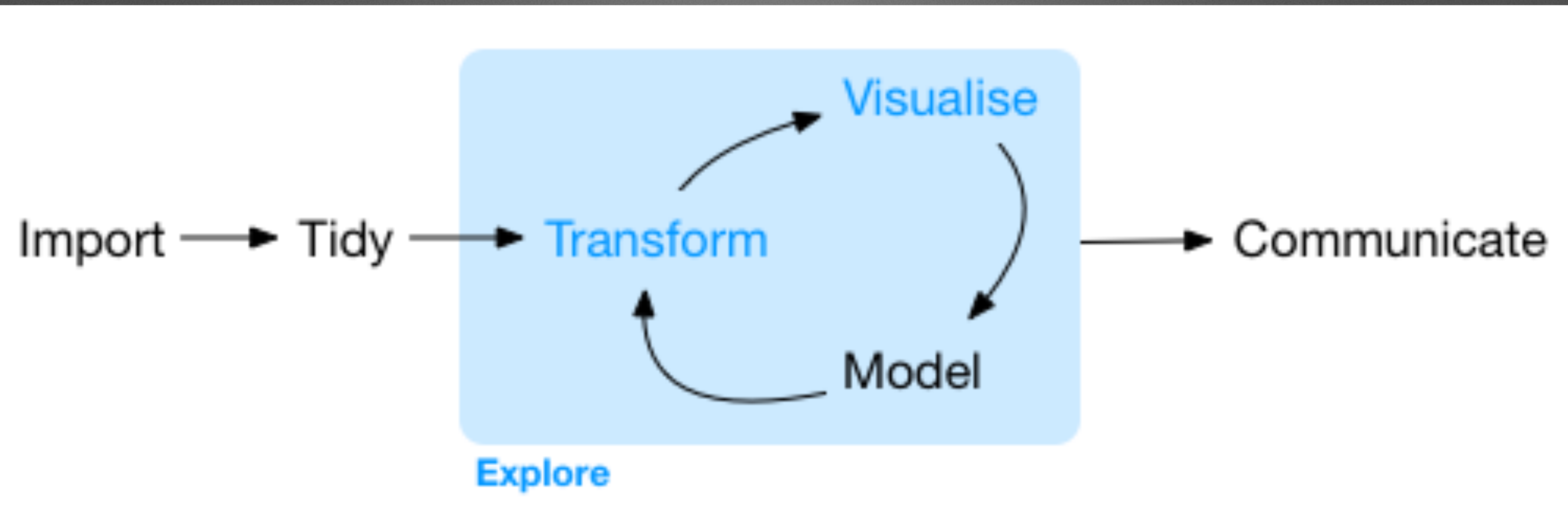
tidyverse[c(“magrittr”, ”dplyr”, ”tidyr”)]

Barret Schloerke
Statistics PhD Candidate
Purdue University

About myself

- **Purdue University**
  - 5th Year PhD Candidate in Statistics
  - Research in large data visualization using R - deltarho.org
    - Dr. William Cleveland and Dr. Ryan Hafen
- Metamarkets.com - 1.5 years
  - Front end engineer - node.js
- **Iowa State University**
  - B.S. in Computer Engineering
  - Research in statistical data visualization with R
    - Dr. Di Cook, Dr. Hadley Wickham, and Dr. Heike Hofmann

Exploratory Data Analysis



http://r4ds.had.co.nz/explore-intro.html

Magrittr

- “structuring sequences of data operations left-to-right (as opposed to from the inside and out),
- avoiding nested function calls,
- minimizing the need for local variables and function definitions, and
- making it easy to add steps anywhere in the sequence of operations.”

https://github.com/tidyverse/magrittr

Children's Story Example

```r
foo_foo <- little_bunny()

bop_on(
  scoop_up(
    hop_through(
      foo_foo,
      forest
    ),
    field_mouse
  ),
  head
)
```

```r
foo_foo %>%
  hop_through(forest) %>%
  scoop_up(field_mouse) %>%
  bop_on(head)
```

https://twitter.com/_inundata/status/557980236130689024

Tidy Data

- "Happy families are all alike; every unhappy family is unhappy in its own way." — Leo Tolstoy
- "Tidy datasets are all alike, but every messy dataset is messy in its own way." — Hadley Wickham

