c(0, max(pulitzer$n)),
          expand = c(0, 0)
        ) +
        guides(fill = "none") +
        labs(
          title = glue("Pulitzer Prize winners: {newspaper}"),
          x = "Total number of winners",
          y = ""
        )
    )
  )
```

```
## # A tibble: 50 × 3
## # Rowwise:  newspaper
##    newspaper                                  data plot
##    <chr>                        <list<tibble[,2]>> <list>
##   1 Arizona Republic                        [2 × 2] <gg>
##   2 Atlanta Journal Constitution            [2 × 2] <gg>
##   3 Baltimore Sun                           [2 × 2] <gg>
##   4 Boston Globe                            [2 × 2] <gg>
```

# Iterating over $n$ vectors

pmap

# pmap



# Simulation

- Simulate data from a normal distribution
  - Vary $n$ from 5 to 150 by increments of 5
  - For each $n$, vary $\mu$ from –2 to 2 by increments of 0.25
  - For each $\sigma$ from 1 to 3 by increments of 0.1

```r
full_conditions <- expand.grid(
  n = seq(5, 150, 5),
  mu = seq(-2, 2, 0.25),
  sd = seq(1, 3, .1)
)
```

```r
head(full_conditions)
```

```
##    n mu sd
## 1  5 -2  1
## 2 10 -2  1
## 3 15 -2  1
## 4 20 -2  1
## 5 25 -2  1
## 6 30 -2  1
```

```r
tail(full_conditions)
```

```
##         n mu sd
## 10705 125  2  3
## 10706 130  2  3
## 10707 135  2  3
## 10708 140  2  3
## 10709 145  2  3
## 10710 150  2  3
```

# Full Simulation

```
fsim <- pmap(
  list(
    number = full_conditions$n,
    average = full_conditions$mu,
    stdev = full_conditions$sd
  ),
  function(number, average, stdev) {
    rnorm(n = number, mean = average, sd = stdev)
  }
)
str(fsim)
```

```
## List of 10710
##  $ : num [1:5] -4.199 -3.204 -2.763 -0.905 -1.841
##  $ : num [1:10] -2.41 -1.7 -3.88 -1.14 -1.27 ...
##  $ : num [1:15] -3.31 -2.45 -2.27 -2.09 -1.73 ...
##  $ : num [1:20] -2.52 -1.39 -3.14 -2.084 -0.328 ...
##  $ : num [1:25] -1.63 -2.88 -2.38 -2.82 -2.91 ...
##  $ : num [1:30] -2.41 -1.81 -2.03 -2.93 -2.28 ...
##  $ : num [1:35] -2.189 -1.206 -2.042 -2.412 0.821 ...
##  $ : num [1:40] -3.43 -1.02 -1.97 -2.94 -2.09 ...
##  $ : num [1:45] -2.79 -2.46 -1.42 -2.8 -3.16 ...
##  $ : num [1:50] -0.151 -1.627 -1.442 -2.382 -3.784 ...
##  $ : num [1:55] -1.5568 -2.8275 -0.8929 -1.3075 -0.0722 ...
##  $ : num [1:60] -0.42 -2.62 -2.3 -2.09 -2.19 ...
##  $ : num [1:65] 0.0461 -3.6882 -1.7017 -0.6819 -1.7574 ...
##  $ : num [1:70] -2.6 -2.04 -3.66 -1.3 -1.98 ...
##  $ : num [1:75] -2.88 -2.57 -0.41 -2.33 -1.35 ...
##  $ : num [1:80] -2.3 -3.94 -3.27 -2.4 -1.52 ...
##  $ : num [1:85] -1.32 -2.07 -1.66 -2.06 -2.15 ...
```

# Alternative spec

```
fsim <- pmap(
  list(
    full_conditions$n,
    full_conditions$mu,
    full_conditions$sd
  ),
  ~rnorm(n = ..1, mean = ..2, sd = ..3)
)
str(fsim)
```

```
## List of 10710
##  $ : num [1:5] -2.87 -1.68 -2.32 -1.66 -1.08
##  $ : num [1:10] -3.8 -2.77 -1.54 -2.09 -1.45 ...
##  $ : num [1:15] -2.06 -2.94 -2.49 -1.95 -1.7 ...
##  $ : num [1:20] -2.252 0.982 -2.533 -1.488 -2.107 ...
##  $ : num [1:25] -1.881 -4.009 -2.054 -0.562 -4.825 ...
##  $ : num [1:30] -2.65 -3.07 -1.71 -1.81 -3.31 ...
##  $ : num [1:35] -2.7959 -0.6448 -1.6797 -1.7245 0.0426 ...
##  $ : num [1:40] -2.06 -1.51 -1.55 -2.01 -2.7 ...
##  $ : num [1:45] -2.0684 -0.0702 -3.0578 -3.8359 -2.5373 ...
##  $ : num [1:50] -1.84 -1.97 -2.24 -2.48 -2.15 ...
##  $ : num [1:55] -3.23 -2.85 -1.79 -3.6 -4.26 ...
##  $ : num [1:60] -1.312 -4.121 0.292 -1.261 -1.614 ...
##  $ : num [1:65] -0.739 -0.758 -2.503 -2.336 -2.551 ...
##  $ : num [1:70] -2.13 -1.82 -1.88 -1.73 -2.97 ...
##  $ : num [1:75] -1.12 -1.34 -1.3 -2.1 -1.83 ...
##  $ : num [1:80] -2.472 -0.824 -3.924 -2.104 -0.749 ...
##  $ : num [1:85] -2.192 -1.415 -1.274 -2.019 -0.942 ...
##  $ : num [1:90] -1.507 -2.312 -0.323 -0.888 -0.207 ...
##  $ : num [1:95] -1.05 -3.42 -3.19 -1.94 -1.66 ...
```

# Simpler

## Maybe a little too clever

- A data frame is a list so...

