Review Session 1

Our R Regression Refresher

Ben Berger

1/27/23

Introduction

Ben Berger

- Public Policy Ph.D. G5
- Hometown: Harrisburg, PA
- Research Interests: Health care, innovation, orphan drugs, regulation
- Teaching experience: 2 sem. API202Z; R experience: 8 years
- Hobbies: **Spending time with my puppy!** And if I still had a life outside of grad school and my puppy... photography, birding, board games, guitar, eating spicy food,

Ringo

Outline

- Fridays 10:30-11:45.
- Goals:
  - Review and expand on key concepts
  - Prepare for problem sets using practice problems
  - Develop R skills.
- Some fraction of the time on concepts and some on R, depending on where supply and demand intersect.

Expectations (for me)

- I will do my best to make 202Z interesting and informative.
  - Review session is an opportunity to engage with your interests.
- You can come to me with any questions you have about the course material, statistical methods, research more broadly, or graduate student life.
  - Best times are before/after class and during office hours, but also feel free to message me on Slack with any questions/comments/concerns.
  - I make myself as available as possible to answer questions (because I like helping out students!) but responses may be delayed if I'm busy with research or if you're messaging me after business hours.

Expectations (for me)

- Review sessions are **homework-focused**. I will cover material that you can directly apply to completing the homework assignment.
- I don't grade the homework and can't comment on why you got a particular grade, but would be happy to help you work through any confusion with the material.
- I will post review session materials in advance, but I encourage you to follow along with the lecture and write your own code for R exercises.

Expectations (for you)

- Show up on time.
  - Especially important because I try to make the materials interactive.
- Bring your computer to follow along with the interactive R exercises.
- Engage with the material.
  - You'll get the most out of review session if you and your peers are participating.
- Be open-minded.
  - Even if you're a stats-skeptic or stats-expert, we have a lot to learn from others.
- Ask questions if you have them.
  - It's always okay to be wrong! And this is the absolute best time to correct any misconceptions.

Today

- Discuss programming style and debugging tips.
- Discuss R model estimation and interpretation.
- Complete some interactive exercises.

On Programming Style

Your computer doesn't care about your programming style, but it helps to be consistent and follow the programming conventions.

Future users (including yourself!) will appreciate understandable code.

Which of these commands is easier to read?

```
# 1.
y<-2 ^ 2+3 *( 4 +2 )

# 2.
y <- 2^2 + 3 * (4 + 2)
```

On Programming Style

- Focus on *readability*.
- Best practices:
  - Follow the style guide.
  - If you diverge from the style guide, at least be consistent.
  - Comment your code clearly and extensively.
  - Write code that is understandable!

Some Problematic Code

- This script plots the opening price of Tesla stock on each day of December 2020.
- What are some (stylistic) things wrong with this code?

```
# Read daily Tesla stock price data.
muskData.df<-read.csv( "TSLA.csv"  )

# Keep only dates in December 2020.
library(tidyverse)
X_Æ_A_12<- filter(muskData.df,
Date>="2020-12-01" & Date <= '2020-12-31')

# Plot
ggplot(X_Æ_A_12,aes(x=Date,y=Open)) +
  geom_col()
```

Some Problematic Code

- The object names are confusing and use non-standard characters. Avoid this where possible.
- `library(tidyverse)` isn't at the top of the script.
- Inconsistent use of single and double quotes.
- Inconsistent spacing & doesn't follow style guide.
- Comments are mostly okay, but doesn't explain that it plots opening prices.

Some Better Code

```
library(tidyverse)
library(lubridate)

# Read daily Tesla stock price data.
tesla_price <- read_csv("TSLA.csv")

# Keep only dates in December 2020.
tesla_price_dec2020 <- tesla_price %>%
  filter(year(Date) == 2020 & month(Date) == 12)

# Plot opening prices on each day.
ggplot(tesla_price_dec2020,
       aes(x = Date, y = Open)) +
  geom_col()
```

Debugging — Feedback

R will give you 3 types of feedback:

- Messages: generally harmless and are used to give some information about the code you just ran.
- Warnings: indicate that the code ran successfully but there is something unexpected that R is trying to flag to you.
  - For example, if you make a plot and some outlier points are excluded due to the limits you specified for the axes, R will supply a warning to remind you of this. If that is what you intended to do, you can safely ignore the warning.
- Errors: indicate that the code did not run successfully.

Debugging — Techniques

- Break your code up into smaller chunks.
  - You can run highlighted chunks of code rather than full lines.
- Read the error message and try to decipher it (Google is your friend!)
- Try clearing your environment and running your code from the beginning, in case you have accidentally overwritten some object.
- Ask us! Post a question on Slack or come to Office Hours.

Debugging – Common Error Messages

Message	**Possible Causes**
Error: could not find function	Did you load the package that contains this function? Did you spell the function name correctly?
Error: object not found	Did you spell the name of the object correctly?
Error : non-numeric argument to a binary operator	Did you try to mix different object types, like numeric and character?
Error: unexpected symbol in...	Did you forget some punctuation, like a comma between arguments in a function?
Error: ... unexpected end of input ...	R thinks your code is unfinished. Did you forget a closing parenthesis?
+	If you just see a "+" sign in the console, this indicates that R is expecting more code. The previous statement was unfinished.

File paths

When you want to load a dataset into R that is saved on your computer, you need to tell R where the file is located (*path*).

Example:
`/Users/ben/Teaching/api202z_2023/review_sessions/review_1/climate_pa`

The above path is called an *absolute path* because it gives the exact address of the file on my computer. However, it's long and won't be useful at all on someone else's computer because the file will almost certainly be saved in a different location.

Instead, you usually want to use a *relative path*. This gives the location of a file relative to the current *working directory*. For example, if my working directory is `/Users/ben/Teaching/api202z_2023/review_sessions/review_1/`, then I can specify a file in that directory using only its name, e.g. `climate_panel.dta`.

Working Directory

The working directory is the path from which R looks for files on your computer. You can check the current working directory using `getwd()` and set a new working directory using `setwd(YOURPATH)`.

Or you can make your life easy by using `.Rproj` (R Project) files.

How?

- Create a folder for your project (e.g. `review_1/` or `problem_set_1/`).
- In RStudio, navigate to File > New Project. Then choose "Existing Directory" and choose the folder you just created.

What does this do?

The new instance of RStudio now has its working directory set by default to the folder you just created. If you close RStudio and reopen it by double clicking the new `.Rproj` file, the working directory will still be correct.

