```
library(dplyr)

rladies_global %>%
  filter(city == 'Austin')
```

R FOR DATA SCIENCE: PROGRAMMING -- MODELING WITH MODELR, PURR, and BROOM

Hello!

Welcome to R-Ladies

Thanks to our sponsors!

R Studio | Web.com

1. Introduction

R language, RStudio,
R4DS Workshop series

Three things
you'll need to install

1. **Install R** -- this is the open-source programming language we'll use (download via CRAN -- Comprehensive R Archive Network)
2. **Install RStudio** -- this is the most popular IDE for R and will make your life a lot easier (download from rstudio.com/download)
3. **Install the tidyverse** -- this is the group of packages we'll use within R to work with data. Install with one line of code in R:
   `install.packages("tidyverse")`

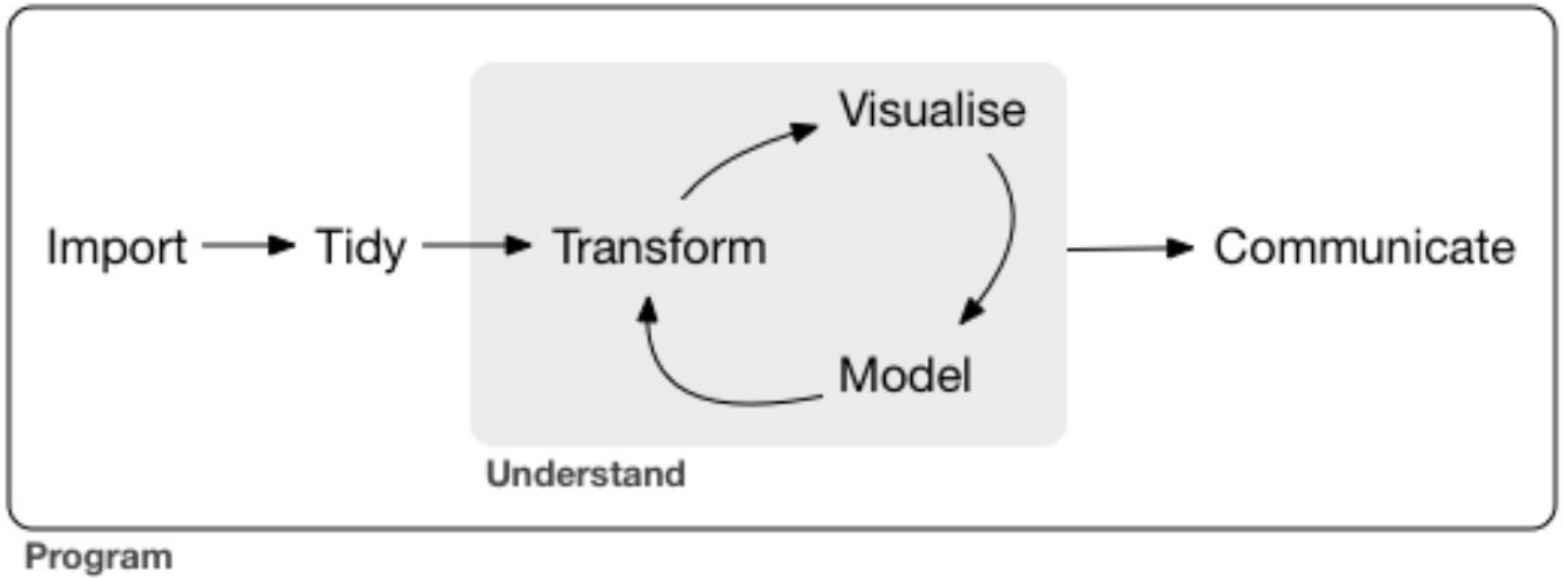
1b. Introduction

R for Data Science Workshop Series

R4DS
Workshop Series

- **Exploring Data with ggplot2 + dplyr** [DONE]
- **Exploratory Data Analysis and Workflow** [DONE]
- **Data Wrangling in the Tidyverse** [November 28]
- **Programming -- Functions, Vectors, and Iteration** [December 13]
- **Modeling with modelr, purrr, and broom** [January 24]
- **Communicating Results with rmarkdown and ggplot2** [February 21]

The data science process (tidied)



What is
the tidyverse?