```
fsim <- pmap(
  full_conditions,
  ~rnorm(n = ..1, mean = ..2, sd = ..3)
)
str(fsim)
```

```
## List of 10710
##  $ : num [1:5] -1.08 -2.65 -1.5 -1.99 -3.24
##  $ : num [1:10] -2.48 -1 -2.7 -2.2 -1.22 ...
##  $ : num [1:15] -1.039 -0.234 -2.134 0.528 -2.764 ...
##  $ : num [1:20] -0.563 -1.456 -2.317 -1.858 -1.123 ...
##  $ : num [1:25] -2.32 -1.09 -3.2 -1.5 -3.91 ...
##  $ : num [1:30] -1.31 -2.11 0.63 -1.98 -1.09 ...
##  $ : num [1:35] -2.77 -3.31 -2.72 -3.66 -2.94 ...
##  $ : num [1:40] -4.33 -2.78 -1.53 -1.95 -2.29 ...
##  $ : num [1:45] -3.124 -2.491 0.411 -1.956 -0.184 ...
##  $ : num [1:50] 0.478 -1.339 -1.519 -2.596 -1.915 ...
##  $ : num [1:55] -2.884 -4.148 -0.759 -2.151 -1.535 ...
##  $ : num [1:60] -1.373 -1.524 -3.11 -2.065 -0.995 ...
##  $ : num [1:65] -2.191 -0.475 -3.031 -1.903 -0.812 ...
##  $ : num [1:70] -1.17 -2.67 -3.83 -3.32 -2.83 ...
##  $ : num [1:75] -2.345 -0.782 -2.943 -2.79 -2.927 ...
##  $ : num [1:80] -1.52 -1.78 -2.61 -2.92 -2.93 ...
##  $ : num [1:85] -2.35 -3 -2.85 -3.3 -2.21 ...
##  $ : num [1:90] -2.59 -2.46 -2.52 -3.01 -2.96 ...
```

# List column version

```r
full_conditions %>%
    as_tibble() %>%
    mutate(sim = pmap(list(n, mu, sd), ~rnorm(..1, ..2, ..3)))
```

```
## # A tibble: 10,710 × 4
##        n    mu    sd sim
##    <dbl> <dbl> <dbl> <list>
##  1     5    -2     1 <dbl [5]>
##  2    10    -2     1 <dbl [10]>
##  3    15    -2     1 <dbl [15]>
##  4    20    -2     1 <dbl [20]>
##  5    25    -2     1 <dbl [25]>
##  6    30    -2     1 <dbl [30]>
##  7    35    -2     1 <dbl [35]>
##  8    40    -2     1 <dbl [40]>
##  9    45    -2     1 <dbl [45]>
## 10    50    -2     1 <dbl [50]>
## # … with 10,700 more rows
```

# Unnest

```
full_conditions %>%
    as_tibble() %>%
    mutate(sim = pmap(
        list(n, mu, sd), ~rnorm(..1, ..2, ..3)
      )
  ) %>%
    unnest(sim)
```

```
## # A tibble: 830,025 × 4
##        n    mu    sd       sim
##    <dbl> <dbl> <dbl>     <dbl>
##  1     5    -2     1 -1.014073
##  2     5    -2     1 -2.302678
##  3     5    -2     1 -1.358109
##  4     5    -2     1 -3.393003
##  5     5    -2     1 -2.740573
##  6    10    -2     1 -2.406099
##  7    10    -2     1 -1.734234
##  8    10    -2     1 -1.031400
##  9    10    -2     1 -2.074230
## 10    10    -2     1 -3.276515
## # … with 830,015 more rows
```

# Replicate with nest_by()

## You try first

```
full_conditions %>%
  rowwise() %>%
  mutate(sim = list(rnorm(n, mu, sd))) %>%
  unnest(sim)
```

```
## # A tibble: 830,025 × 4
##          n     mu     sd        sim
##      <dbl> <dbl> <dbl>      <dbl>
##  1       5     -2      1 -1.732164
##  2       5     -2      1 -1.446031
##  3       5     -2      1 -2.720625
##  4       5     -2      1 -2.883673
##  5       5     -2      1 -1.490718
##  6      10     -2      1 -1.546600
##  7      10     -2      1 -1.520583
##  8      10     -2      1 -4.065966
##  9      10     -2      1 -0.5051152
## 10      10     -2      1 -1.517951
## # … with 830,015 more rows
```

03:00

# Plot

Add a caption stating the total number of Pulitzer prize winners across years

# Add column for total

```
pulitzer <- pulitzer %>%
    group_by(newspaper) %>%
    mutate(tot = sum(n))
pulitzer
```

```
## # A tibble: 100 × 4
## # Groups:   newspaper [50]
##    newspaper           year_range     n   tot
##    <chr>               <chr>      <int> <int>
##  1 USA Today           1990-2003      1     2
##  2 USA Today           2004-2014      1     2
##  3 Wall Street Journal 1990-2003     30    50
##  4 Wall Street Journal 2004-2014     20    50
##  5 New York Times      1990-2003     55   117
##  6 New York Times      2004-2014     62   117
##  7 Los Angeles Times   1990-2003     44    85
##  8 Los Angeles Times   2004-2014     41    85
##  9 Washington Post     1990-2003     52   100
## 10 Washington Post     2004-2014     48   100
## # … with 90 more rows
```

# Easiest way (imo)

Create a column to represent exactly the label you want.