Let's Get Started

Download the entire Review 1 folder from Canvas and open the file `review_1.Rproj`. This should load up a new RStudio instance with the correct working directory.

I encourage creating an R Project file for each assignment because it ensures that your R session will always open in the correct working directory, making it simple to refer to data files in that directory as soon as you boot up RStudio.

Now create a new script using Cmd+Shift+N (Mac) or Ctrl+Shift+N (Windows). Use this script to do the exercises.

Navigating Rstudio

- Source (Ctrl+1)
  - Edit scripts.
  - View datasets.
- Console (Ctrl+2)
  - Code output appears here.
  - Can also enter code here, but write your programs in a script!
  - Use console to `install.packages()`.
- Help (Ctrl+3)
  - Documentation on packages, functions, objects, etc.
- Environment (Ctrl+8)
  - Objects (e.g. datasets) you have loaded into R.
  - Click on dataset objects to view them.

Navigating RStudio

It's annoying to frequently need to move the mouse when you're programming. Using some of these shortcuts may increase your productivity.

Table 1: Shortcuts

Command	Mac	Windows
Change focus to Source	Ctrl+1	Ctrl+1
Change focus to Console	Ctrl+2	Ctrl+2
Change focus to Help	Ctrl+3	Ctrl+3
Change focus to Environment	Ctrl+8	Ctrl+8
Run current line	Cmd+Enter	Ctrl+Enter
Run highlighted code chunk	Cmd+Enter	Ctrl+Enter
Run entire script	Cmd+Shift+Enter	Ctrl+Shift+Enter
Pipe Operator %>%	Cmd+Shift+M	Ctrl+Shift+M

For more shortcuts, go to Tools -> Keyboard Shortcuts Help.

Ultimate RStudio Hack

- Multi-line editing: hold down control + option (control + alt on PC) then press the up or down key to repeat your cursor on multiple lines.

Simple (Bivariate) Linear Regression

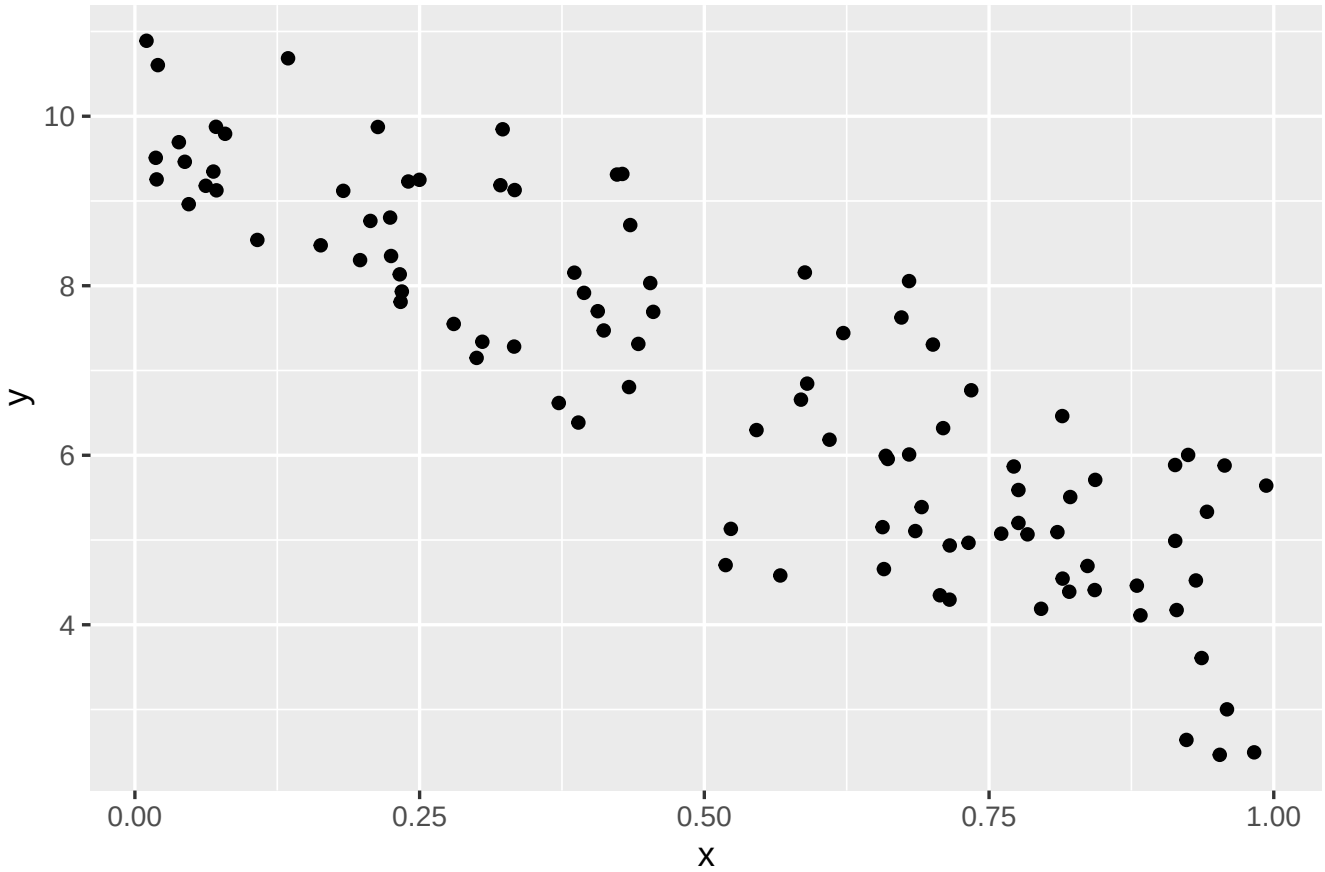
$$Y_i = \beta_0 + \beta_1 X_i + u_i \tag{1}$$

- We refer to this as the *regression of Y on X*.
- β_1 captures the average linear relationship between X and Y.
- Whether or not there is a causal relationship between X and Y, we can make the descriptive statement: *a one unit increase in X is on average associated with a* β_1 change in Y.
- In R, estimate and store a bivariate model (via OLS) with `bivariate <- lm(y ~ x, data)`.