- Collection of R packages based on tidy data principles
- Designed to work together
- An easier way to code!
- AKA “Hadleyverse” (most packages written by Hadley Wickham)

What is
the tidyverse?

What is
tidy data?

- Each variable is a column
- Each observation is a row
- Each type of observational unit is a table

id	artist	track	time
1	2 Pac	Baby Don't Cry	4:22
2	2Ge+her	The Hardest Part Of ...	3:15
3	3 Doors Down	Kryptonite	3:53
4	3 Doors Down	Loser	4:24
5	504 Boyz	Wobble Wobble	3:35
6	98^0	Give Me Just One Nig...	3:24
7	A*Teens	Dancing Queen	3:44
8	Aaliyah	I Don't Wanna	4:15
9	Aaliyah	Try Again	4:03
10	Adams, Yolanda	Open My Heart	5:30
11	Adkins, Trace	More	3:05
12	Aguilera, Christina	Come On Over Baby	3:38
13	Aguilera, Christina	I Turn To You	4:00
14	Aguilera, Christina	What A Girl Wants	3:18
15	Alice Deejay	Better Off Alone	6:50

2.
Model Basics

Model basics

What is a model?

“All models are wrong, but some are useful.”

-George Box

Models attempt to summarize/model a dataset in a simple way. Never discover truth, but discover a useful approximation.

Using modelr

```
library(tidyverse)


library(modelr)
options(na.action = na.warn)
```

Linear regression equation

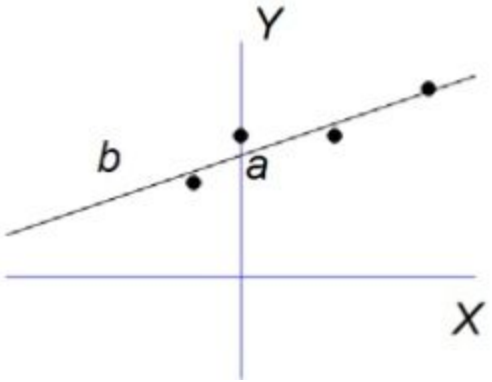
(without error)