```r
#install.packages("english")
library(english)
pulitzer <- pulitzer %>%
    mutate(
      label = glue(
        "{str_to_title(as.english(tot))} Total Pulitzer Awards"
    )
)
```

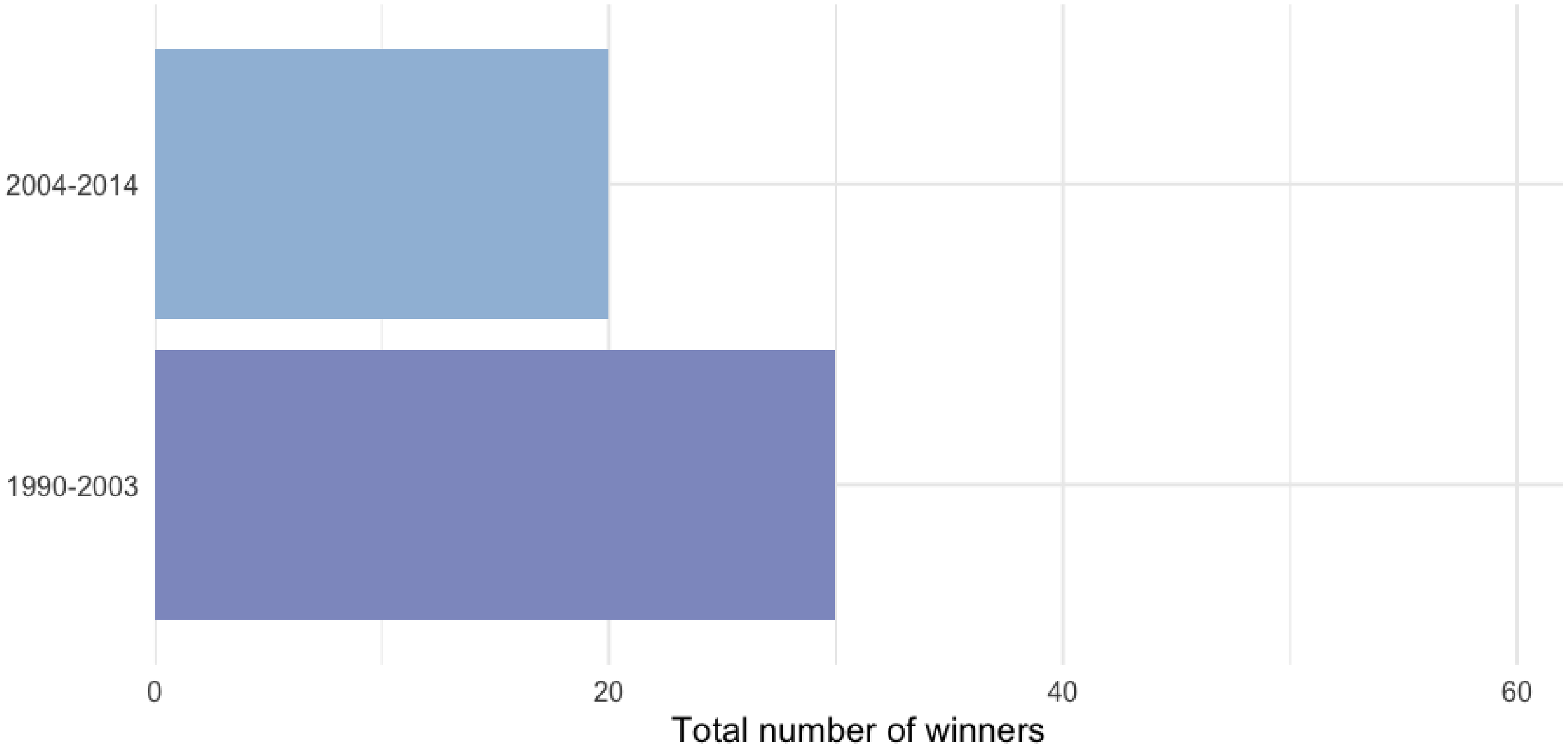
```
select(pulitzer, newspaper, label)
```

```
## # A tibble: 100 × 2
## # Groups:   newspaper [50]
##    newspaper           label
##    <chr>               <glue>
##  1 USA Today           Two Total Pulitzer Awards
##  2 USA Today           Two Total Pulitzer Awards
##  3 Wall Street Journal Fifty Total Pulitzer Awards
##  4 Wall Street Journal Fifty Total Pulitzer Awards
##  5 New York Times      One Hundred Seventeen Total Pulitzer Awards
##  6 New York Times      One Hundred Seventeen Total Pulitzer Awards
##  7 Los Angeles Times   Eighty-Five Total Pulitzer Awards
##  8 Los Angeles Times   Eighty-Five Total Pulitzer Awards
##  9 Washington Post     One Hundred Total Pulitzer Awards
## 10 Washington Post     One Hundred Total Pulitzer Awards
## # … with 90 more rows
```

# Produce one plot

```r
wsj2 <- pulitzer %>%
    filter(newspaper == "Wall Street Journal")

ggplot(wsj2, aes(n, year_range)) +
  geom_col(aes(fill = n)) +
  scale_fill_distiller(
    type = "seq",
    limits = c(0, max(pulitzer$n)),
    palette = "BuPu",
    direction = 1
  ) +
  scale_x_continuous(
    limits = c(0, max(pulitzer$n)),
    expand = c(0, 0)
  ) +
  guides(fill = "none") +
  labs(
    title = glue("Pulitzer Prize winners: Wall Street Journal"),
    x = "Total number of winners",
    y = "",
    caption = unique(wsj2$label)
  )
```

## Pulitzer Prize winners: Wall Street Journal



Fifty Total Pulitzer Awards

# Produce all plots

## Nest first

```
by_newspaper_label <- pulitzer %>%
    group_by(newspaper, label) %>%
    nest()

by_newspaper_label
```

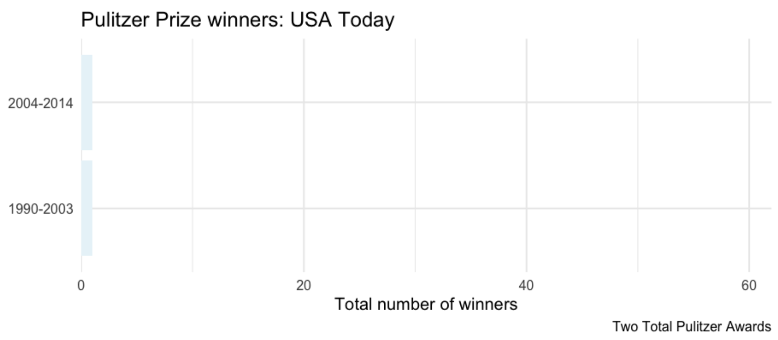
```
## # A tibble: 50 × 3
## # Groups:   newspaper, label [50]
##    newspaper             label                                       data
##    <chr>                 <glue>                                      <list>
##  1 USA Today             Two Total Pulitzer Awards                   <tibble [2 × 3]>
##  2 Wall Street Journal   Fifty Total Pulitzer Awards                 <tibble [2 × 3]>
##  3 New York Times        One Hundred Seventeen Total Pulitzer Awards <tibble [2 × 3]>
##  4 Los Angeles Times     Eighty-Five Total Pulitzer Awards           <tibble [2 × 3]>
##  5 Washington Post       One Hundred Total Pulitzer Awards           <tibble [2 × 3]>
##  6 New York Daily News   Six Total Pulitzer Awards                   <tibble [2 × 3]>
##  7 New York Post         Zero Total Pulitzer Awards                  <tibble [2 × 3]>
##  8 Chicago Tribune       Thirty-Eight Total Pulitzer Awards          <tibble [2 × 3]>
##  9 San Jose Mercury News Six Total Pulitzer Awards                   <tibble [2 × 3]>
## 10 Newsday               Eighteen Total Pulitzer Awards              <tibble [2 × 3]>
## # … with 40 more rows
```

# Produce plots