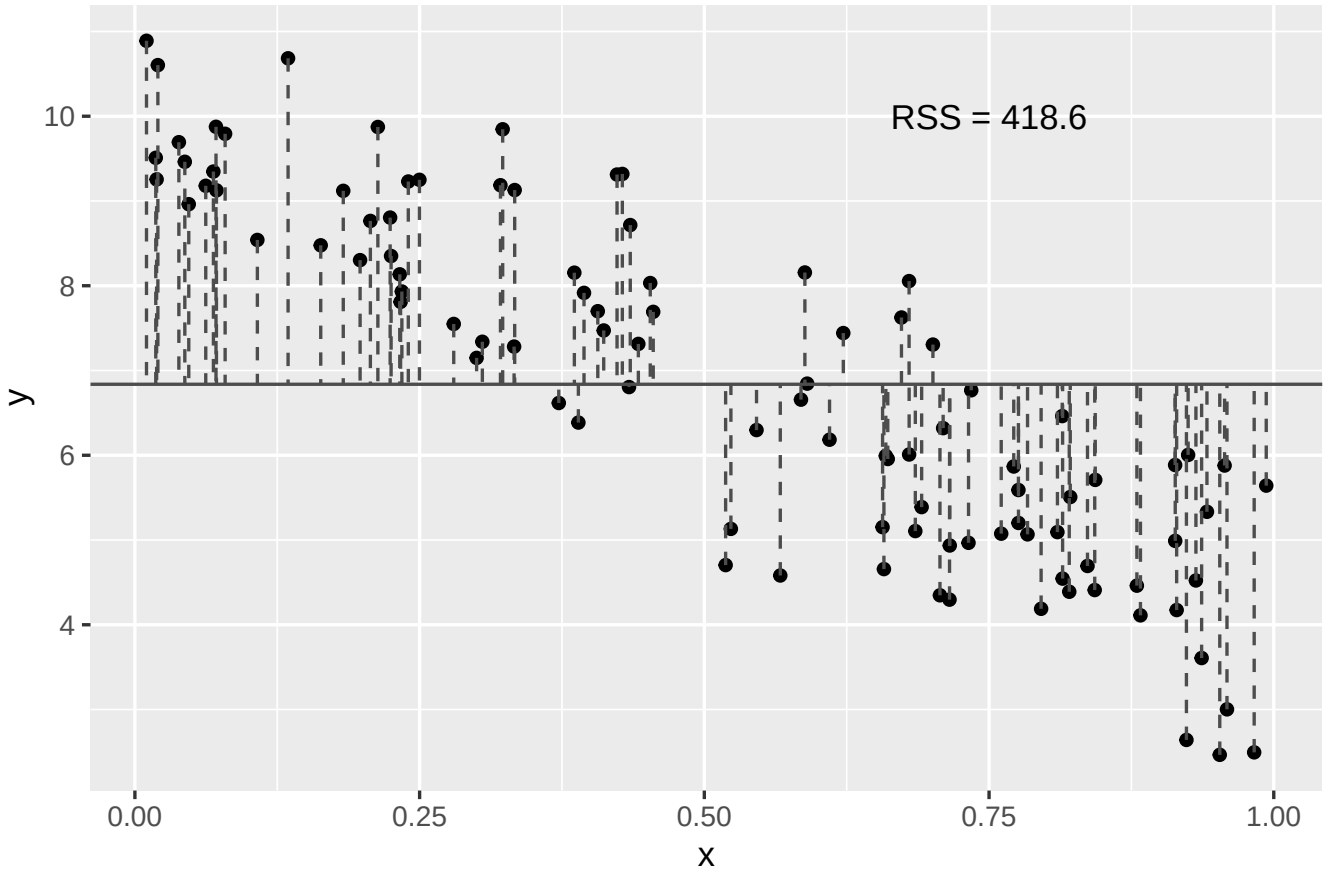
Estimation by OLS

- No hats (e.g. β_1) denote *estimands*, the population parameters we are trying to estimate.
- Hats (e.g. $\hat{\beta}_1$) denote *estimators*, quantities computed from our sample of data to estimate the population parameter.
- Ordinary Least Squares (OLS) is the method most commonly used to estimate linear regression models.
- We want predictions to be close to true values, therefore, OLS chooses values for $\hat{\beta}$ that minimize the sum of squared deviations from predicted values (residuals).
- $\hat{\beta}^{OLS}$ is the $\hat{\beta}$ that minimizes $RSS = \sum_{i=1}^{n}(Y_i - \hat{\beta}_0 + \hat{\beta}_1 X_i)^2$
- Solving for the OLS estimate, we obtain $\hat{\beta}_1^{OLS} = \frac{Cov(Y_i, X_i)}{Var(X_i)}$.