$$\hat{Y} = bX + a$$

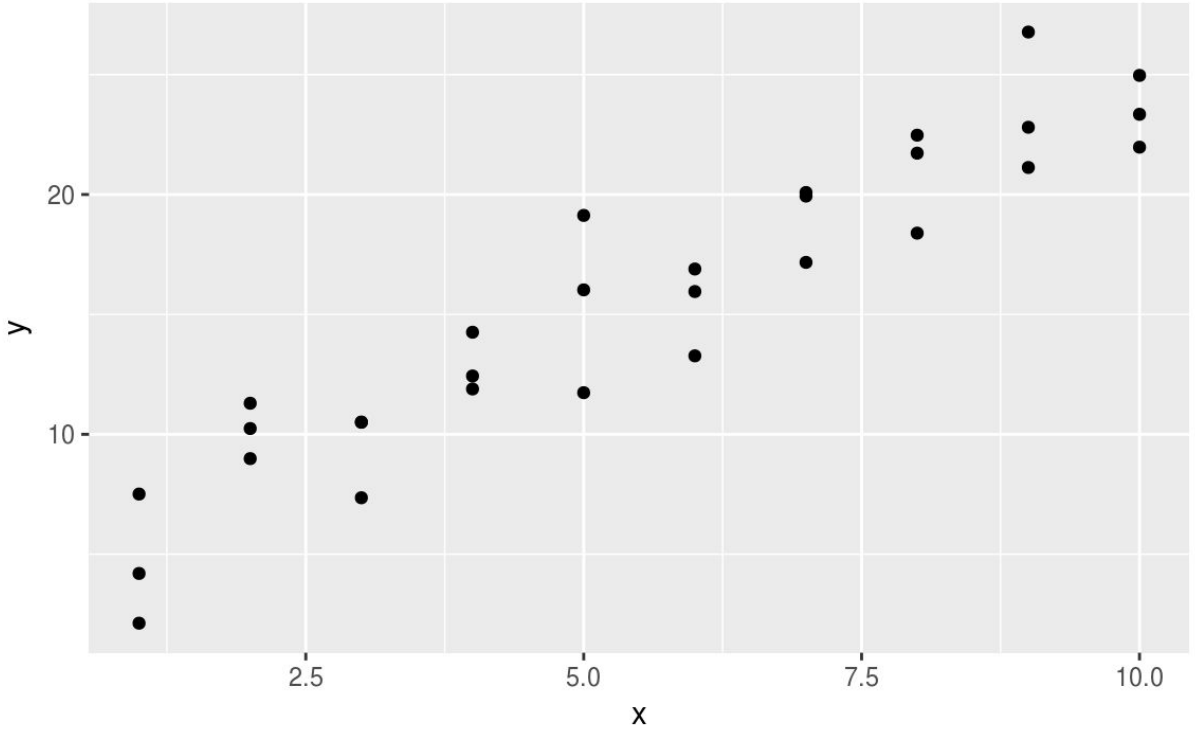
predicted values of *Y*

b = slope = rate of predicted ↑/↓ for Y scores for each unit increase in *X*

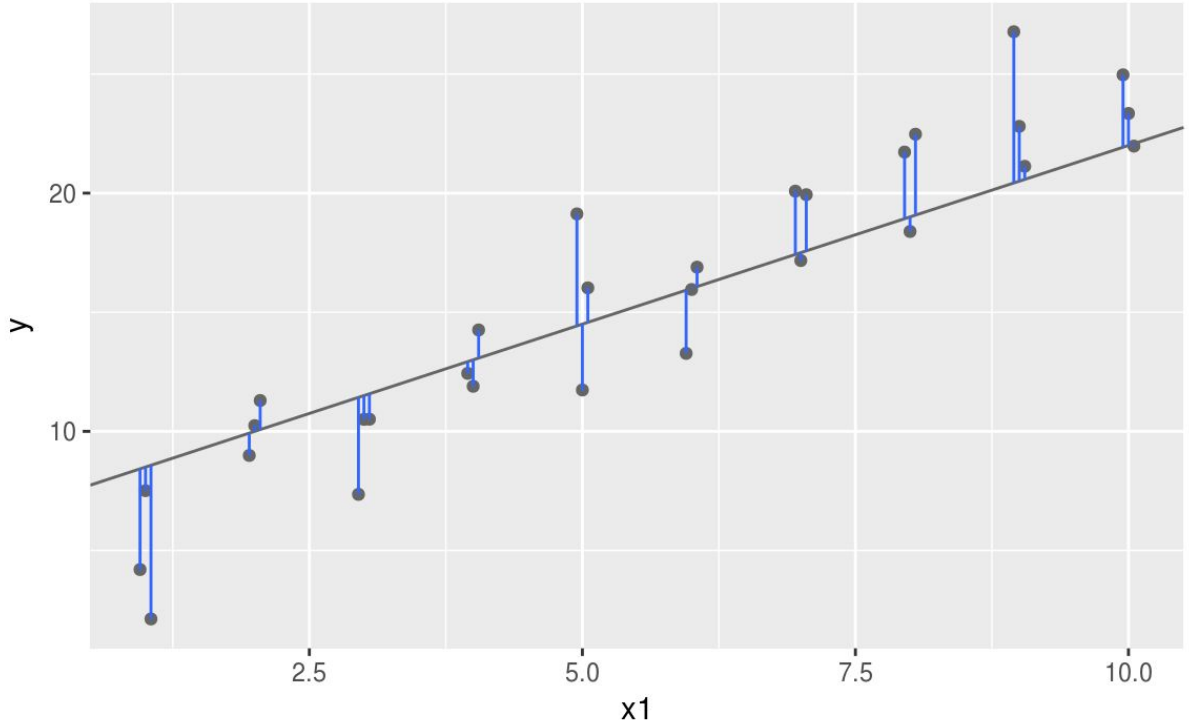
Y-intercept = level of *Y* when X is 0



Intro to a simple model



Intro to a simple model



A simple model-activity

See R script to follow along! It will cover the following:

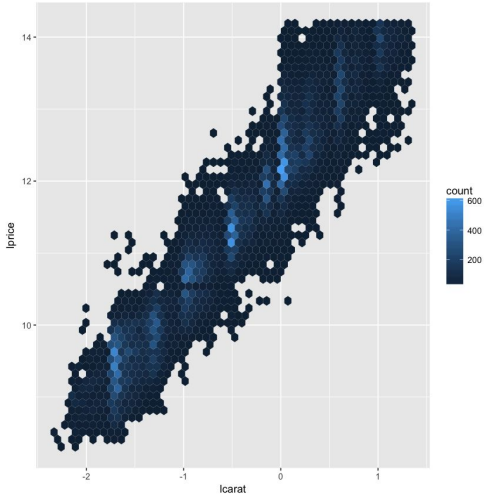
1) Creating a model
2) Visualizing model
    a) Predictions
    b) Residuals
3) Categorical variables
4) Interactions
    a) Continuous and categorical
5) Transformations
6) Missing Values

3.
Model Building

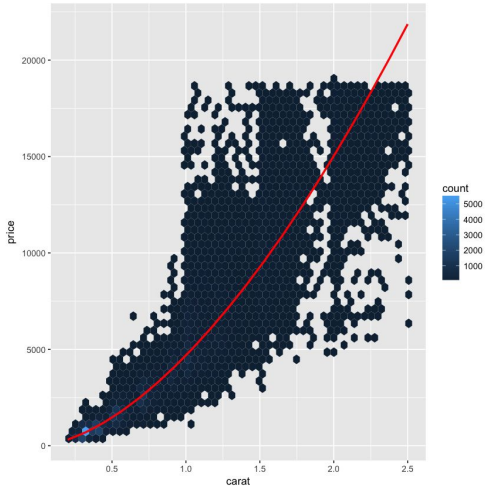
Model building
for exploration

You can use models to partition your dataset into patterns and residuals.

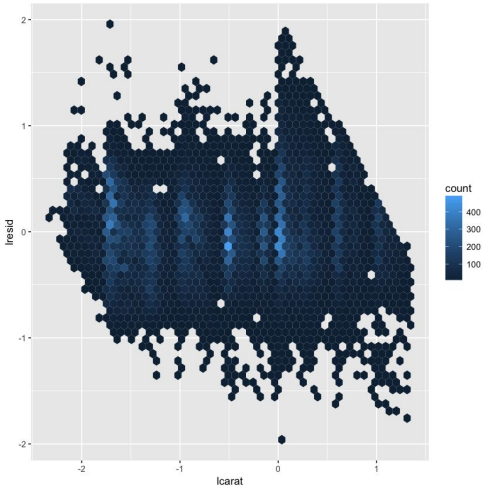
Data (Transformed)