```r
final_plots <- by_newspaper_label %>%
    mutate(plots = pmap(list(newspaper, label, data), ~{
    ggplot(..3, aes(n, year_range)) +
      geom_col(aes(fill = n)) +
      scale_fill_distiller(
        type = "seq",
        limits = c(0, max(pulitzer$n)),
        palette = "BuPu",
        direction = 1
      ) +
        scale_x_continuous(
          limits = c(0, max(pulitzer$n)),
          expand = c(0, 0)
      ) +
        guides(fill = "none") +
        labs(
       title = glue("Pulitzer Prize winners: {..1}"),
          x = "Total number of winners",
          y = "",
       caption = ..2
        )
      })
)
```

# Look at a couple plots

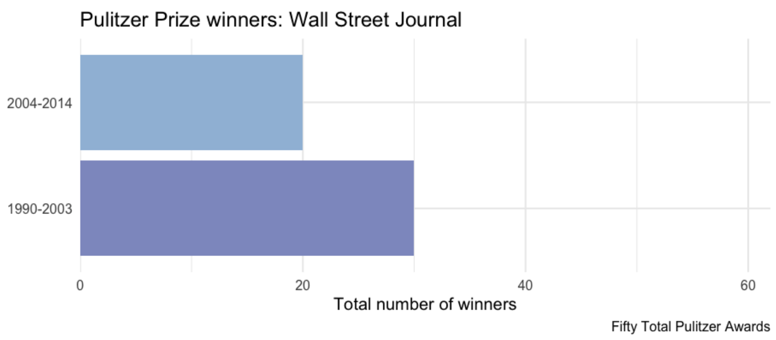
```
final_plots$plots[[1]]
```



Pulitzer Prize winners: USA Today

Two Total Pulitzer Awards

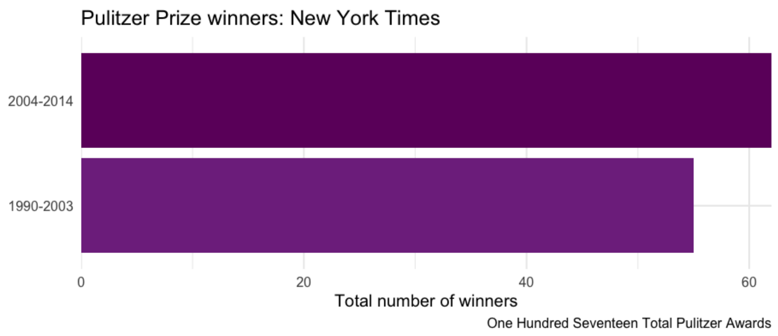
```
final_plots$plots[[2]]
```



Pulitzer Prize winners: Wall Street Journal

Fifty Total Pulitzer Awards

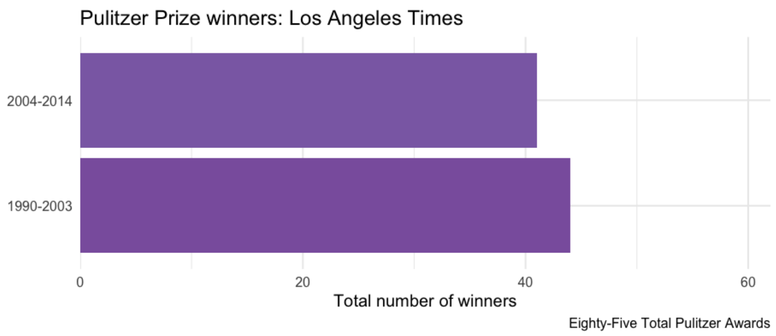
```
final_plots$plots[[3]]
```



Pulitzer Prize winners: New York Times

One Hundred Seventeen Total Pulitzer Awards

```
final_plots$plots[[4]]
```



Pulitzer Prize winners: Los Angeles Times

Eighty-Five Total Pulitzer Awards

# Replicate with `nest_by()`

You try first

03:00

```r
final_plots2 <- pulitzer %>%
  ungroup() %>%
  nest_by(newspaper, label) %>%
    mutate(
      plots = list(
        ggplot(data, aes(n, year_range)) +
          geom_col(aes(fill = n)) +
        scale_fill_distiller(
          type = "seq",
          limits = c(0, max(pulitzer$n)),
          palette = "BuPu",
          direction = 1
        ) +
          scale_x_continuous(
            limits = c(0, max(pulitzer$n)),
            expand = c(0, 0)
        ) +
          guides(fill = "none") +
          labs(
            title = glue("Pulitzer Prize winners: {newspaper}"),
            x = "Total number of winners",
            y = "",
            caption = label
        )
      )
)
```

```r
final_plots2
```

```
## # A tibble: 50 × 4
## # Rowwise:  newspaper, label
##    newspaper                    label                                        data plot
##    <chr>                        <glue>                             <list<tibble[> <lis
##  1 Arizona Republic             Seven Total Pulitzer Awards               [2 × 3] <gg>
##  2 Atlanta Journal Constitution Six Total Pulitzer Awards                 [2 × 3] <gg>
##  3 Baltimore Sun                Thirteen Total Pulitzer Awards            [2 × 3] <gg>
##  4 Boston Globe                 Forty-One Total Pulitzer Awards           [2 × 3] <gg>
##  5 Boston Herald                Zero Total Pulitzer Awards                [2 × 3] <gg>
##  6 Charlotte Observer           Four Total Pulitzer Awards                [2 × 3] <gg>
##  7 Chicago Sun-Times            Two Total Pulitzer Awards                 [2 × 3] <gg>
##  8 Chicago Tribune              Thirty-Eight Total Pulitzer Awards        [2 × 3] <gg>
##  9 Cleveland Plain Dealer       Eleven Total Pulitzer Awards              [2 × 3] <gg>
## 10 Columbus Dispatch            One Total Pulitzer Awards                 [2 × 3] <gg>
## # … with 40 more rows
```

# Save all plots

We'll have to iterate across at least two things: (a) file path/names, and (b) the plots themselves

We can do this with the `map()` family, but instead we'll use a different function, which we'll talk about more momentarily.

As an aside, what are the **steps** we would need to take to do this?

Could we use a `nest_by()` solution?

# Try with `nest_by()`

You try first:

- Create a vector of file paths
- "loop" through the file paths and the plots to save them

04:00

# Example

## Create a directory