http://r4ds.had.co.nz/tidy-data.html

Tidy Data

variables

observations

values

http://r4ds.had.co.nz/tidy-data.html

Tidy Data

country	year	cases
Afghanistan	1999	745
Afghanistan	2000	2666
Brazil	1999	37737
Brazil	2000	80488
China	1999	212258
China	2000	213766

country	1999	2000
Afghanistan	745	2666
Brazil	37737	80488
China	212258	213766

table4

http://r4ds.had.co.nz/tidy-data.html

Tidy Data

country	year	key	value
Afghanistan	1999	cases	745
Afghanistan	1999	population	19987071
Afghanistan	2000	cases	2666
Afghanistan	2000	population	20595360
Brazil	1999	cases	37737
Brazil	1999	population	172006362
Brazil	2000	cases	80488
Brazil	2000	population	174504898
China	1999	cases	212258
China	1999	population	1272915272
China	2000	cases	213766
China	2000	population	1280428583

table2

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20595360
Brazil	1999	37737	172006362
Brazil	2000	80488	174504898
China	1999	212258	1272915272
China	2000	213766	1280428583

http://r4ds.had.co.nz/tidy-data.html

Tidy Data

- Disadvantages
  - More work after data is loaded
  - May require slightly more memory (~1 - 3x)
- Advantage
  - Consistent storage structure speeds up learning curve
  - Leverage the power of R’s vector operations

http://r4ds.had.co.nz/tidy-data.html

When to NOT use tidy Data

- Alternative representations may have substantial performance or space advantages.
- Specialized fields have evolved their own conventions for storing data that may be quite different to the conventions of tidy data.

http://r4ds.had.co.nz/tidy-data.html

dplyr

- dplyr is the next iteration of plyr, focussed on tools for working with data frames (hence the d in the name). It has three main goals:
  - Identify the most important data manipulation tools needed for data analysis and make them **easy to use** from R.
  - Provide blazing **fast performance** for in-memory data by writing key pieces in C++.
  - Use the **same interface** to work with data no matter where it's stored, whether in a data frame, a data table or database.

https://github.com/hadley/dplyr

dplyr verbs

- select(): focus on a subset of variables
- filter(): focus on a subset of rows
- mutate(): add new columns
- summarise(): reduce each group to a smaller number of summary statistics
- arrange(): re-order the rows
- group_by(): add subset rule for summarizations

https://github.com/hadley/dplyr

dplyr example

```
library(nycflights13)
dim(flights)
#> [1] 336776     19
head(flights)
#> # A tibble: 6 x 19
#>    year month   day dep_time sched_dep_time dep_delay arr_time
#>   <int> <int> <int>    <int>          <int>     <dbl>    <int>
#> 1  2013     1     1      517            515         2      830
#> 2  2013     1     1      533            529         4      850
#> 3  2013     1     1      542            540         2      923
#> 4  2013     1     1      544            545        -1     1004
#> ... with 2 more rows, and 12 more variables: sched_arr_time <int>,
#>   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
#>   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour
<dbl>,
#>   minute <dbl>, time_hour <time>
```

https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

dplyr::mutate

```
mutate(flights,
  gain = arr_delay - dep_delay,
  gain_per_hour = gain / (air_time / 60)
)
#> # A tibble: 336,776 x 21
#>     year month   day dep_time sched_dep_time dep_delay arr_time
#>    <int> <int> <int>    <int>          <int>     <dbl>    <int>
#> 1  2013     1     1      517            515         2      830
#> 2  2013     1     1      533            529         4      850
#> 3  2013     1     1      542            540         2      923
#> 4  2013     1     1      544            545        -1     1004
#> ... with 336,772 more rows, and 14 more variables: sched_arr_time <int>,
#>   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
#>   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
#>   minute <dbl>, time_hour <time>, gain <dbl>, gain_per_hour <dbl>
```

https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

dplyr::transmute

```
transmute(flights,
  gain = arr_delay - dep_delay,
  gain_per_hour = gain / (air_time / 60)
)
#> # A tibble: 336,776 x 2
#>    gain gain_per_hour
#>   <dbl>         <dbl>
#> 1     9      2.378855
#> 2    16      4.229075
#> 3    31     11.625000
#> 4   -17     -5.573770
#> ... with 336,772 more rows
```

