# RMarkdown Lesson Plan

Gus Pendleton

2024-02-06

## Set-up

Make new RMarkdown document. Discuss yaml header, code chunk, and prose

Save RMarkdown document in analysis folder

Confirm everyone is in Source mode

```
gapminder_df <- read.csv("data/gapminder.csv")

getwd()
```

```
## [1] "/Users/augustuspendleton/Desktop/Schmidt_Lab/Markdown_Lesson"
```

Discuss knitting directory - knitting chunks might have a different working directory than your r session!

Discuss chunk output in console vs. in document - look at yaml header

Assign read.csv to gapminder_df

## Adding new prose with markdown settings

### Headers

**Keep getting smaller**

**As we add hash tags** Point out how we can navigate document

We can make lists with asterisks or hyphens

List 1:

- Item 1
- Item 2
- Item 3

List 2:

- Other Item 1
- Other Item 2

We can make numbered list with numbers:

1. Numbered thing 1
2. Numbered thing 2

And we can make more complex lists:

1. Numbered thing 1
    - Sub list

    - Sub list
2. Numbered thing 2

We can make text **bold** or *italic*

We can incorporate greek symbols like $\mu$ or $\Delta$

We can make hyperlinks

Knit again

## Let's insert another R chunk

```
# Find the median population
median(gapminder_df$pop)
```

```
## [1] 7023596
```

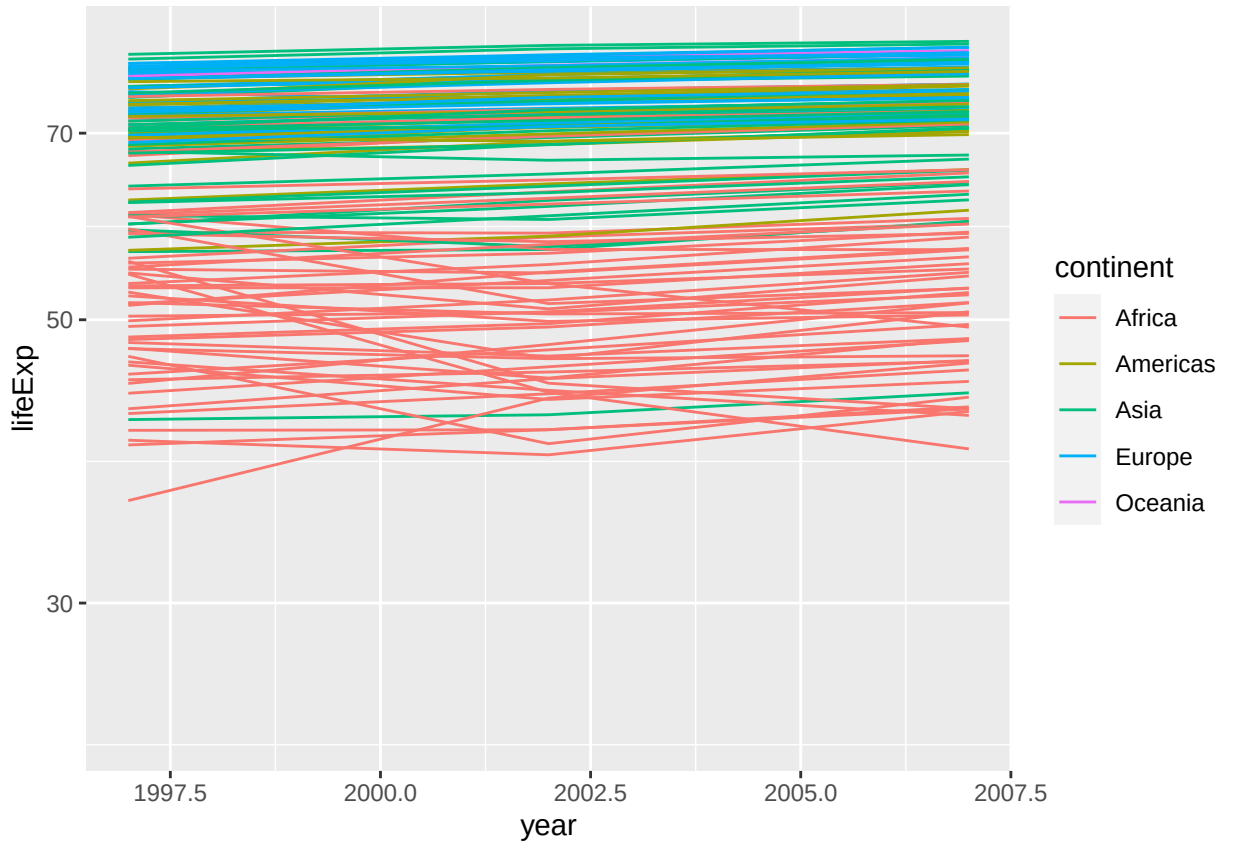
This is okay, but especially if people can't read understand our code, it's not that helpful. It would be better if we contextualized these numbers in writing.

Over 1704 observations, there was a median population of 0.7023596 million people.

## Let's make a plot

```
library(ggplot2)

ggplot(gapminder_df, aes(x = year, y = lifeExp, group = country, color = continent)) +
  geom_line() +
  scale_y_log10() +
  scale_x_continuous(limits = c(1997, 2007))
```




```
  #+ geom_bar()
```

Adding information to code chunks

First add echo = FALSE

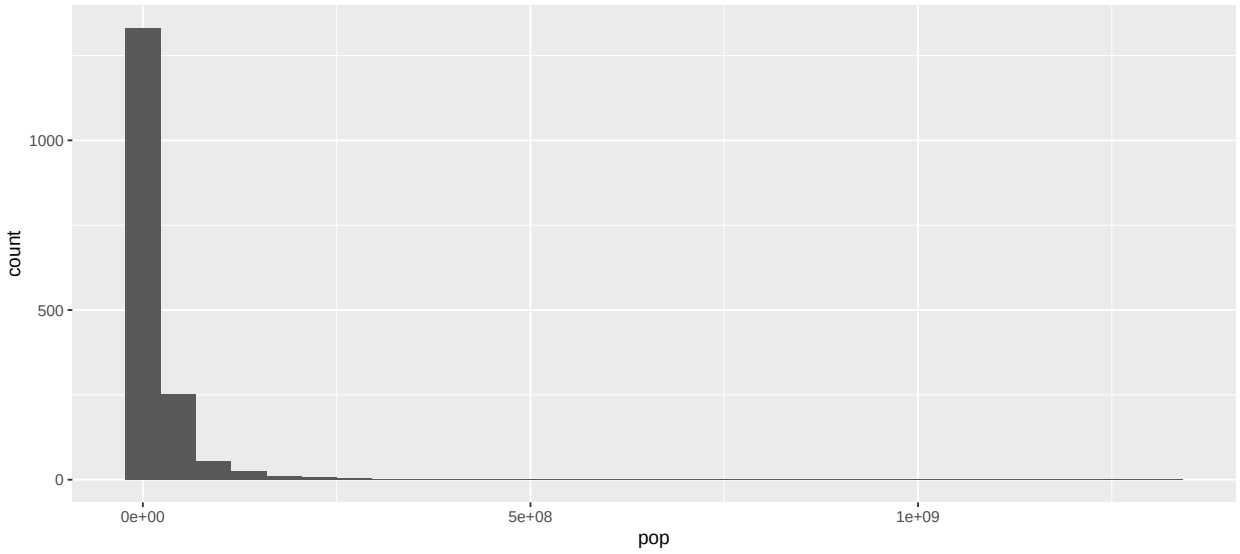
Next, add eval = FALSE

Next, add warning = FALSE

Next, simulate what happens when you have an error - can't knit

Make another type of plot

```
ggplot(gapminder_df, aes(x = pop)) +
  geom_histogram()
```




Next, add message = FALSE

Finally, change figure width

Add some new yaml code code_folding: hide toc: yes toc_float: yes

## Transition to git lesson

Why do we use version control?

Etherpad - how do you collaborate and share documents now?

How do you integrate figures and statistical analyses into writing?

Open up Github in your browser, log in

Open up terminal

Navigate to our lesson directory

git status - error (good!)

First, we're going to *configure* some of our Git settings

git config –global user.name "Gus Pendleton" git config –global user.email "guspendleton@gmail.com"

Email should be the same one you use for github!

git config –global core.editor = "nano -w"

git config –global init.defaultBranch main

## Okay, done setting up. Let's initialize a git repo

MAKE SURE YOU ARE IN THE PROJECT DIRECTORY git init

ls -a

Discuss .git folder - this is something we DONT TOUCH

git status

If necessary

git checkout -b main

Do not want to include git repos inside of git repos

We're about to start tracking our files. But sometimes, there's things we don't want to track: 1. Private information 2. Big files 3. Temporary or user-specific files

nano .gitignore .Rproj_user/ .DS_Store

Ready to add a change!

git add Markdown_Lesson.Rproj

git status

git add .gitignore

git commit -m "initialized repo"

git add data/

git commit -m "Added data file"

git add .

git commit -m "Added analysis folder"

Go back and add some lines to our markdown Now we're tracking with git!