```
fs::dir_create(here::here("plots", "pulitzers"))
```

## Create file paths

```
files <- str_replace_all(
  tolower(final_plots$newspaper),
  " ",
  "_"
)
paths <- here::here("plots", "pulitzers", glue("{files}.png"))
paths
```

```
##  [1] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [2] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [3] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [4] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [5] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [6] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [7] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [8] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
##  [9] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
## [10] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
## [11] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
## [12] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
## [13] "/Users/daniel/Teaching/data_sci_specialization/2021-22/c3-fp-2022/plots/pulitze
```

# Add paths to data frame

```r
final_plots %>%
  ungroup() %>%
  mutate(path = paths) %>%
  select(plots, path)
```

```
## # A tibble: 50 × 2
##    plots
##    <list>
##  1 <gg>
##  2 <gg>
##  3 <gg>
##  4 <gg>
##  5 <gg>
##  6 <gg>
##  7 <gg>
##  8 <gg>
##  9 <gg>
## 10 <gg>
## # … with 40 more rows, and 1 more variable: path <chr>
```

# Save

```r
final_plots %>%
  ungroup() %>%
  mutate(path = paths) %>%
  rowwise() %>%
  summarize(
    ggsave(
      path,
      plots,
      width = 9.5,
      height = 6.5,
      dpi = 500
    )
  )
```

```
## # A tibble: 50 × 1
## # … with 40 more rows, and 1 more variable:
## #   `ggsave(path, plots, width = 9.5, height = 6.5, dpi = 500)` <chr>
```

# Wrap–up

- Parallel iterations greatly increase the things you can do – iterating through at least two things simultaneously is pretty common

- The `nest_by()` approach can regularly get you the same result as `group_by() %>% nest() %>% mutate() %>% map()`

  - Caveat – must be in a data frame, which means working with list columns
  - My view – it's still worth learning both. Looping with **{purrr}** is super flexible and often safer than base versions (type safe). Doesn't have to be used within a data frame.

# Break

# Looping variants

# Agenda

- `walk()` and friends
- `modify()`
- `safely()`
- `reduce()`

# Reminder

## Learning Objectives (for this part)

- Know when to apply `walk` instead of `map`, and why it may be useful
- Understand the parallels and differences between `map` and `modify`
- Diagnose errors with `safely` and understand other situations where it may be helpful
- Collapsing/reducing lists with `purrr::reduce()` or `base::Reduce()`

# Setup

Let's go back to our plotting example:

## Saving

- We saw last time that we could use `nest_by()`
  - Required a bit of awkwardness with adding the paths to the data frame
  - Instead, we'll do it again but with the `walk()` family

# Why `walk()`?

> Walk is an alternative to map that you use when you want to call a function for its side effects, rather than for its return value. You typically do this because you want to render output to the screen or save files to disk – the important thing is the action, not the return value.

–r4ds

# More practical

If you use `walk()`, nothing will get printed to the screen. This is particularly helpful for RMarkdown files.

# Example

Please do the following

- Create a new RMarkdown document
- Paste the code you have for creating the plots in a code chunk there (along with the library loading, data cleaning, etc.)

03:00

# Create a directory

We already did this, but in case we hadn't...

```
fs::dir_create(here::here("plots", "pulitzers"))
```

## Create file paths

```
newspapers <- str_replace_all(
  tolower(final_plots$newspaper),
  " ",
  "_"
)
paths <- here::here(
  "plots",
  "pulitzers",
  glue("{newspapers}.png")
)
```

# Challenge

- Use a `map()` family function to loop through `paths` and `final_plots$plots` to save all plots.
- Render (knit) your file. What do you notice?

03:00

# `walk()`

Just like `map()`, we have parallel variants of `walk()`, including, `walk2()`, and `pwalk()`

These work just like `map()` but don't print to the screen

Try replacing your prior code with a `walk()` version.

How does the rendered output change?

02:00

# Save plots

```
walk2(paths, final_plots$plots, ggsave,
      width = 9.5,
      height = 6.5,
      dpi = 500)
```

# modify

> Unlike `map()` and its variants which always return a fixed object type (list for `map()`, integer vector for `map_int()`, etc), the `modify()` family always returns the same type as the input object.

# `map` vs `modify`

## map

```
map(mtcars, ~as.numeric(scale(.x)))
```

```
## $mpg
##  [1]  0.15088482  0.15088482  0.44954345  0.21725341 -0.23073453 -0.33028740
##  [7] -0.96078893  0.71501778  0.44954345 -0.14777380 -0.38006384 -0.61235388
## [13] -0.46302456 -0.81145962 -1.60788262 -1.60788262 -0.89442035  2.04238943
## [19]  1.71054652  2.29127162  0.23384555 -0.76168319 -0.81145962 -1.12671039
## [25] -0.14777380  1.19619000  0.98049211  1.71054652 -0.71190675 -0.06481307
## [31] -0.84464392  0.21725341
## 
## $cyl
##  [1] -0.1049878 -0.1049878 -1.2248578 -0.1049878  1.0148821 -0.1049878  1.0148821
##  [8] -1.2248578 -1.2248578 -0.1049878 -0.1049878  1.0148821  1.0148821  1.0148821
## [15]  1.0148821  1.0148821  1.0148821 -1.2248578 -1.2248578 -1.2248578 -1.2248578
## [22]  1.0148821  1.0148821  1.0148821  1.0148821 -1.2248578 -1.2248578 -1.2248578
## [29]  1.0148821 -0.1049878  1.0148821 -1.2248578
## 
## $disp
##  [1] -0.57061982 -0.57061982 -0.99018209  0.22009369  1.04308123 -0.04616698
##  [7]  1.04308123 -0.67793094 -0.72553512 -0.50929918 -0.50929918  0.36371309
## [13]  0.36371309  0.36371309  1.94675381  1.84993175  1.68856165 -1.22658929
## [19] -1.25079481 -1.28790993 -0.89255318  0.70420401  0.59124494  0.96239618
## [25]  1.36582144 -1.22416874 -0.89093948 -1.09426581  0.97046468 -0.69164740
## [31]  0.56703942 -0.88529152
## 
## $hp
##  [1] -0.53509284 -0.53509284 -0.78304046 -0.53509284  0.41294217 -0.60801861
##  [7]  1.43390296 -1.23518023 -0.75387015 -0.34548584 -0.34548584  0.48586794
```

# modify