https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

dplyr::summarise

```
summarise(flights,
  delay = mean(dep_delay, na.rm = TRUE))
#> # A tibble: 1 x 1
#>       delay
#>       <dbl>
#> 1 12.63907
```

https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

dplyr::sample_n
dplyr::sample_frac

```
sample_n(flights, 10)
#> # A tibble: 10 x 19
#>    year month   day dep_time sched_dep_time dep_delay arr_time
#>   <int> <int> <int>    <int>          <int>     <dbl>    <int>
#> 1  2013     7     8     2205           2019       106      103
#> 2  2013     9    12     1602           1545        17       NA
#> 3  2013    11     4     1459           1459         0     1642
#> 4  2013    10    25     1354           1350         4     1534
#> ... with 6 more rows, and 12 more variables: sched_arr_time <int>,
#>   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
#>   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
#>   minute <dbl>, time_hour <time>

sample_frac(flights, 0.01)
#> # A tibble: 3,368 x 19
#>    year month   day dep_time sched_dep_time dep_delay arr_time
#>   <int> <int> <int>    <int>          <int>     <dbl>    <int>
#> 1  2013     5    14      850            850         0     1237
#> 2  2013    11     8      832            840        -8     1016
#> 3  2013    12     1     1155           1155         0     1309
#> 4  2013     1     1      929            925         4     1220
#> ... with 3,364 more rows, and 12 more variables: sched_arr_time <int>,
#>   arr_delay <dbl>, carrier <chr>, flight <int>, tailnum <chr>,
#>   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
#>   minute <dbl>, time_hour <time>
```

https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

dplyr Example

```
flights %>%
  group_by(year, month, day) %>%
  select(arr_delay, dep_delay) %>%
  summarise(
    arr = mean(arr_delay, na.rm = TRUE),
    dep = mean(dep_delay, na.rm = TRUE)
  ) %>%
  filter(arr > 30 | dep > 30)
```

```
Source: local data frame [49 x 5]
Groups: year, month [11]

    year month   day      arr      dep
   <int> <int> <int>    <dbl>    <dbl>
1   2013     1    16 34.24736 24.61287
2   2013     1    31 32.60285 28.65836
3   2013     2    11 36.29009 39.07360
4   2013     2    27 31.25249 37.76327
5   2013     3     8 85.86216 83.53692
6   2013     3    18 41.29189 30.11796
7   2013     4    10 38.41231 33.02368
8   2013     4    12 36.04814 34.83843
9   2013     4    18 36.02848 34.91536
10  2013     4    19 47.91170 46.12783
# ... with 39 more rows
```