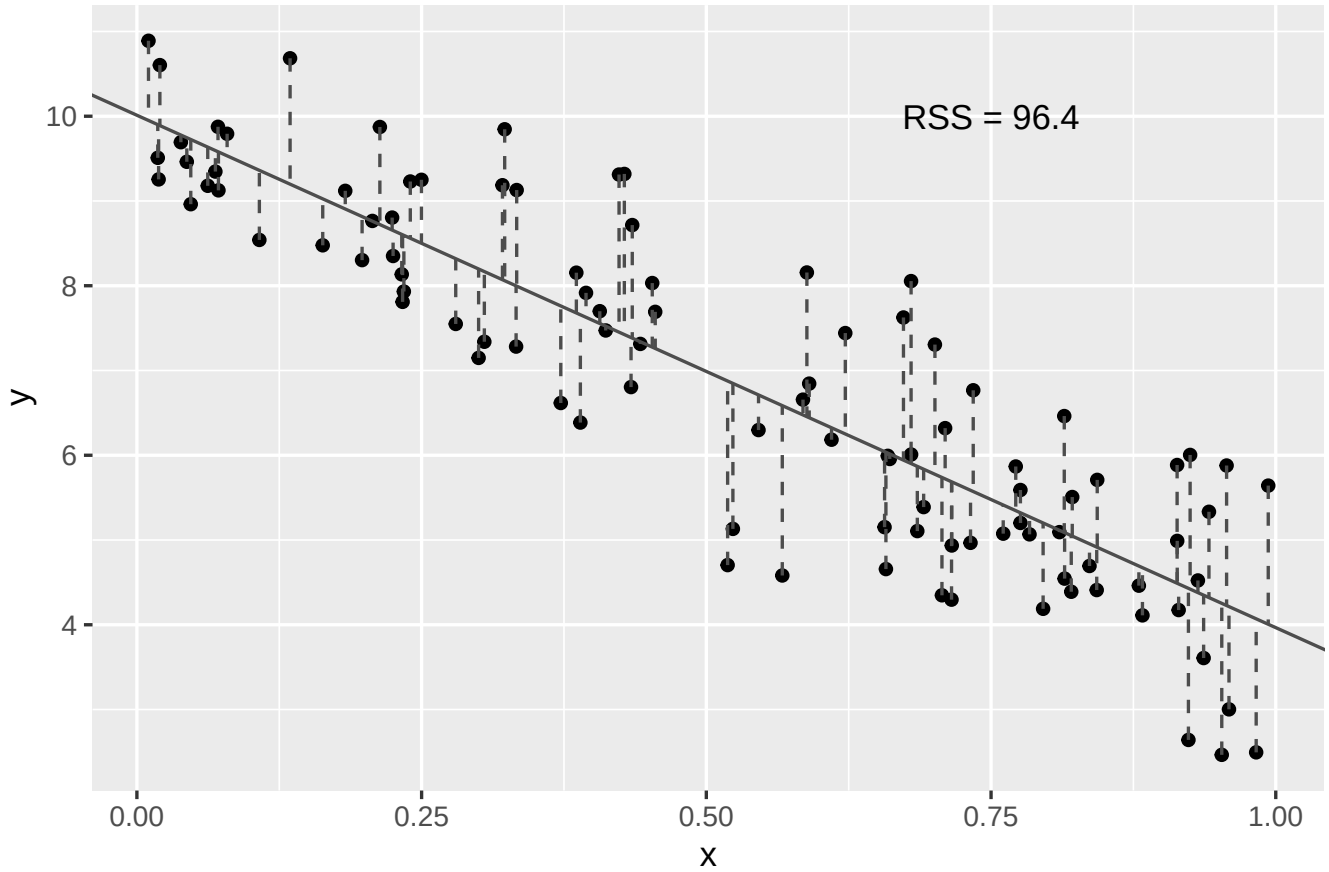
Estimation by OLS



Estimation by OLS



Estimation by OLS



R Exercise: Introduction

Suppose we want to study if global average rainfall has changed over time. We have access to the dataset `climate_panel.dta`, which includes two relevant variables:

- `wpre` – countries' average precipitation (in 100s of mm), weighted by landmass
- `wtem` – countries' average temperature (in Celcius), seemingly not weighted by landmass?
- `year` – the year the observation is from

What we will do today: load and manipulate this data, run and interpret bivariate and multivariate regressions, and graph our results.

R Exercise: Setup

First load the data as below:

```
# Load packages
library(tidyverse) # for data manipulation, graphing, etc.
library(haven) # to import a .dta file

# Load data
climate_data <- read_dta("climate_panel.dta")
```

Now use `climate_data` to create a new dataset of global average **precipitation** and global average **temperature** in each year of the data. So, each row should comprise a unique year and the associated global rainfall.

Hint: you'll want to use the `group_by()`, `summarize()`, and `mean()` functions.

R Exercise: Model Estimation

```
# Compute new dataset of annual rainfall
annual_rain <- climate_data %>%
  group_by(year) %>%
  summarize(mean_rain = mean(wpre),
            mean_temp = mean(wtem))
```

Now estimate the bivariate regression of global average rain on year.

```
# Regress average rainfall on year, save result to object
bivariate <- lm(mean_rain ~ year, data = annual_rain)
```

- I ran this code and there was no output. Why?
- What would happen if I changed `bivariate` to `model_bivariate`?
- What if I changed `mean_rain` to `mean_temp`?

R Exercise: Model Summary

```
summary(bivariate)
```

```
Call:
lm(formula = mean_rain ~ year, data = annual_rain)

Residuals:
     Min       1Q   Median       3Q      Max
-1.04564 -0.21571 -0.00875  0.26884  1.29593

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept) 106.926622   6.299224   16.98   <2e-16 ***
year         -0.047979   0.003185  -15.07   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3955 on 55 degrees of freedom
Multiple R-squared:  0.805, Adjusted R-squared:  0.8014
F-statistic:   227 on 1 and 55 DF,  p-value: < 2.2e-16
```

R Exercise: Interpreting Regression Output

$$\text{Rainfall}_i = \beta_0 + \beta_1 \text{Year}_i + u_i$$

- Based on the regression output, what are $\hat{\beta}_0$ and $\hat{\beta}_1$?
  $\hat{\beta}_0$ is 106.92 and $\hat{\beta}_1$ is -0.048.
- What is the SRF?
  $\widehat{Rainfall}_i = 106.92 - 0.048 Year_i$
  or
  $Rainfall_i = 106.92 - 0.048 Year_i + \hat{u}_i$
- Interpret $\hat{\beta}_1$. Recall that $Rainfall_i$ is measured in 100s of millimeters.
  Each additional year is associated with a 4.8mm decline in rainfall.

R Exercise: Model Details

```
# Extract coefficients from model
bivariate$coefficients
```

```
 (Intercept)          year
106.92662244  -0.04797856
```

```
# Extract a specific coefficient
bivariate$coefficients["year"]
```

```
       year
-0.04797856
```

```
# Standard errors are a bit trickier
sqrt(diag(vcov(bivariate)))
```

```
(Intercept)          year
6.299224423 0.003184533
```

R Exercise: Model Details

```
# Extract residuals and fitted values
bivariate$residuals
bivariate$fitted.values
```

```
# Predict values of the dependent variable
predict(bivariate, tibble(year = 2023))
```

```
       1
9.865999
```

How could you calculate the sum of squared residuals (RSS)?

```
sum(bivariate$residuals^2)
```