```
modify(mtcars, ~as.numeric(scale(.x)))
```

```
##                             mpg        cyl        disp          hp        drat
## Mazda RX4            0.15088482 -0.1049878 -0.57061982 -0.53509284  0.56751369
## Mazda RX4 Wag        0.15088482 -0.1049878 -0.57061982 -0.53509284  0.56751369
## Datsun 710           0.44954345 -1.2248578 -0.99018209 -0.78304046  0.47399959
## Hornet 4 Drive       0.21725341 -0.1049878  0.22009369 -0.53509284 -0.96611753
## Hornet Sportabout   -0.23073453  1.0148821  1.04308123  0.41294217 -0.83519779
## Valiant             -0.33028740 -0.1049878 -0.04616698 -0.60801861 -1.56460776
## Duster 360          -0.96078893  1.0148821  1.04308123  1.43390296 -0.72298087
## Merc 240D            0.71501778 -1.2248578 -0.67793094 -1.23518023  0.17475447
## Merc 230             0.44954345 -1.2248578 -0.72553512 -0.75387015  0.60491932
## Merc 280            -0.14777380 -0.1049878 -0.50929918 -0.34548584  0.60491932
## Merc 280C           -0.38006384 -0.1049878 -0.50929918 -0.34548584  0.60491932
## Merc 450SE          -0.61235388  1.0148821  0.36371309  0.48586794 -0.98482035
## Merc 450SL          -0.46302456  1.0148821  0.36371309  0.48586794 -0.98482035
## Merc 450SLC         -0.81145962  1.0148821  0.36371309  0.48586794 -0.98482035
## Cadillac Fleetwood  -1.60788262  1.0148821  1.94675381  0.85049680 -1.24665983
## Lincoln Continental -1.60788262  1.0148821  1.84993175  0.99634834 -1.11574009
## Chrysler Imperial   -0.89442035  1.0148821  1.68856165  1.21512565 -0.68557523
## Fiat 128             2.04238943 -1.2248578 -1.22658929 -1.17683962  0.90416444
## Honda Civic          1.71054652 -1.2248578 -1.25079481 -1.38103178  2.49390411
## Toyota Corolla       2.29127162 -1.2248578 -1.28790993 -1.19142477  1.16600392
## Toyota Corona        0.23384555 -1.2248578 -0.89255318 -0.72469984  0.19345729
## Dodge Challenger    -0.76168319  1.0148821  0.70420401  0.04831332 -1.56460776
## AMC Javelin         -0.81145962  1.0148821  0.59124494  0.04831332 -0.83519779
## Camaro Z28          -1.12671039  1.0148821  0.96239618  1.43390296  0.24956575
## Pontiac Firebird    -0.14777380  1.0148821  1.36582144  0.41294217 -0.96611753
## Fiat X1-9            1.19619000 -1.2248578 -1.22416874 -1.17683962  0.90416444
## Porsche 914-2        0.98049211 -1.2248578 -0.89093948 -0.81221077  1.55876313
## Lotus Europa         1.71054652 -1.2248578 -1.09426581 -0.49133738  0.32437703
## Ford Pantera L      -0.71190675  1.0148821  0.97046468  1.71102089  1.16600392
```

```r
map2(LETTERS[1:3], letters[1:3], paste0)
```

```
## [[1]]
## [1] "Aa"
##
## [[2]]
## [1] "Bb"
##
## [[3]]
## [1] "Cc"
```

```r
modify2(LETTERS[1:3], letters[1:3], paste0)
```

```
## [1] "Aa" "Bb" "Cc"
```

# safely

# Errors during iterations

Sometimes a loop will work for most cases, but return an error on a few

Often, you want to return the output you can

Alternatively, you might want to diagnose *where* the error is occurring

`purrr::safely()`

# Example

```
by_cyl <- mpg %>%
  group_by(cyl) %>%
  nest()
by_cyl
```

```
## # A tibble: 4 × 2
## # Groups:   cyl [4]
##     cyl data
##   <int> <list>
## 1     4 <tibble [81 × 10]>
## 2     6 <tibble [79 × 10]>
## 3     8 <tibble [70 × 10]>
## 4     5 <tibble [4 × 10]>
```

Please run the code above

01:00

# Try to fit a model

(please follow along)

Notice the error message is *super* helpful! (this is new)

```
by_cyl %>%
  mutate(mod = map(data, ~lm(hwy ~ displ + drv, data = .x)))
```

```
## Error in `mutate()`:
## ! Problem while computing `mod = map(data, ~lm(hwy ~ displ +
##   drv, data = .x))`.
## ℹ The error occurred in group 2: cyl = 5.
## Caused by error in `contrasts<-`:
## ! contrasts can be applied only to factors with 2 or more levels
```

# Safe return

- First, define safe function – note that this will work for any function

```
safe_lm <- safely(lm)
```

- Next, loop the safe function, instead of the standard function

```
safe_models <- by_cyl %>%
  mutate(safe_mod = map(data, ~safe_lm(hwy ~ displ + drv, data = .x)))
safe_models
```

```
## # A tibble: 4 × 3
## # Groups:   cyl [4]
##     cyl data               safe_mod
##   <int> <list>             <list>
## 1     4 <tibble [81 × 10]> <named list [2]>
## 2     6 <tibble [79 × 10]> <named list [2]>
## 3     8 <tibble [70 × 10]> <named list [2]>
## 4     5 <tibble [4 × 10]>  <named list [2]>
```

# What's returned?