https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

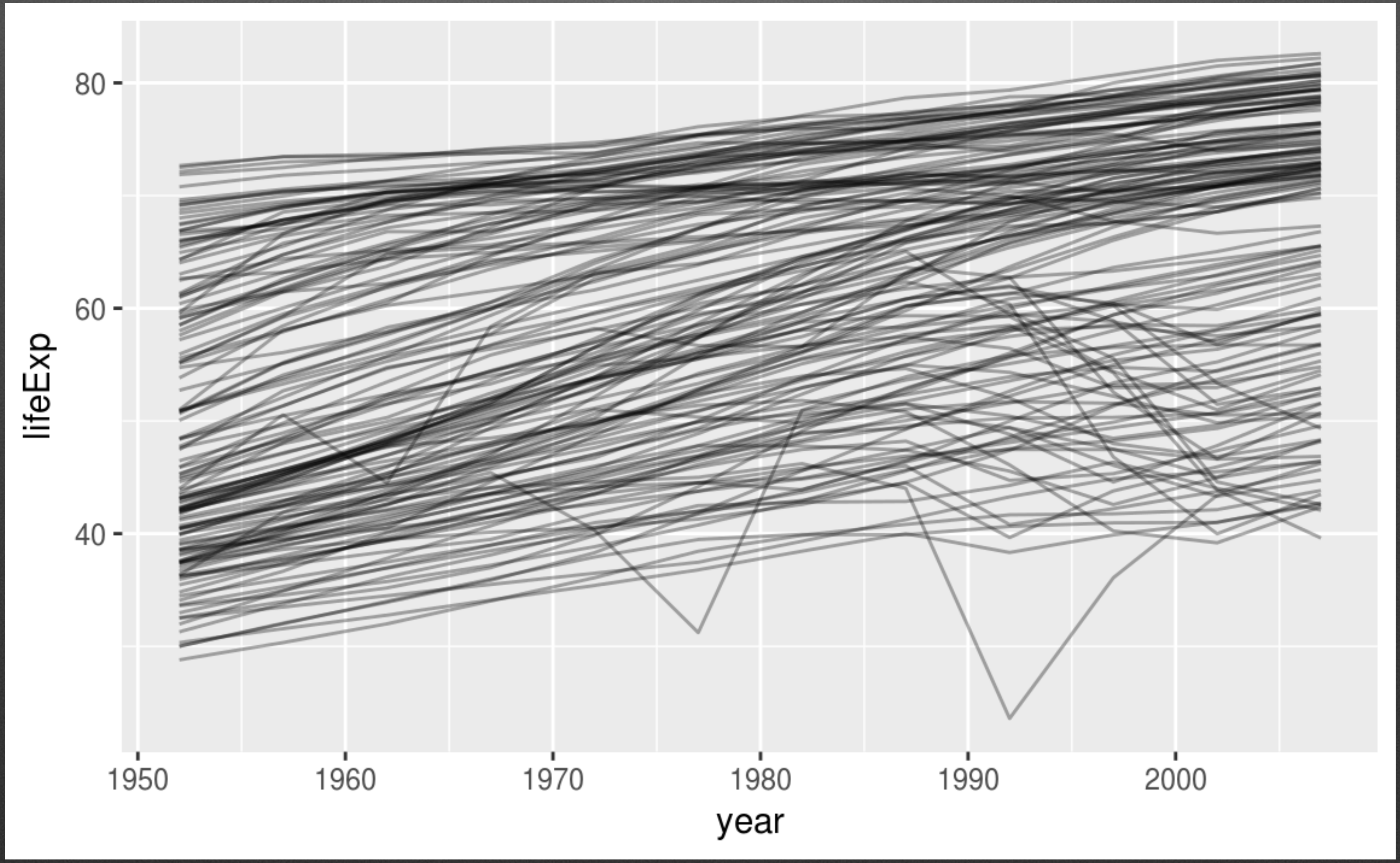
gapminder

```
library(gapminder)
gapminder
#> # A tibble: 1,704 x 6
#>       country continent  year lifeExp      pop gdpPercap
#>        <fctr>    <fctr> <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
#> # ... with 1,698 more rows
```

http://r4ds.had.co.nz/many-models.html#nested-data

gapminder

```
gapminder %>%
  ggplot(aes(year, lifeExp, group = country)) +
    geom_line(alpha = 1/3)
```



http://r4ds.had.co.nz/many-models.html#nested-data

`gapminder` goals

- Want to compute a linear model for each country
- Want to see how well each country's life expectancy follows a linear model

http://r4ds.had.co.nz/many-models.html#nested-data

`data.frame()` rules

- All `data.frame()`'s are lists
- All columns have the same length
- Each column (or list section) is of the same type

- Important!!
  - Lists may contain lists!

gapminder

```
by_country <- gapminder %>%
  group_by(country, continent) %>%
  nest()

by_country
# A tibble: 142 x 3
       country continent              data
        <fctr>    <fctr>            <list>
1  Afghanistan      Asia <tibble [12 x 4]>
2      Albania    Europe <tibble [12 x 4]>
3      Algeria    Africa <tibble [12 x 4]>
4       Angola    Africa <tibble [12 x 4]>
5    Argentina  Americas <tibble [12 x 4]>
6    Australia   Oceania <tibble [12 x 4]>
7      Austria    Europe <tibble [12 x 4]>
8      Bahrain      Asia <tibble [12 x 4]>
9   Bangladesh      Asia <tibble [12 x 4]>
10     Belgium    Europe <tibble [12 x 4]>
# ... with 132 more rows
```

http://r4ds.had.co.nz/many-models.html#nested-data

gapminder: find model

```r
country_model <- function(df) {
  lm(lifeExp ~ year, data = df)
}

models <- map(by_country$data, country_model)
```