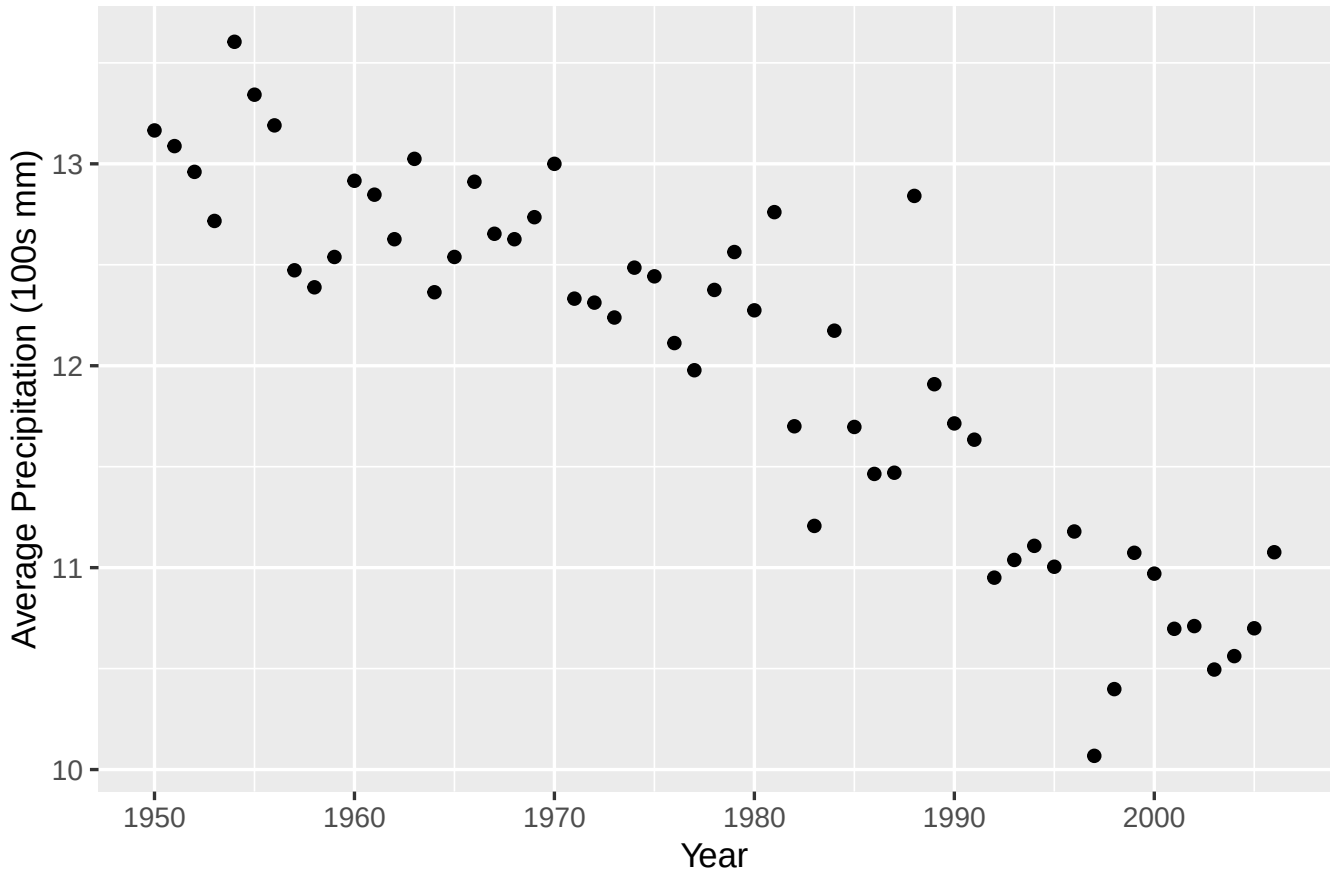
```
[1] 8.605257
```

R Exercise: Plotting

```
# Save a scatterplot as an object
rain_scatter <- ggplot(annual_rain,
                       aes(x = year, y = mean_rain)) +
  geom_point() +
  labs(x = "Year",
       y = "Average Precipitation (100s mm)")
```

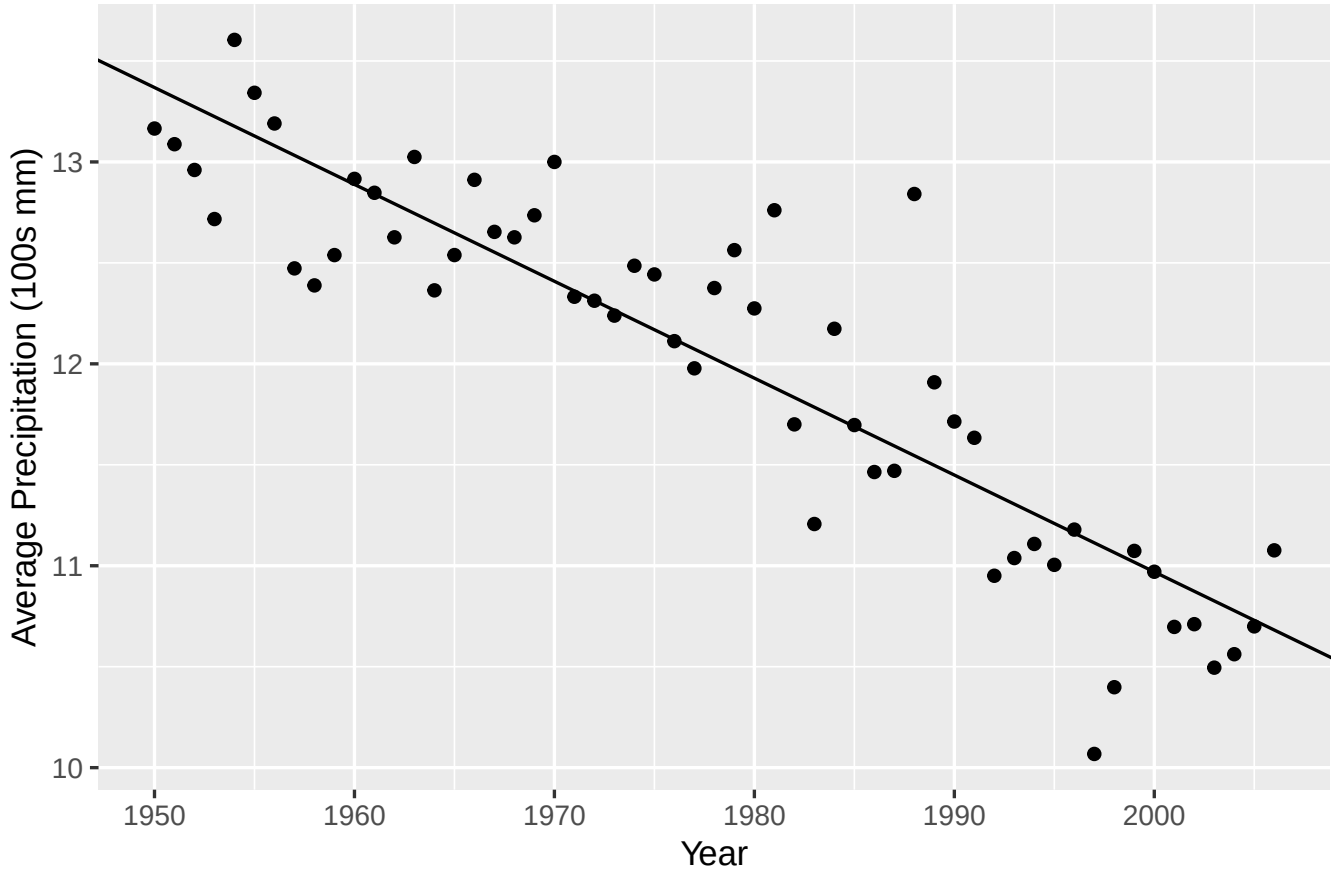
R Exercise: Plotting

```
rain_scatter
```