Model



Residuals



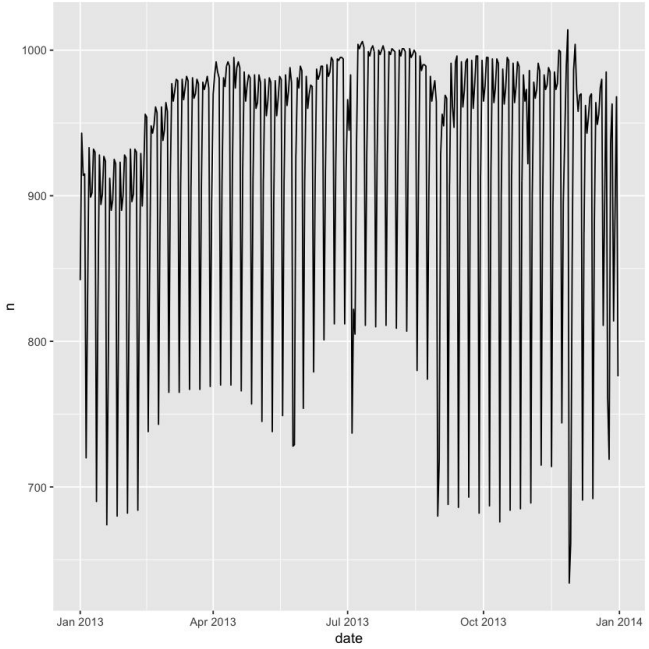
Model building:
flights example

Dataset: library(nycflights13)

flights by day (raw)

date	n
2013-01-01	842
2013-01-02	943
2013-01-03	914
2013-01-04	915
2013-01-05	720
2013-01-06	832
2013-01-07	933

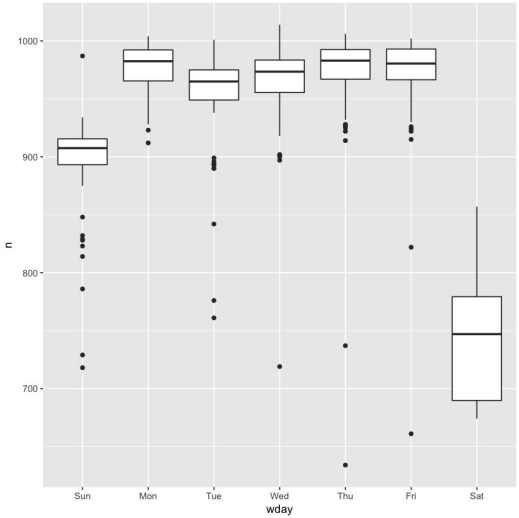
flights by day (visualized)



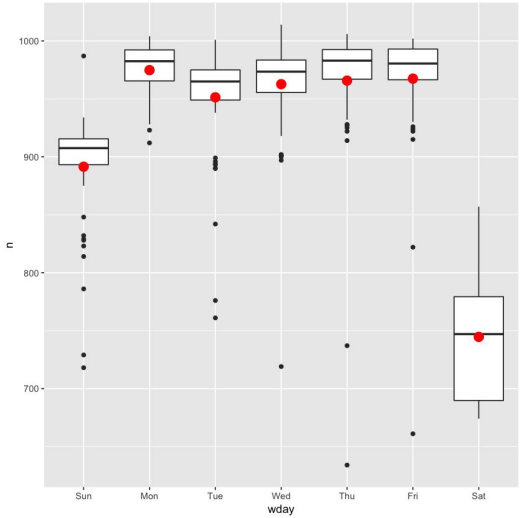
Model building:
flights example (cont.)

Let's look at total flights by day of week:

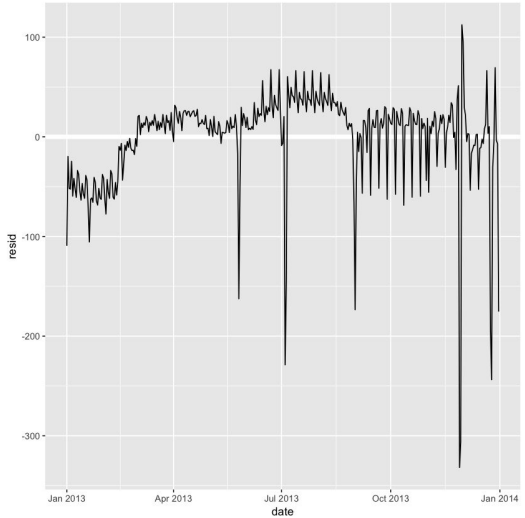
Data



Model



Residuals



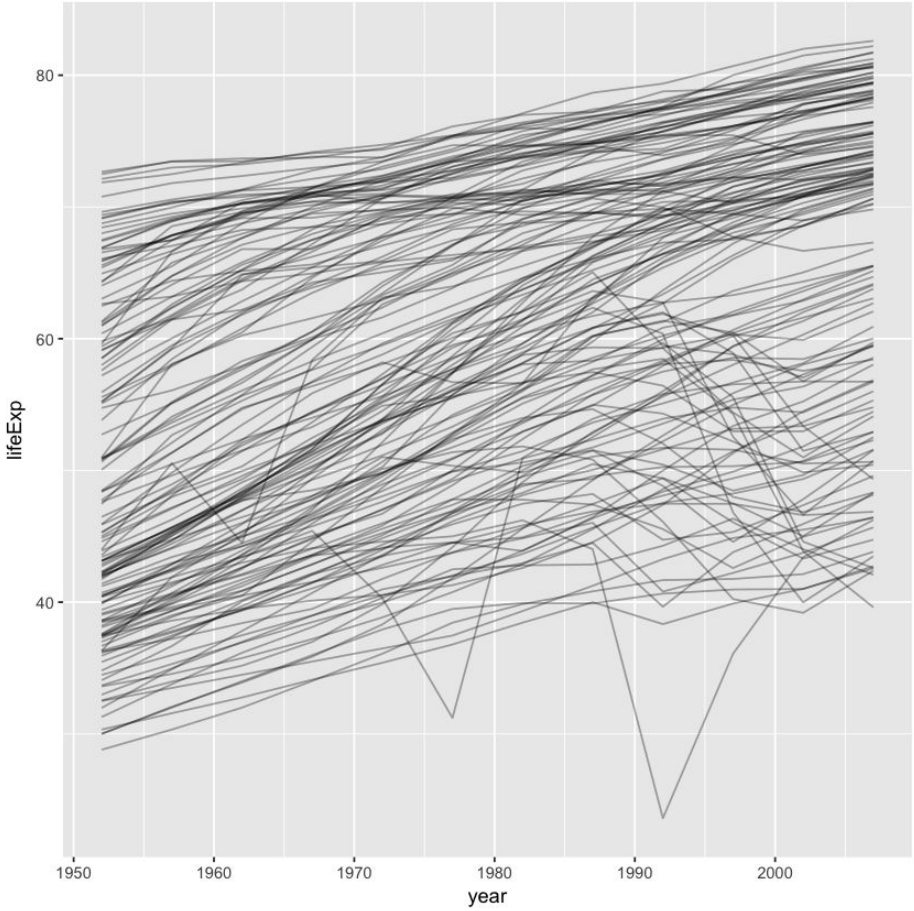
4.
Many Models

Gapminder dataset

How does life expectancy change over time for each country?

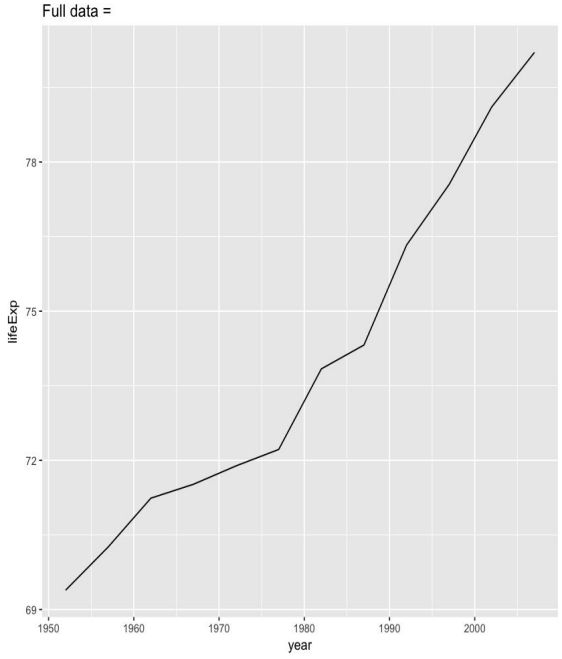
```
library(gapminder)
```

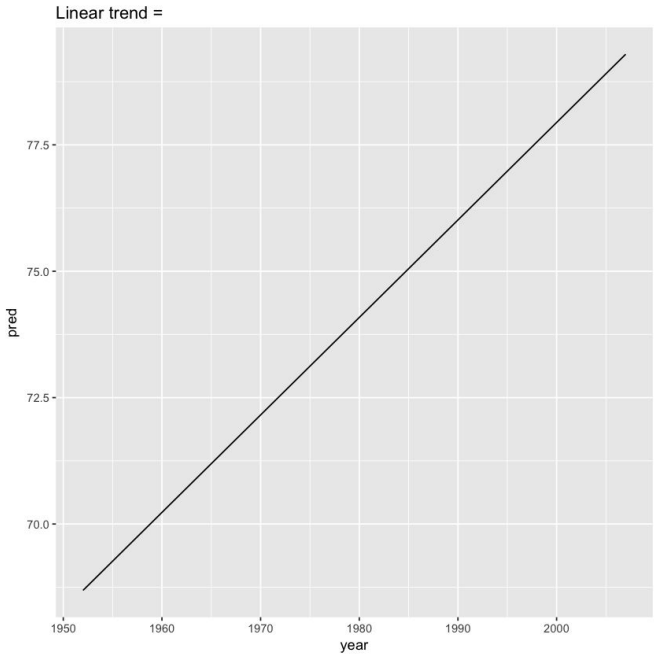
country	continent	year	lifeExp	pop	gdpPercap
Afghanistan	Asia	1952	28.801	8425333	779.4453
Afghanistan	Asia	1957	30.332	9240934	820.8530
Afghanistan	Asia	1962	31.997	10267083	853.1007
Afghanistan	Asia	1967	34.020	11537966	836.1971
Afghanistan	Asia	1972	36.088	13079460	739.9811
Afghanistan	Asia	1977	38.438	14880372	786.1134



Gapminder dataset

Approach for a single country is below. How do we do many countries?





Nested dataframes

Nested dataframes contain one row per group, and a column, data, which is a list of data frames (tibbles, specifically).

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

country	continent	year	lifeExp	pop	gdpPercap
Afghanistan	Asia	1952	28.801	8425333	779.4453
Afghanistan	Asia	1957	30.332	9240934	820.8530
Afghanistan	Asia	1962	31.997	10267083	853.1007
Afghanistan	Asia	1967	34.020	11537966	836.1971
Afghanistan	Asia	1972	36.088	13079460	739.9811
Afghanistan	Asia	1977	38.438	14880372	786.1134

country	continent	data
Afghanistan	Asia	1.952000e+03, 1.957000e+03, 1.962000e+03, 1.967000e+03, 1.972000e+03, 1.977000e+03, 1.982000e+03, 1.987000e+03, 1.992000e+03, 1.997000e+03, 2.002000e+03, 2.007000e+03, 2.880100e+01, 3.033200e+01, 3.199700e+01, 3.402000e+01, 3.608800e+01, 3.843800e+01, 3.985400e+01, 4.082200e+01, 4.167400e+01, 4.176300e+01, 4.212900e+01, 4.382800e+01, 8.425333e+06, 9.240934e+06, 1.026708e+07, 1.153797e+07, 1.307946e+07, 1.488037e+07, 1.288182e+07, 1.386796e+07, 1.631792e+07, 2.222742e+07, 2.526840e+07, 3.188992e+07, 7.794453e+02, 8.208530e+02, 8.531007e+02, 8.361971e+02, 7.399811e+02, 7.861134e+02, 9.780114e+02, 8.523959e+02, 6.493414e+02, 6.353414e+02, 7.267341e+02, 9.745803e+02
Albania	Europe	1952.000, 1957.000, 1962.000, 1967.000, 1972.000, 1977.000, 1982.000, 1987.000, 1992.000, 1997.000, 2002.000, 2007.000, 55.230, 59.280, 64.820, 66.220, 67.690, 68.930, 70.420, 72.000, 71.581, 72.950, 75.651, 76.423, 1282697.000, 1476505.000, 1728137.000, 1984060.000, 2263554.000, 2509048.000, 2780097.000, 3075321.000, 3326498.000, 3428038.000, 3508512.000, 3600523.000, 1601.056, 1942.284, 2312.889, 2760.197, 3313.422, 3533.004, 3630.881, 3738.933, 2497.438, 3193.055, 4604.212, 5937.030

Notes on
nested dataframes

In a grouped data frame, each row is an observation;
In a nested dataframe, each group is a row.

In a nested dataframe, we have a meta-observation:
a row that represents the complete time-course for a country,
Rather than a single point in time.

To look at a single element from the data column:

```
by_country$data[[1]]
```

year	lifeExp	pop	gdpPercap
1952	28.801	8425333	779.4453
1957	30.332	9240934	820.8530
1962	31.997	10267083	853.1007
1967	34.020	11537966	836.1971
1972	36.088	13079460	739.9811
1977	38.438	14880372	786.1134

Nested dataframes + purrr::map()

We have a model fitting function:

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

… and we want to apply it to every country. Since the country-level dataframes are in a list, we can use `purrr::map()` to apply to each one:

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

Nested dataframes + purrr::map() (continued)

Rather than saving models separately, we should add them to our existing dataframe as a new variable.

```
by_country <- by_country %>%
    mutate(model = map(data, country_model))
```

Using this approach, the models stay with the data and we can filter or arrange as needed.

Unnesting

Use `unnest()` to turn a nested dataframe into a regular dataframe.

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

```
resids <- unnest(by_country, resids)
resids
#> # A tibble: 1,704 × 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
```

Model quality +
broom()

broom package provides functions to turn models into tidy data:

- `tidy()`: constructs a dataframe which summarizes a model's statistical findings (including coefficients and p-values for each regression term)
- `augment()`: adds columns to the original data modeled
- `glance()`: constructs a one-row summary of the model (including R-squared and other values calculated at the model level)

Model quality:
broom example

```
broom::glance(nz_mod)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residual
0.9535846	0.9489431	0.8043472	205.4459	5.407324e-08	2	-13.32064	32.64128	34.096	6.469743	10

```
glance <- by_country %>%
    mutate(glance = map(model, broom::glance)) %>%
    unnest(glance, .drop = TRUE)
```

country	continent	r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance
Afghanistan	Asia	0.9477123	0.94248348	1.2227880	181.2494098	9.835213e-08	2	-18.3469348	42.693870	44.148590	14.9521045
Albania	Europe	0.9105778	0.90163554	1.9830615	101.8290138	1.462763e-06	2	-24.1490356	54.298071	55.752791	39.3253302
Algeria	Africa	0.9851172	0.98362893	1.3230064	661.9170864	1.808143e-10	2	-19.2922136	44.584427	46.039147	17.5034589

R-Ladies Austin

Upcoming Events

Data Day Texas Events [Jan 26 + Jan 27]

Book Club: Weapons of Math Destruction [January 31]

R4DS: Communicating Results with rmarkdown and ggplot2 [February 21]

From Zero to Web App with Shiny! [March 28]