http://r4ds.had.co.nz/many-models.html#nested-data

gapminder: find model

```
by_country <- by_country %>%
  mutate(model = purrr::map(data, country_model))
by_country
#> # A tibble: 142 x 4
#>       country continent               data    model
#>        <fctr>    <fctr>             <list>   <list>
#> 1 Afghanistan      Asia <tibble [12 x 4]> <S3: lm>
#> 2     Albania    Europe <tibble [12 x 4]> <S3: lm>
#> 3     Algeria    Africa <tibble [12 x 4]> <S3: lm>
#> 4      Angola    Africa <tibble [12 x 4]> <S3: lm>
#> 5   Argentina  Americas <tibble [12 x 4]> <S3: lm>
#> 6   Australia   Oceania <tibble [12 x 4]> <S3: lm>
#> # ... with 136 more rows
```

http://r4ds.had.co.nz/many-models.html#nested-data

gapminder: residuals (use dplyr!)

```
by_country <- by_country %>%
  mutate(
    resids = purrr::map2(data, model, modelr::add_residuals)
  )
by_country
#> # A tibble: 142 x 5
#>       country continent               data     model              resids
#>        <fctr>    <fctr>             <list>    <list>              <list>
#> 1 Afghanistan      Asia <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 2     Albania    Europe <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 3     Algeria    Africa <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 4      Angola    Africa <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 5   Argentina  Americas <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 6   Australia   Oceania <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> # ... with 136 more rows
```

http://r4ds.had.co.nz/many-models.html#nested-data

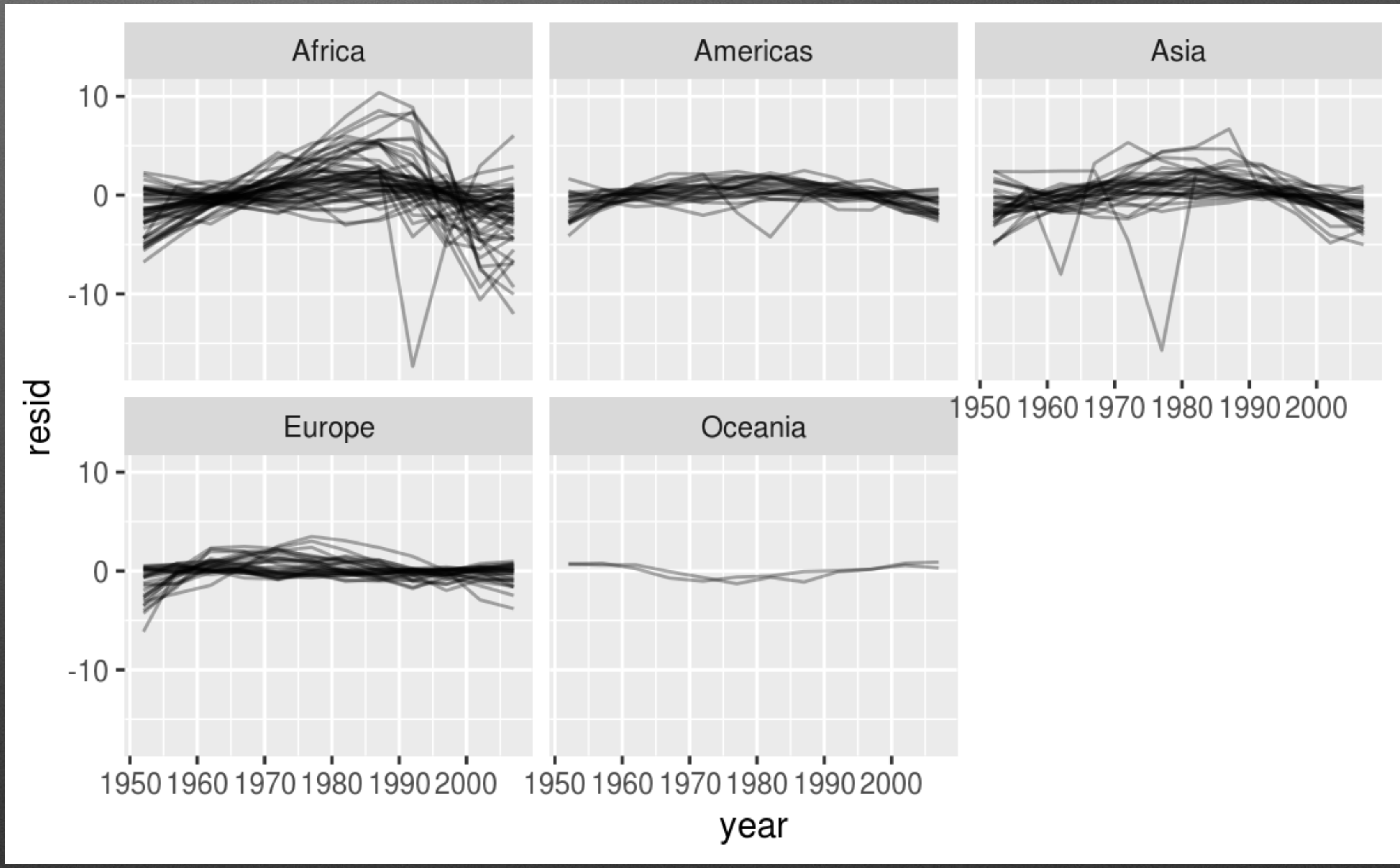
gapminder: unnest

```
resids <- unnest(by_country, resids)
resids
#> # A tibble: 1,704 x 7
#>        country continent  year lifeExp      pop gdpPercap   resid
#>         <fctr>    <fctr> <int>   <dbl>    <int>     <dbl>   <dbl>
#> 1 Afghanistan      Asia  1952    28.8  8425333       779 -1.1063
#> 2 Afghanistan      Asia  1957    30.3  9240934       821 -0.9519
#> 3 Afghanistan      Asia  1962    32.0 10267083       853 -0.6636
#> 4 Afghanistan      Asia  1967    34.0 11537966       836 -0.0172
#> 5 Afghanistan      Asia  1972    36.1 13079460       740  0.6741
#> 6 Afghanistan      Asia  1977    38.4 14880372       786  1.6475
#> # ... with 1,698 more rows
```

http://r4ds.had.co.nz/many-models.html#nested-data

gapminder: model quality

```
resids %>%
  ggplot(
    aes(
      x = year,
      y = resid,
      group = country
    )
  ) +
    geom_line(
      alpha = 1 / 3
    ) +
    facet_wrap(
      ~continent
    )
```



http://r4ds.had.co.nz/many-models.html#unnesting

gapminder: model quality

```
by_country %>%
  mutate(glance = purrr::map(model, broom::glance)) %>%
  unnest(glance)
#> # A tibble: 142 x 16
#>       country continent              data     model               resids
#>        <fctr>    <fctr>            <list>    <list>               <list>
#> 1 Afghanistan      Asia <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 2     Albania    Europe <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 3     Algeria    Africa <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 4      Angola    Africa <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 5   Argentina  Americas <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> 6   Australia   Oceania <tibble [12 x 4]> <S3: lm> <tibble [12 x 5]>
#> # ... with 136 more rows, and 11 more variables: r.squared <dbl>,
#> #   adj.r.squared <dbl>, sigma <dbl>, statistic <dbl>, p.value <dbl>,
#> #   df <int>, logLik <dbl>, AIC <dbl>, BIC <dbl>, deviance <dbl>,
#> #   df.residual <int>
```

http://r4ds.had.co.nz/many-models.html#unnesting

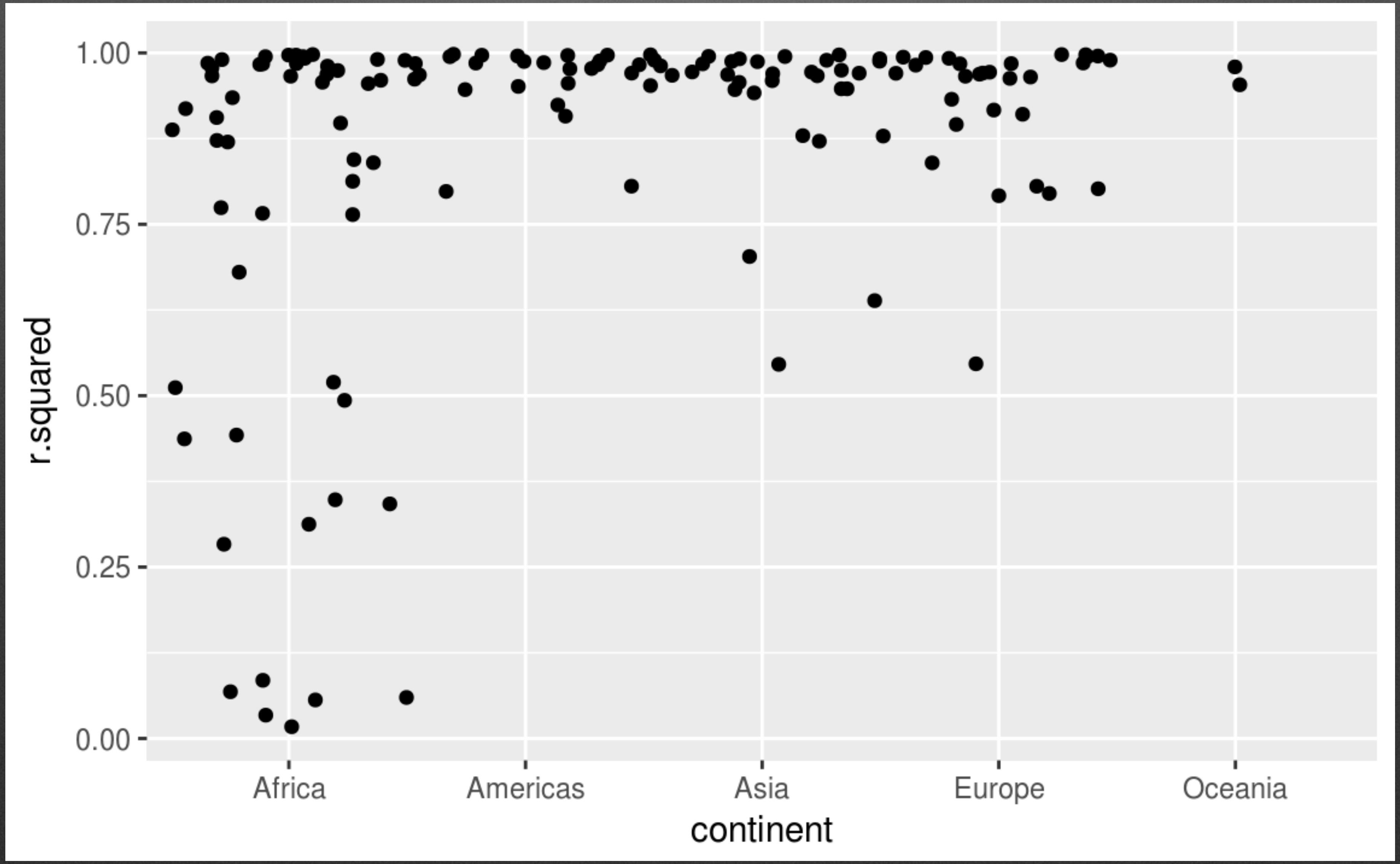
gapminder: model quality

```
glance <- by_country %>%
  mutate(glance = purrr::map(model, broom::glance)) %>%
  unnest(glance, .drop = TRUE)
glance
#> # A tibble: 142 x 13
#>       country continent r.squared adj.r.squared sigma statistic  p.value
#>        <fctr>    <fctr>     <dbl>         <dbl> <dbl>     <dbl>    <dbl>
#> 1 Afghanistan      Asia     0.948         0.942 1.223     181.2 9.84e-08
#> 2     Albania    Europe     0.911         0.902 1.983     101.8 1.46e-06
#> 3     Algeria    Africa     0.985         0.984 1.323     661.9 1.81e-10
#> 4      Angola    Africa     0.888         0.877 1.407      79.1 4.59e-06
#> 5   Argentina  Americas     0.996         0.995 0.292    2246.4 4.22e-13
#> 6   Australia   Oceania     0.980         0.978 0.621     481.3 8.67e-10
#> # ... with 136 more rows, and 6 more variables: df <int>, logLik <dbl>,
#> #   AIC <dbl>, BIC <dbl>, deviance <dbl>, df.residual <int>
```

http://r4ds.had.co.nz/many-models.html#unnesting

gapminder: model quality

```
glance %>%
  ggplot(aes(continent, r.squared)) +
    geom_jitter(width = 0.5)
```

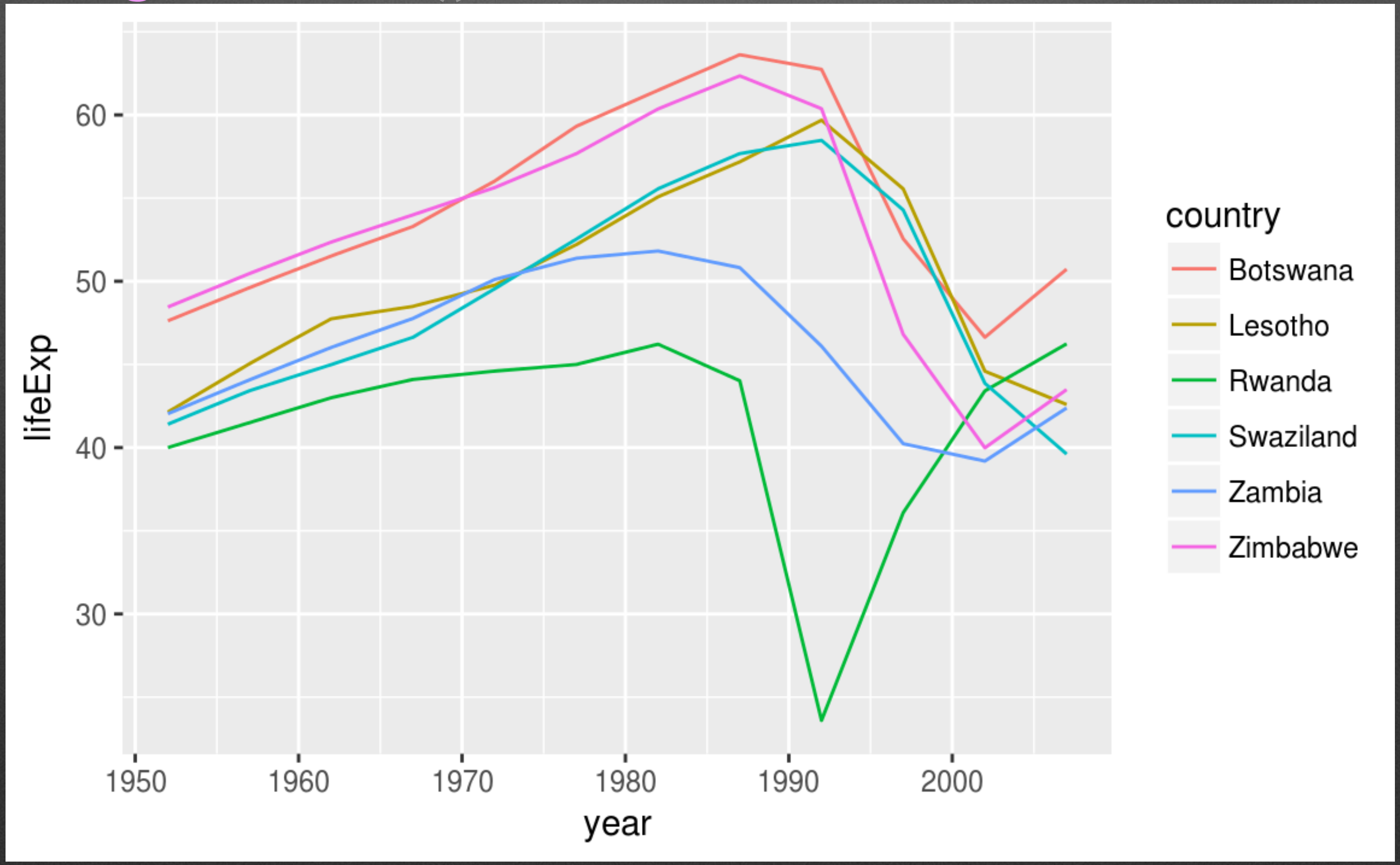


http://r4ds.had.co.nz/many-models.html#unnesting

gapminder: model quality

```
bad_fit <- filter(glance, r.squared < 0.25)

gapminder %>%
  semi_join(bad_fit, by = "country") %>%
  ggplot(aes(year, lifeExp, colour = country)) +
    geom_line()
```



http://r4ds.had.co.nz/many-models.html#unnesting

Recap

- `magrittr`
  - Keep code readable
- `dplyr`
  - Works with data.frames
- `tidyr`
  - Nest your data.frames

Questions?