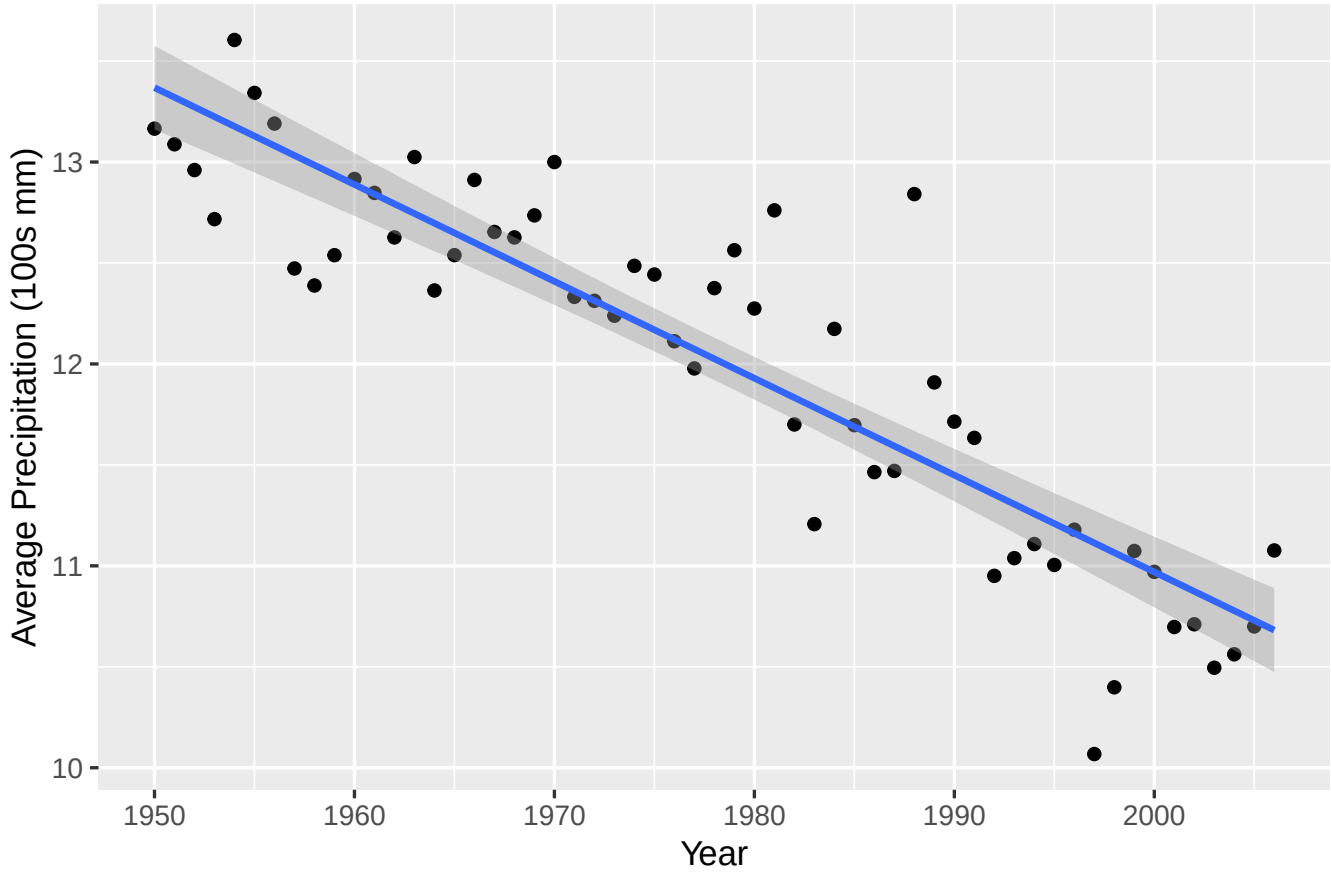
R Exercise: Plotting

```
# Add regression line by supplying coefficients
rain_scatter +
  geom_abline(intercept = bivariate$coefficients[1],
              slope = bivariate$coefficients[2])
```



R Exercise: Plotting

```
# Add regression line using ggplot geometry
rain_scatter + geom_smooth(method = "lm")
```