```r
safe_models$safe_mod[[1]]
```

```
## $result
## 
## Call:
## .f(formula = ..1, data = ..2)
## 
## Coefficients:
## (Intercept)        displ         drvf  
##      37.370       -5.289        3.882  
## 
## 
## $error
## NULL
```

```r
safe_models$safe_mod[[4]]
```

```
## $result
## NULL
## 
## $error
## <simpleError in `contrasts<-`(`*tmp*`, value = contr.funs[1 + isOF[nn]]): contrasts ca
```

# Inspecting

I often use `safely()` to help me de–bug. Why is it failing *there* (but note the new error messages help with this too).

First – create a new variable to filter for results with errors

```
safe_models %>%
  mutate(error = map_lgl(safe_mod, ~!is.null(.x$error)))
```

```
## # A tibble: 4 × 4
## # Groups:   cyl [4]
##     cyl data                safe_mod         error
##   <int> <list>              <list>           <lgl>
## 1     4 <tibble [81 × 10]> <named list [2]> FALSE
## 2     6 <tibble [79 × 10]> <named list [2]> FALSE
## 3     8 <tibble [70 × 10]> <named list [2]> FALSE
## 4     5 <tibble [4 × 10]>  <named list [2]> TRUE
```

# Inspecting the data

```r
safe_models %>%
  mutate(error = map_lgl(safe_mod, ~!is.null(.x$error))) %>%
  filter(isTRUE(error)) %>%
  select(cyl, data) %>%
  unnest(data)
```

```
## # A tibble: 4 × 11
## # Groups:   cyl [1]
##     cyl manufacturer model      displ  year trans      drv     cty   hwy fl   
##   <int> <chr>        <chr>      <dbl> <int> <chr>      <chr> <int> <int> <chr>
## 1     5 volkswagen   jetta        2.5  2008 auto(s6)   f        21    29 r    
## 2     5 volkswagen   jetta        2.5  2008 manual(m5) f        21    29 r    
## 3     5 volkswagen   new beetle   2.5  2008 manual(m5) f        20    28 r    
## 4     5 volkswagen   new beetle   2.5  2008 auto(s6)   f        20    29 r    
## # … with 1 more variable: class <chr>
```

The `displ` and `drv` variables are constant, so no relation can be estimated.

# Pull results that worked

```r
safe_models %>%
  mutate(results = map(safe_mod, "result"))
```

```
## # A tibble: 4 × 4
## # Groups:   cyl [4]
##     cyl data               safe_mod         results
##   <int> <list>             <list>           <list>
## 1     4 <tibble [81 × 10]> <named list [2]> <lm>
## 2     6 <tibble [79 × 10]> <named list [2]> <lm>
## 3     8 <tibble [70 × 10]> <named list [2]> <lm>
## 4     5 <tibble [4 × 10]>  <named list [2]> <NULL>
```

Now we can `broom::tidy()` or whatevs

## Notice that there is no `cyl == 5`.

```
safe_models %>%
  mutate(results = map(safe_mod, "result"),
         tidied = map(results, broom::tidy)) %>%
  select(cyl, tidied) %>%
  unnest(tidied)
```

```
## # A tibble: 11 × 6
## # Groups:   cyl [3]
##      cyl term          estimate std.error  statistic      p.value
##    <int> <chr>            <dbl>     <dbl>      <dbl>        <dbl>
##  1     4 (Intercept) 37.37023   3.537572  10.56381   1.052943e-16
##  2     4 displ       -5.288562  1.436068  -3.682668  4.235795e- 4
##  3     4 drvf         3.882134  0.9971876  3.893083  2.073699e- 4
##  4     6 (Intercept) 27.96536   2.347630  11.91217   5.718039e-19
##  5     6 displ       -2.333261  0.6373304 -3.660991  4.651570e- 4
##  6     6 drvf         4.570840  0.6012367  7.602397  6.789988e-11
##  7     6 drvr         6.384355  1.229277   5.193585  1.713129e- 6
##  8     8 (Intercept) 14.82265   2.887289   5.133759  2.708515e- 6
##  9     8 displ        0.3060487 0.5719058  0.5351383 5.943528e- 1
## 10     8 drvf         8.555294  2.679129   3.193311  2.156229e- 3
## 11     8 drvr         3.709336  0.7319048  5.068058  3.473594e- 6
```

# When else might we use this?

Any sort of web scraping – pages change and URLs don't always work

# Example

```r
library(rvest)
links <- list(
  "https://en.wikipedia.org/wiki/FC_Barcelona",
  "https://nosuchpage",
  "https://en.wikipedia.org/wiki/Rome"
)
pages <- map(links, ~{
  Sys.sleep(0.1)
  read_html(.x)
})
```

```
## Error in open.connection(x, "rb"): Failed to connect to nosuchpage port 443: Connecti
```

# The problem

I can't connect to https://nosuchpage because it doesn't exist

BUT

That also means I can't get *any* of my links because *one* page errored (imagine it was 1 in 1,000 instead of 1 in 3)

`safely()` to the rescue

# Safe version

```
safe_read_html <- safely(read_html)
pages <- map(links, ~{
  Sys.sleep(0.1)
  safe_read_html(.x)
})
str(pages)
```

```
## List of 3
##  $ :List of 2
##   ..$ result:List of 2
##   .. ..$ node:<externalptr>
##   .. ..$ doc :<externalptr>
##   .. ..- attr(*, "class")= chr [1:2] "xml_document" "xml_node"
##   ..$ error : NULL
##  $ :List of 2
##   ..$ result: NULL
##   ..$ error :List of 2
##   .. ..$ message: chr "Timeout was reached: [nosuchpage] Failed to connect to nosuchpa
##   .. ..$ call   : language open.connection(x, "rb")
##   .. ..- attr(*, "class")= chr [1:3] "simpleError" "error" "condition"
##  $ :List of 2
##   ..$ result:List of 2
##   .. ..$ node:<externalptr>
##   .. ..$ doc :<externalptr>
##   .. ..- attr(*, "class")= chr [1:2] "xml_document" "xml_node"
##   ..$ error : NULL
```

# Non-results

In a real example, we'd probably want to double-check the pages where we got no results

```
errors <- map_lgl(pages, ~!is.null(.x$error))
links[errors]
```

```
## [[1]]
## [1] "https://nosuchpage"
```

# reduce

# Reducing a list

The `map()` family of functions will always return a vector the same length as the input

`reduce()` will collapse or reduce the list to a single element

# Example

```r
l <- list(
  c(1, 3),
  c(1, 5, 7, 9),
  3,
  c(4, 8, 12, 2)
)

reduce(l, sum)
```

```
## [1] 55
```

# Compare to `map()`

```r
map(l, sum)
```

```
## [[1]]
## [1] 4
##
## [[2]]
## [1] 22
##
## [[3]]
## [1] 3
##
## [[4]]
## [1] 26
```

# What's going on?

The code `reduce(l, sum)` is the same as

```r
sum(l[[4]], sum(l[[3]], sum(l[[1]], l[[2]])))
```

```
## [1] 55
```

Or slidghlty differently

```r
first_sum <- sum(l[[1]], l[[2]])
second_sum <- sum(first_sum, l[[3]])
final_sum <- sum(second_sum, l[[4]])
final_sum
```

```
## [1] 55
```

# Why might you use this?

What if you had a list of data frames like this

```
l_df <- list(
  tibble(id = 1:3, score = rnorm(3)),
  tibble(id = 1:5, treatment = rbinom(5, 1, .5)),
  tibble(id = c(1, 3, 5, 7), other_thing = rnorm(4))
)
```

We can join these all together with a single loop – we want the output to be of length 1!

```r
reduce(l_df, full_join)
```

```
## # A tibble: 6 × 4
##      id      score treatment other_thing
##   <dbl>      <dbl>     <int>       <dbl>
## 1     1 -1.251776          1  -1.191128
## 2     2  1.074910          1  NA
## 3     3 -0.9096793         1   1.555894
## 4     4 NA                 0  NA
## 5     5 NA                 0  -1.420784
## 6     7 NA                NA   0.6982053
```

# Note – you have to be careful on directionality

```
reduce(l_df, left_join)
```

```
## # A tibble: 3 × 4
##       id      score treatment other_thing
##    <dbl>      <dbl>     <int>       <dbl>
## 1     1 -1.251776          1   -1.191128
## 2     2  1.074910          1   NA
## 3     3 -0.9096793         1    1.555894
```

```
reduce(l_df, right_join)
```

```
## # A tibble: 4 × 4
##       id      score treatment other_thing
##    <dbl>      <dbl>     <int>       <dbl>
## 1     1 -1.251776          1  -1.191128
## 2     3 -0.9096793         1   1.555894
## 3     5 NA                 0  -1.420784
## 4     7 NA                NA   0.6982053
```

# Another example

You probably just want to `bind_rows()`

```r
l_df2 <- list(
  tibble(id = 1:3, scid = 1, score = rnorm(3)),
  tibble(id = 1:5, scid = 2, score = rnorm(5)),
  tibble(id = c(1, 3, 5, 7), scid = 3, score = rnorm(4))
)
reduce(l_df2, bind_rows)
```

```
## # A tibble: 12 × 3
##       id  scid       score
##    <dbl> <dbl>       <dbl>
##  1     1     1  1.671069
##  2     2     1 -0.2444534
##  3     3     1 -0.1864957
##  4     1     2 -0.7310893
##  5     2     2  0.1925646
##  6     3     2  1.797250
##  7     4     2  0.2418765
##  8     5     2  3.328079
##  9     1     3  0.7411117
## 10     3     3  0.8653901
## 11     5     3 -0.1097661
## 12     7     3  0.09372232
```

# Non–loop version

Luckily, the prior slide has become obsolete, because `bind_rows()` will do the list reduction for us.

```
bind_rows(l_df2)
```

```
## # A tibble: 12 × 3
##       id  scid       score
##    <dbl> <dbl>       <dbl>
##  1     1     1  1.671069
##  2     2     1 -0.2444534
##  3     3     1 -0.1864957
##  4     1     2 -0.7310893
##  5     2     2  0.1925646
##  6     3     2  1.797250
##  7     4     2  0.2418765
##  8     5     2  3.328079
##  9     1     3  0.7411117
## 10     3     3  0.8653901
## 11     5     3 -0.1097661
## 12     7     3  0.09372232
```

# Another example

This is a poor example, but there are use cases like this

```r
library(palmerpenguins)
map(penguins, as.character) %>%
  reduce(paste)
```

```
##   [1] "Adelie Torgersen 39.1 18.7 181 3750 male 2007"
##   [2] "Adelie Torgersen 39.5 17.4 186 3800 female 2007"
##   [3] "Adelie Torgersen 40.3 18 195 3250 female 2007"
##   [4] "Adelie Torgersen NA NA NA NA NA 2007"
##   [5] "Adelie Torgersen 36.7 19.3 193 3450 female 2007"
##   [6] "Adelie Torgersen 39.3 20.6 190 3650 male 2007"
##   [7] "Adelie Torgersen 38.9 17.8 181 3625 female 2007"
##   [8] "Adelie Torgersen 39.2 19.6 195 4675 male 2007"
##   [9] "Adelie Torgersen 34.1 18.1 193 3475 NA 2007"
##  [10] "Adelie Torgersen 42 20.2 190 4250 NA 2007"
##  [11] "Adelie Torgersen 37.8 17.1 186 3300 NA 2007"
##  [12] "Adelie Torgersen 37.8 17.3 180 3700 NA 2007"
##  [13] "Adelie Torgersen 41.1 17.6 182 3200 female 2007"
##  [14] "Adelie Torgersen 38.6 21.2 191 3800 male 2007"
##  [15] "Adelie Torgersen 34.6 21.1 198 4400 male 2007"
##  [16] "Adelie Torgersen 36.6 17.8 185 3700 female 2007"
##  [17] "Adelie Torgersen 38.7 19 195 3450 female 2007"
##  [18] "Adelie Torgersen 42.5 20.7 197 4500 male 2007"
##  [19] "Adelie Torgersen 34.4 18.4 184 3325 female 2007"
##  [20] "Adelie Torgersen 46 21.5 194 4200 male 2007"
##  [21] "Adelie Biscoe 37.8 18.3 174 3400 female 2007"
##  [22] "Adelie Biscoe 37.7 18.7 180 3600 male 2007"
##  [23] "Adelie Biscoe 35.9 19.2 189 3800 female 2007"
##  [24] "Adelie Biscoe 38.2 18.1 185 3950 male 2007"
```

# Why use `reduce()`

This is one that I use a fair bit, but have a hard time coming up with good examples for.

The tidyverse makes it less needed, generally.

Still a good "tool" to have

# Wrap up

- Lots more to `{purrr}` but we've covered a lot
- Functional programming can *really* help your efficiency, and even if it slows you down initially, I'd recommend always striving toward it, because it will ultimately be a huge help.

## Questions?

If we have any time left – let's work on the homework

# Next time (fully remote)

## Functions

Beginning next class, the focus of the course will shift