R Exercise: Prediction

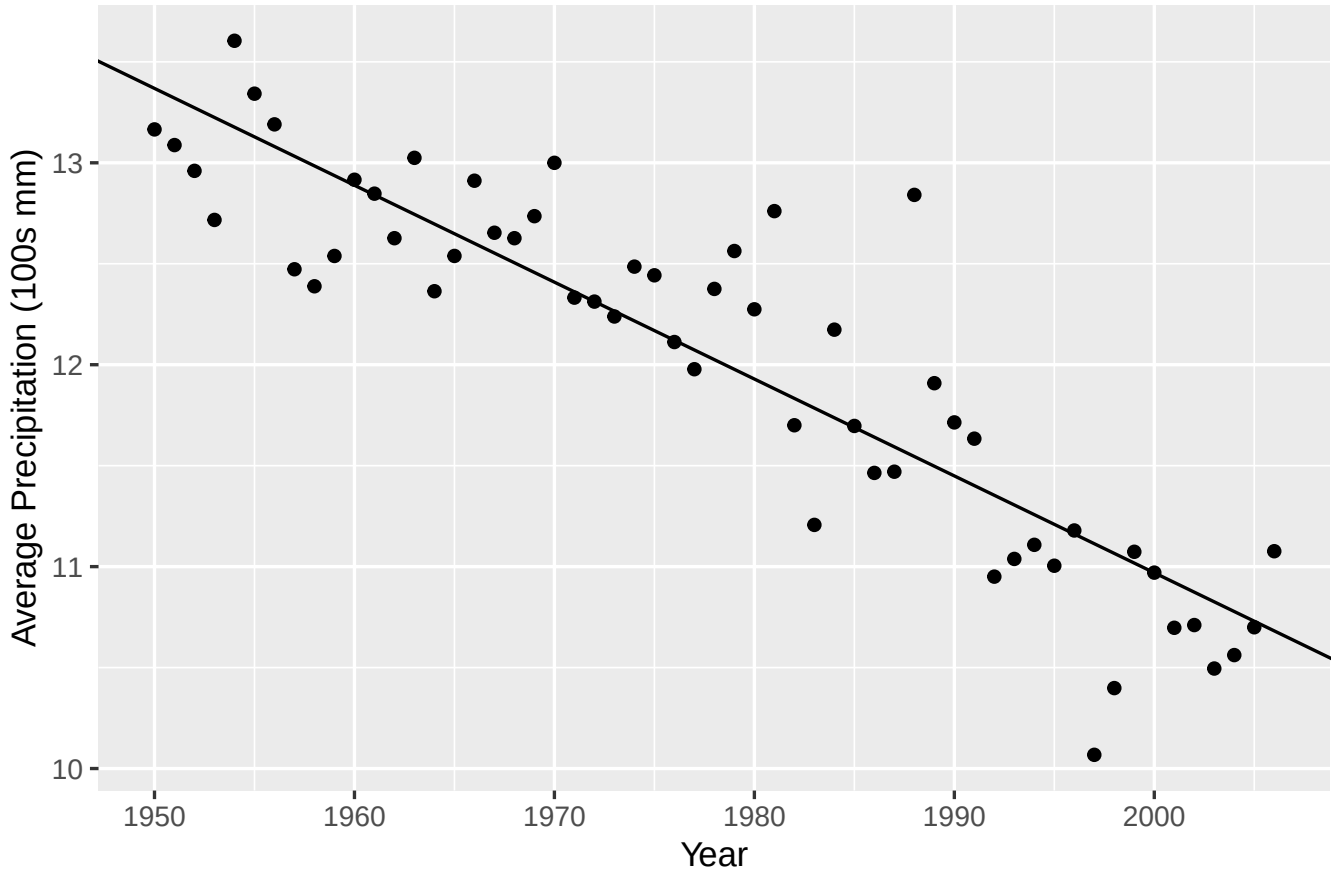
What does this linear regression predict rainfall was in 2022? How about in the year 1022? What about 3022? What do you think about the validity of those predictions?

```
predict(bivariate, newdata = tibble(year = c(2022, 1022, 3022)))
```

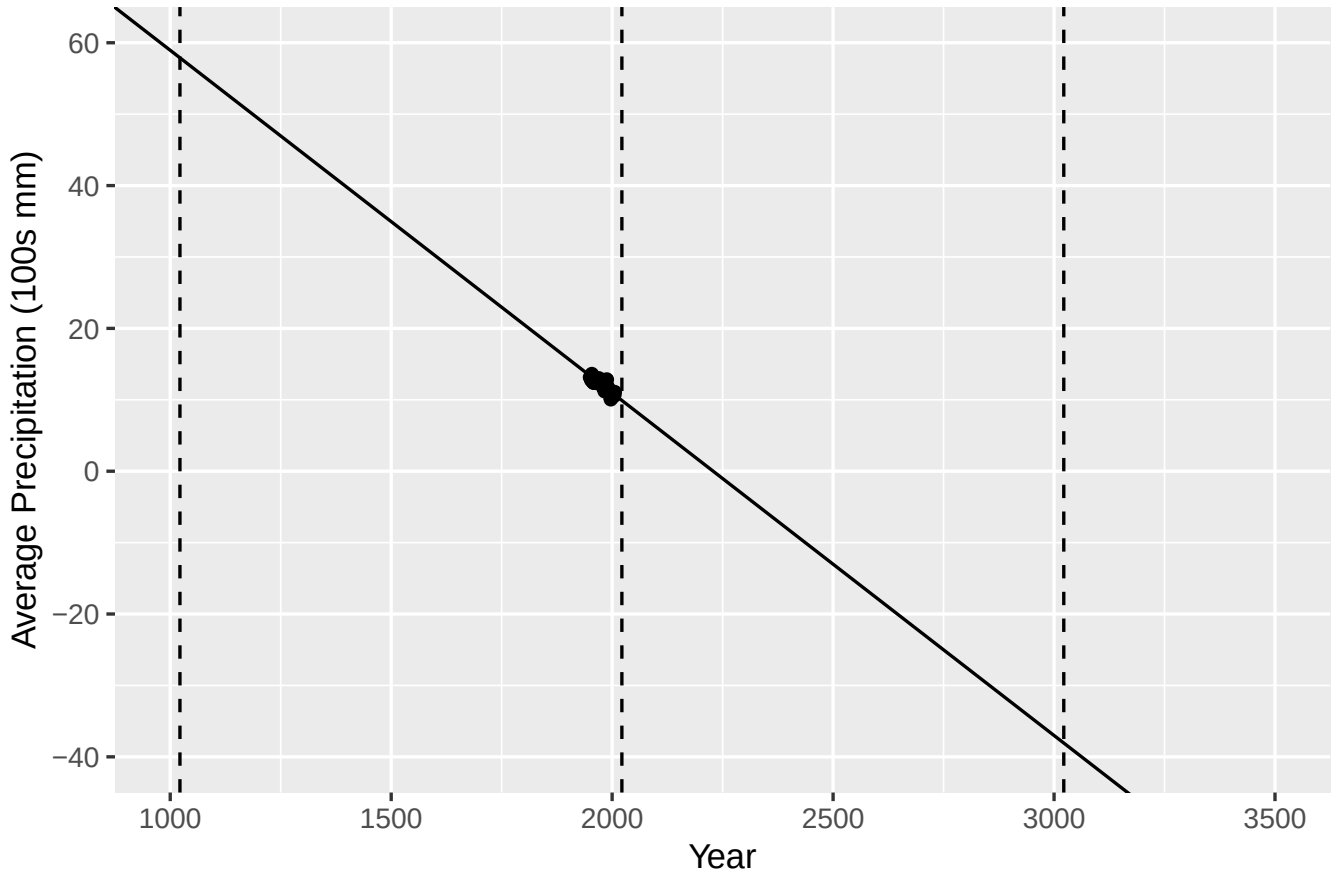
```
        1         2         3
9.913977 57.892536 -38.064581
```

- 991 mm of rain in 2022.
- 5,789 mm of rain in 1022.
- -3806 mm of rain in 3022.

R Exercise: Prediction



R Exercise: Prediction



Multiple Linear Regression

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + u_i \tag{2}$$

- We refer to this as the *regression of Y on* X_1 and X_2. More generally we can regress Y on K independent variables $X_1 ... X_K$.
- β_1 captures the average linear relationship between X_1 and Y, holding X_2 constant, or equivalently, controlling for X_2.
- We can say that *a one unit increase in* X_1 is on average associated with a β_1 change in Y, holding X_2 constant.
- In R, estimate and store a multivariate model (via OLS) with `multivariate <- lm(y ~ x1 + x2, data)`.

Exercise: Multiple Linear Regression

You suspect that the decline in global precipitation is related to an increase in global temperature. Estimate the following multivariate regression of mean global precipitation on year and mean global temperature:

$$\text{Rainfall}_i = \beta_0 + \beta_1 \text{Year}_i + \beta_2 \text{Temperature}_i + u_i \tag{3}$$

```
multivariate <- lm(mean_rain ~ year + mean_temp,
                   data = annual_rain)
```

Exercise: Multiple Linear Regression

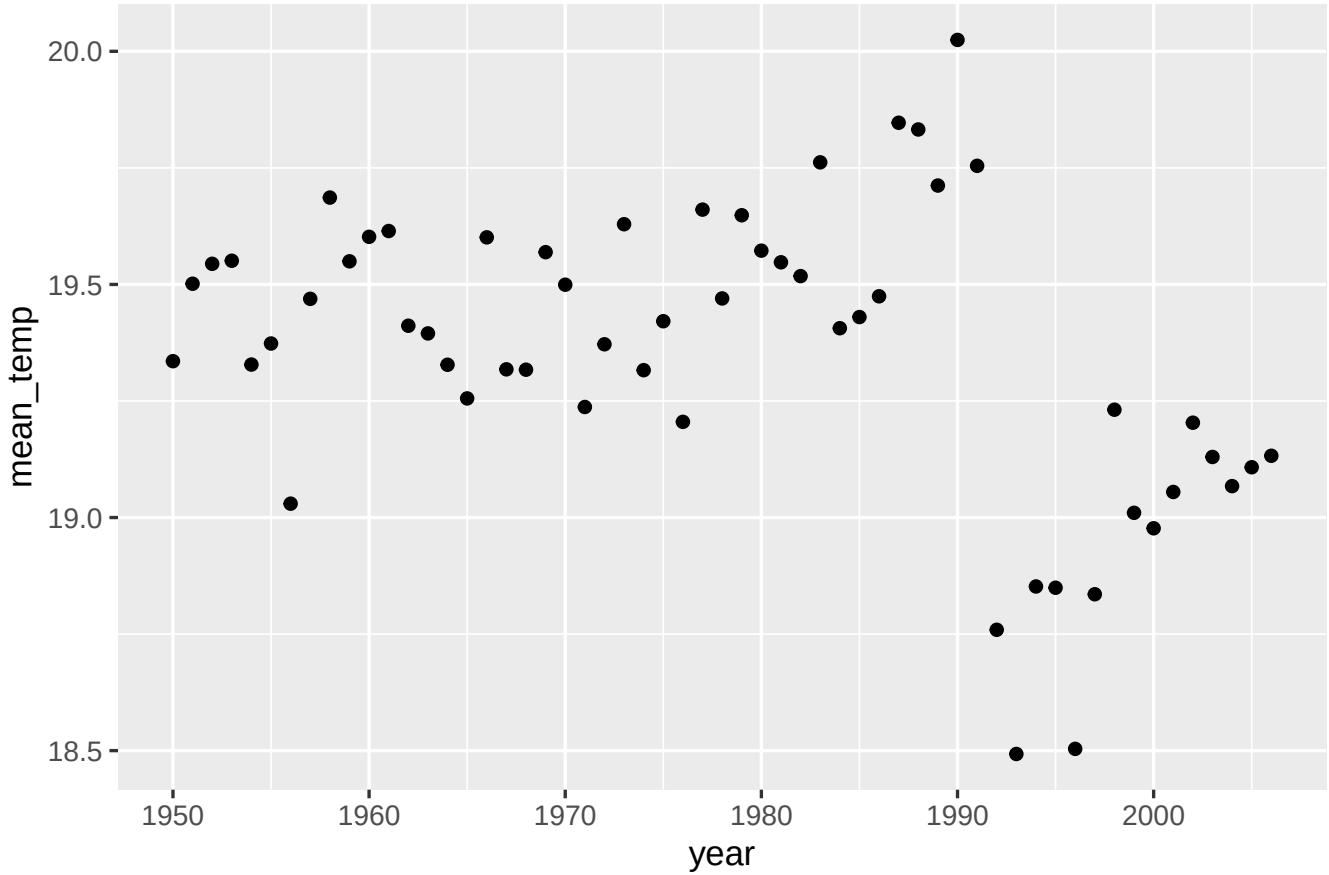
```
summary(multivariate)
```

```
Call:
lm(formula = mean_rain ~ year + mean_temp, data = annual_rain)

Residuals:
    Min      1Q  Median      3Q     Max
-0.8717 -0.2036 -0.0109  0.2400  1.0395

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 90.886604   8.411495  10.805 4.11e-15 ***
year        -0.044408   0.003292 -13.489  < 2e-16 ***
mean_temp    0.463838   0.171472   2.705  0.00912 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3746 on 54 degrees of freedom
Multiple R-squared:  0.8282,    Adjusted R-squared:  0.8219
F-statistic: 130.2 on 2 and 54 DF,  p-value: < 2.2e-16
```

Exercise: Multiple Linear Regression

Interpretation of $\hat{\beta}_1$?: An additional year is associated with a decline in rainfall of 4.4 mm, holding temperature constant.

This seemed weird to me, so I looked at the data on temperature more critically. Why is there a huge drop in temperature in 1992?



Exercise: Multiple Linear Regression

Moral of the story: know your data! Calculating average temperature only using countries that are observed in every year produces the following plot: