Before the workshop

1. Download R :

      https://cloud.r-project.org/

2. Download Rstudio Desktop (Open Source License):

      https://www.rstudio.com/products/rstudio/download/

3. Access the workshop materials:
   - PowerPoint Presentation
   - Sample Code

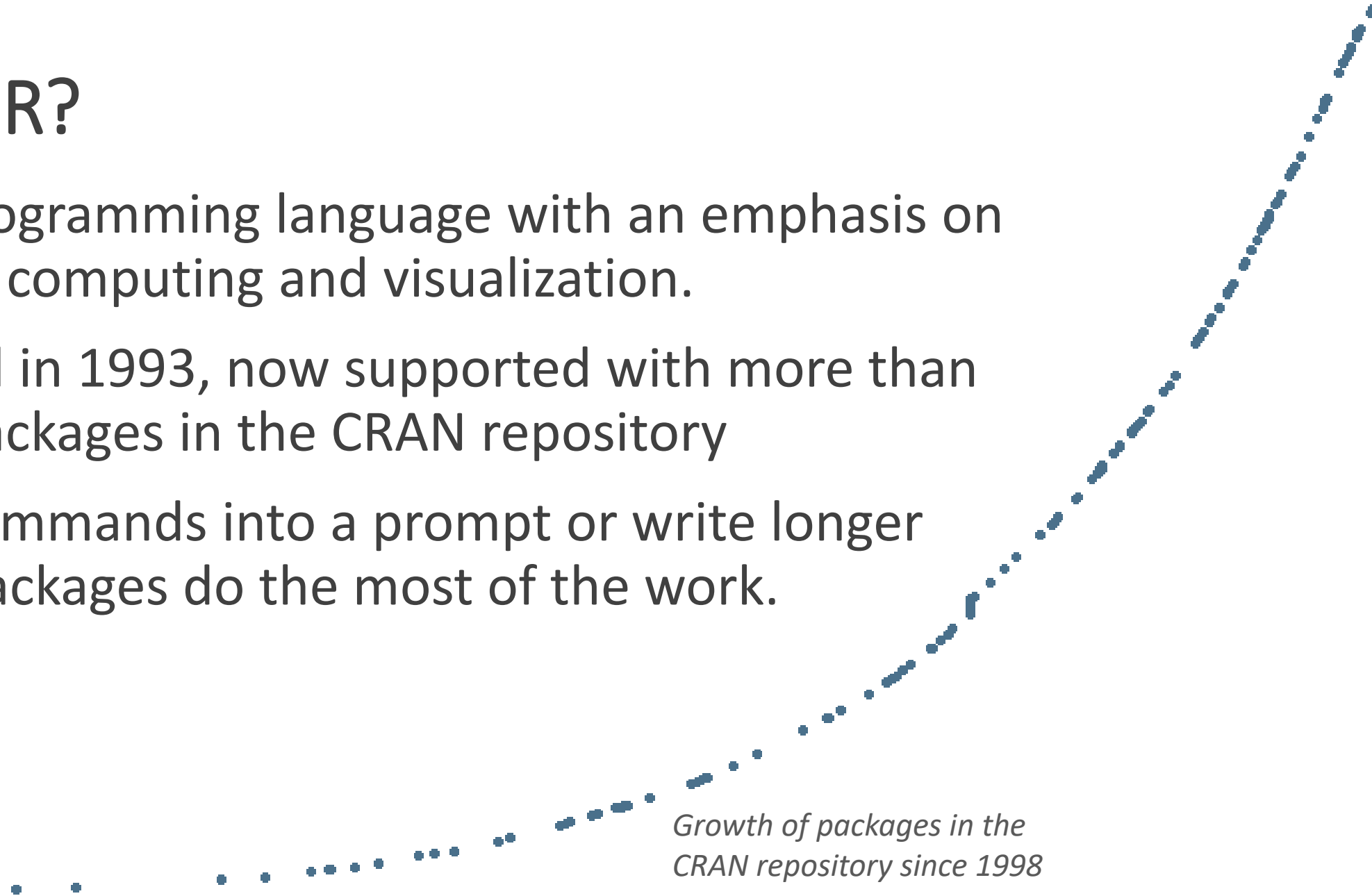MC LIBRARY

Introduction to R
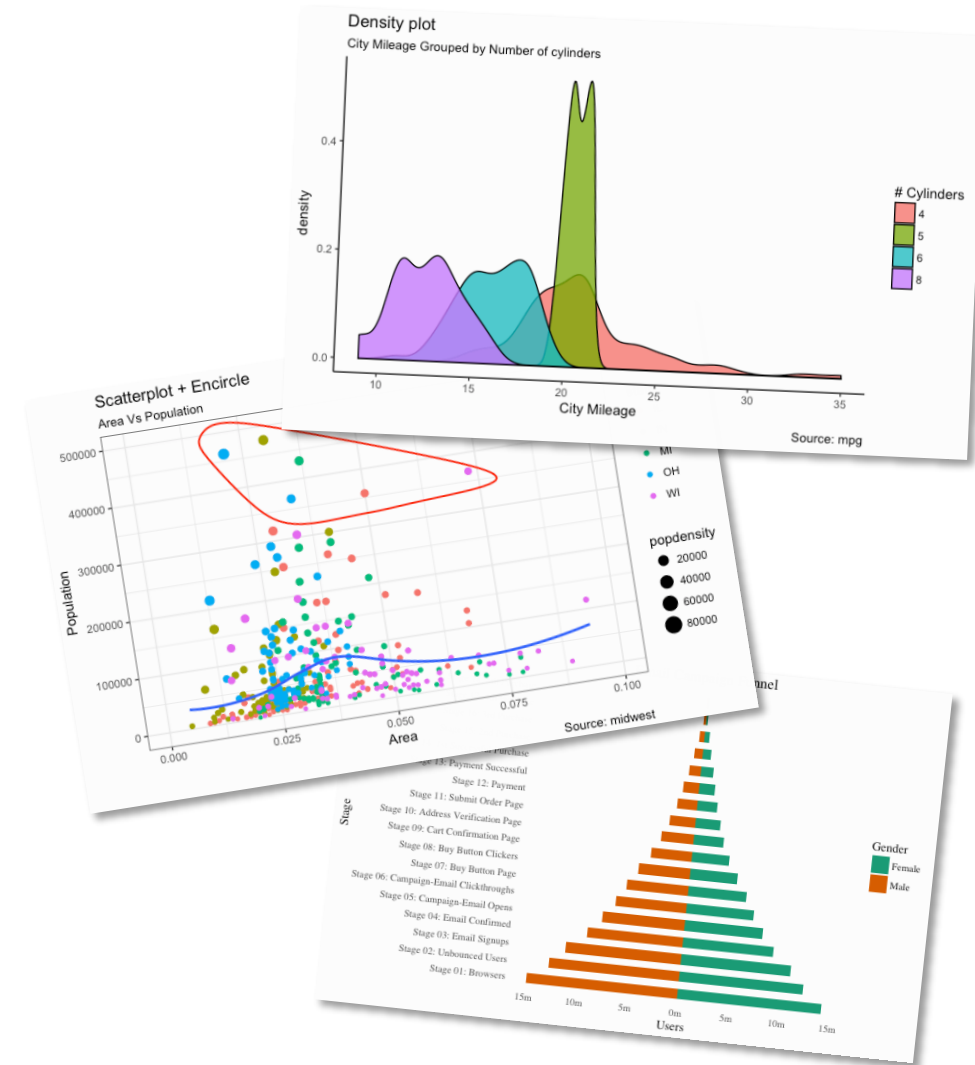
Amy Trost

SCIR XXX

September YY, 2022

What Is R?

o Free programming language with an emphasis on statistical computing and visualization.

o Created in 1993, now supported with more than 14,000 packages in the CRAN repository

o Type commands into a prompt or write longer scripts. Packages do the most of the work.

Growth of packages in the CRAN repository since 1998

Why learn R?

o Open source

o Works with bigger and more diverse data sets than Excel

o Repeat the same analysis on different data using scripts

o Perform sophisticated statistical functions, text mining, data visualization

o Specialized packages have been created for specific disciplines (e.g. economics, biology)

Sample visualizations with ggplot2. Source: http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html

Disadvantages of R

o Syntax is not intuitive

o Documentation can be hard to interpret (but it's all in one place and formatted the same way)

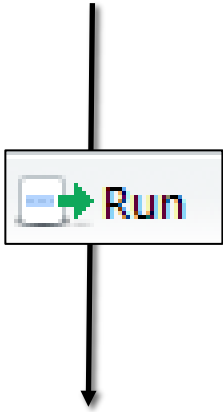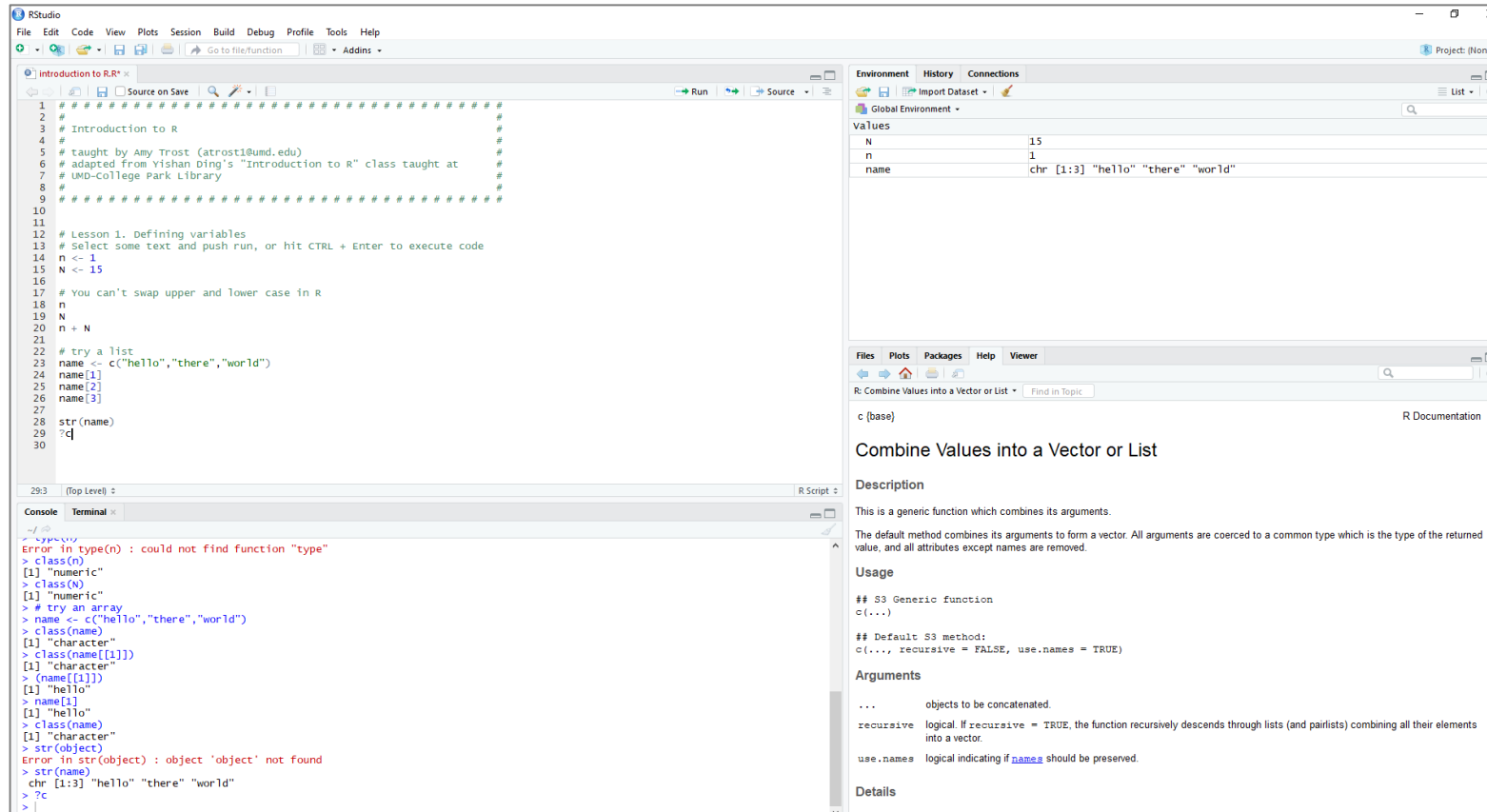o Many packages use a lot of memory and processing power

Today we will learn:

- ❑ **Basic Commands**

- ❑ **Importing Data**

- ❑ **Data Types**

- ❑ **Manipulating Data**

- ❑ **Visualizing Data**

RStudio: an integrated development environment (IDE) for R

Source editor: Text editor for scripts.

Objects: names, properties, and variables

Console: Execute code and examine the log. -- Can also type commands here too!

Library: View plots, install packages, access help

Getting Started—Launching a script
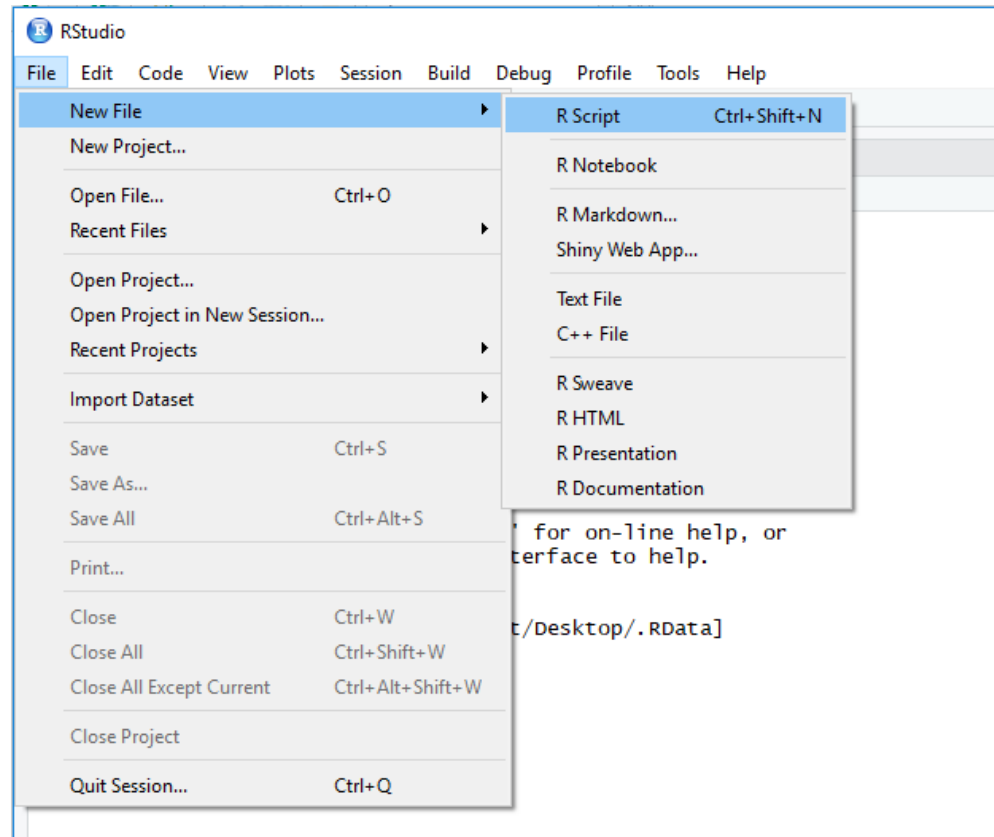
Scripts allow you to edit commands, save your work, and execute many commands at once

Getting Started--Basic Commands

```
# Assigning values to variables
n <- 1
N <- 15

n
N            # You can't swap upper and lower case in R
n + N

# try a list
name <- c("hello","there","world")
name[1]
name[2]
name[3]

# try a simple function
test_scores <- c(90, 72, 85, 93, 84)
test_average <- mean(test_scores)


## ----------------
## Statistical Functions
## ----------------

max(test_scores)        # Find the maximum value in x, excluding missing values
min(test_scores)        # minimum
mean(test_scores)       # mean
median(test_scores)     # median
sum(test_scores)        # sum
var(test_scores)        # variance
sd(test_scores)         # standard deviation

# get help for a command
?mean
```
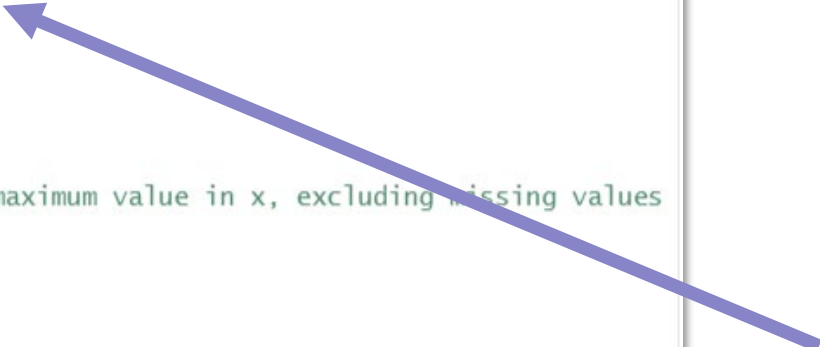
 # marks a comment; R ignores these

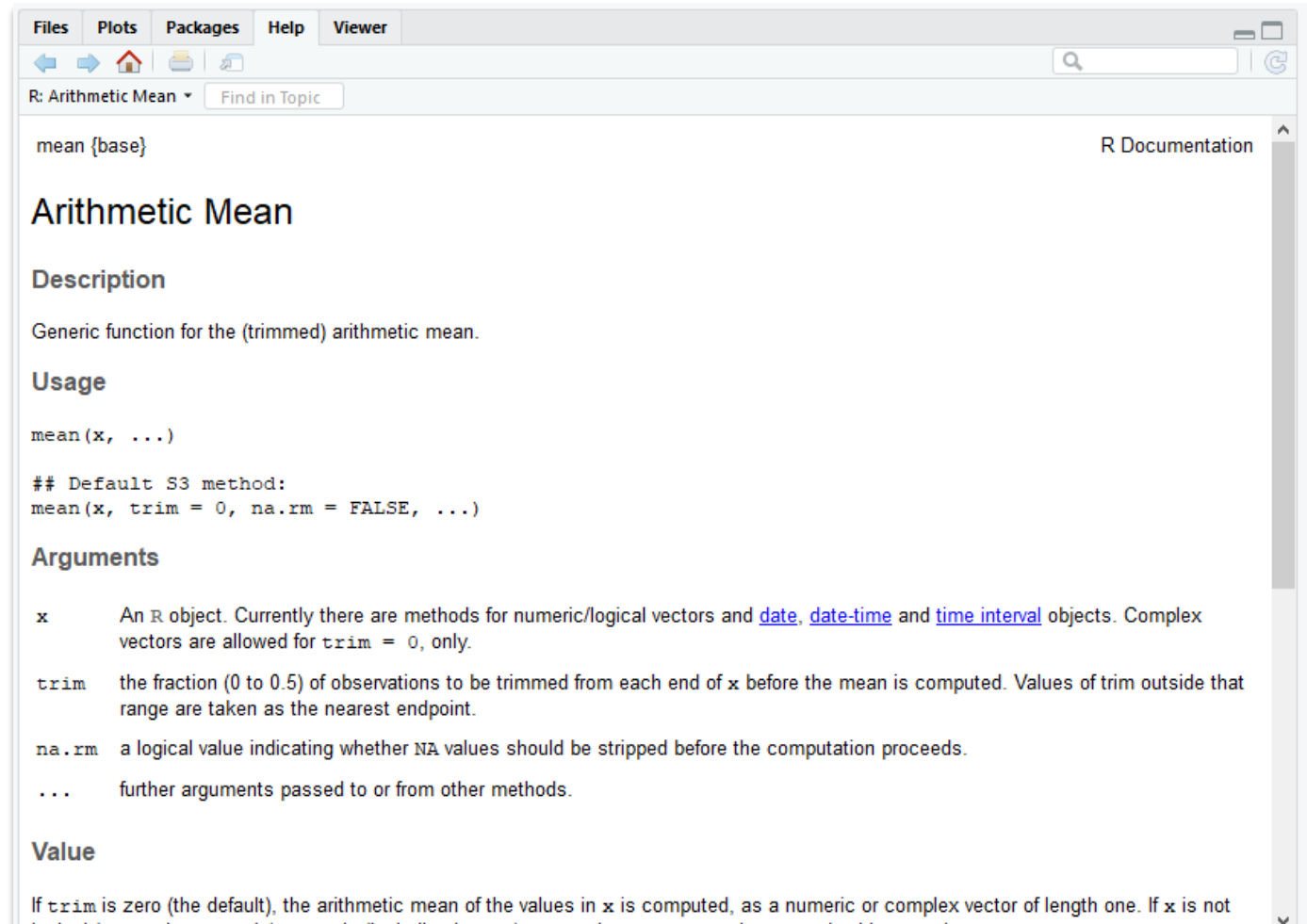**Try typing some of this in to your console (or just run the sample code)**

Statistical Functions

- ❑ `max(x, na.rm = TRUE)`

- ❑ `min(x, na.rm = TRUE)`

- ❑ `mean(x, na.rm = TRUE)`

- ❑ `median(x, na.rm = TRUE)`

- ❑ `sum(x, na.rm = TRUE)`

- ❑ `sd(x, na.rm = TRUE)`

Getting Help

A question mark before a command will launch a help page in Rstudio, for example, typing "?mean" launches

Importing Data (2 files for this session)

1. **Set Working Directory** (under Session) to choose the folder where your data is.

2. **Use the read.csv command:** `species <- read.csv("wikipedia species list.csv")`

   `countries <- read.csv("worldbank google country data.csv")`
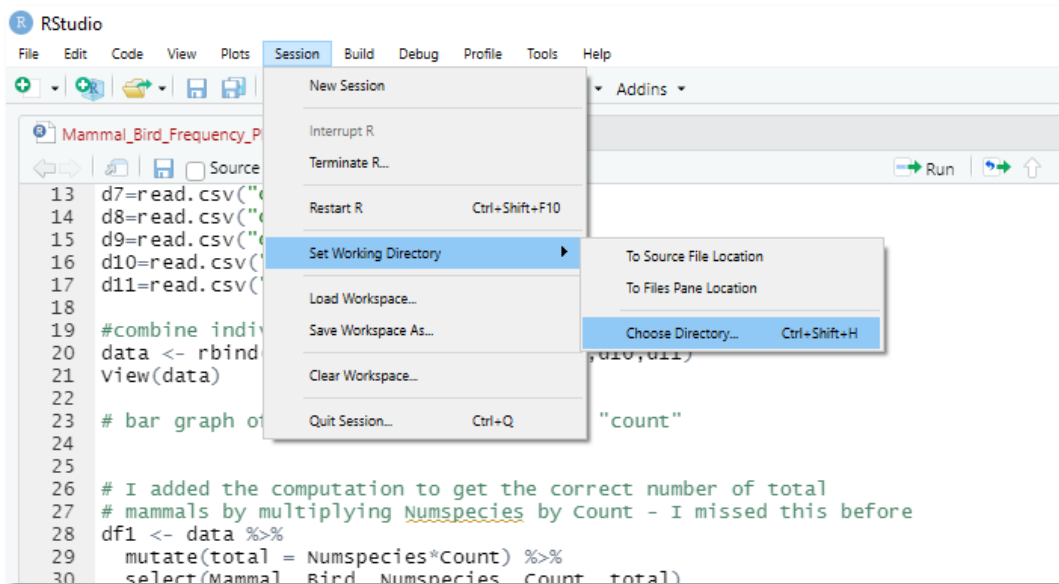
You can also use read.csv on the full file path, e.g.

`countries <- read.csv("C:/Users/atrost/Desktop/worldbank google country data.csv")`

*This can get messy, though, and on windows you need to make sure the slashes are forward / and not backward *

Explore the dataset

- `view()`   open the dataset in a new window
- `class()`   type of data set
- `head()`   preview the dataset in console
- `dim()`   number of rows & columns (dimension) of data set
- `nrow()`   number of rows
- `ncol()`   number of columns
- `colnames()`   column names
- `rownames()`   row names
- `summary()`   shows type and distribution of all of the variables in a data set

Selecting elements of a data set

1. Subset by row/column index

`DATA [ROW, COLUMN]`

- ❑ `countries[6,]`
- ❑ `countries[6:10,]`
- ❑ `countries[,1]`
- ❑ `countries[10:20, 1:3]`

2. Subset by variable name

`DATA$VARIABLE_NAME`

- ❑ `species$common_name`
- ❑ `species$common_name[30:55]`

Data Types

	Class	Example
1	Integer	• 3, 4, 17484, -65
2	Numeric	Can be a decimal, irrational: • 3 • 3.0 • 3.14159
3	Character	"a", "b", "Montgomery College"
4	Logical	• TRUE • FALSE
5	Complex	• 1 + 4i

How to know the class?

o `class()` *what kind of object is it (high-level)?*

o `typeof()` *what is the object's data type (low-level)?*

o `length()` *how long is it? What about two dimensional objects?*

o `attributes()` *does it have any metadata?*

How to change the class?

o `as. numeric(), as.logical(), as.character(),` etc.

Some data structures

	Structure type	What is this?
1	Vector	A simple group of numbers, characters, or logical values
2	List	Flexible, sometimes complicated, stores many types of variables nested within each other.
3	Matrix	Like a vector with more dimensions
4	Data Frame	Very useful!! Represents a data table, with rows and columns of information.
5	Factors	Represent categories of information
6	Tibble	A specialized data frame-like object that is actually a list; used in tidyr and some other packages.

View data types in the "environment" pane of RStudio

Changing Data Types

```r
# changing from integer to numeric
countries$avg_elev_m <- as.numeric(countries$avg_elev_m)


# turning a bunch of variables into factors (categories)
countries$Region <- as.factor(countries$Region)
countries$Income.Classification <- as.factor(countries$Income.Classification)


# view newly organized data set
summary(countries)
levels(countries$Region)
```

Exercise 1

1. How many columns are in the endangered species data set? How many rows?

2. How could you select just the first 3 columns of the countries data set?

3. How could you select just the latin names for the species data set?

4. What kind of variable is the countries data set?

5. What kind of variable is cropland_pct in the countries data set?

6. Use as.factor() to change the species data set. With 3 separate lines of code, turn species_group, IUCN_abbr, and IUCN_category into factors. Summarize the species data set when you are through.

Manipulating Data: Install dplyr package

1. Select "Install" under Packages in R studio. You only have to do this once for a specific computer.
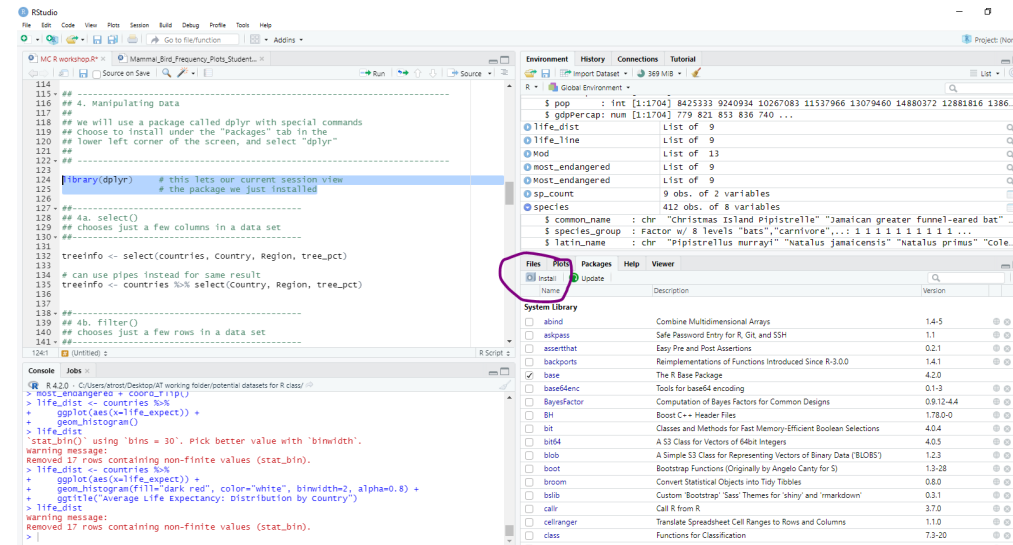
2. Run the library command so that R knows to look for this package in a particular session.

```
library(dplyr)        # this lets our current session view

                      # the package we just installed
```
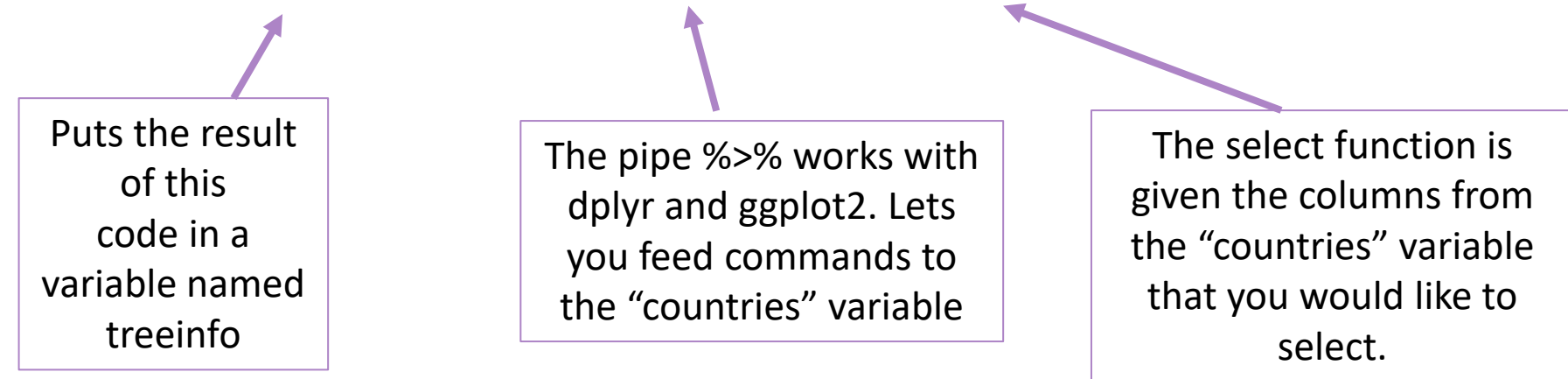
Manipulating Data: Select

```
treeinfo <- select(countries, Country, Region, tree_pct)


# can use pipes instead for same result

treeinfo <- countries %>% select(Country, Region, tree_pct)
```

Puts the result of this code in a variable named treeinfo

The pipe %>% works with dplyr and ggplot2. Lets you feed commands to the "countries" variable

The select function is given the columns from the "countries" variable that you would like to select.

Manipulating Data: Filter

```
# can filter on multiple categories
# just continue long commands on a new line
whales_dolphins_endangered <- species %>%
    filter(species_group =="cetacean",IUCN_abbr==c("CR","EN"))


# can also combine with select to get certain columns
whales_dolphins_endangered <- species %>%
    filter(species_group =="cetacean",IUCN_abbr==c("CR","EN")) %>%
    select(common_name, est_pop, IUCN_category, Trend)
```

Using the "c" function inside the filter function to filter for more than one value. Two equals signs lets R know that you're not trying to change the values of species_group and IUCN_abbr, you just want to find places where the values already match

A second pipe lets you perform a second operation (in sequence) on the "species" data

Manipulating Data: group_by, summarize, arrange

```
species %>% group_by(IUCN_category) %>%

    summarize(n())
```

Summarize with "n" means you are counting the number of species in each category. The pipe %>% lets you perform two functions, group_by and summarize

```
countries %>% group_by(Income_class) %>%

    summarize(median(income_percap, na.rm=TRUE))
```

In this example summarize by median income (remove NAs)

```
countries %>% group_by(Income_class) %>%

    summarize(med_income = median(income_percap, na.rm=TRUE)) %>%

    arrange(med_income)
```

Give median income a variable name so you can reference it in the next line and arrange your results

Exercise 2

1. Filter the species data to only include species with an estimated population of less than 10,000. Select just the species_name, the species_group, and the est_pop .

2. Group the country data by region, then summarize using the median rainfall in mm. Arrange from largest to smallest, using:

```
arrange(desc(med_rainfall))
```

Manipulating Data: Mutate

The backwards arrow feeds the new (mutated) column back in to the original countries data set, making an additional column (variable)

```
countries <- countries %>%
    mutate(rainfall_in = rainfall_mm * .0393701)
```

Mutate creates a new data column (variable) based on existing information. Here we are creating a new column that converts rainfall in mm to inches.

Visualizing Data: Scatterplot

```
ggplot(countries, aes(x=income_percap,y=life_expect)) +
    geom_point()
```

Geom_point tells R you want a scatter plot. You can ask for multiple geoms at a time
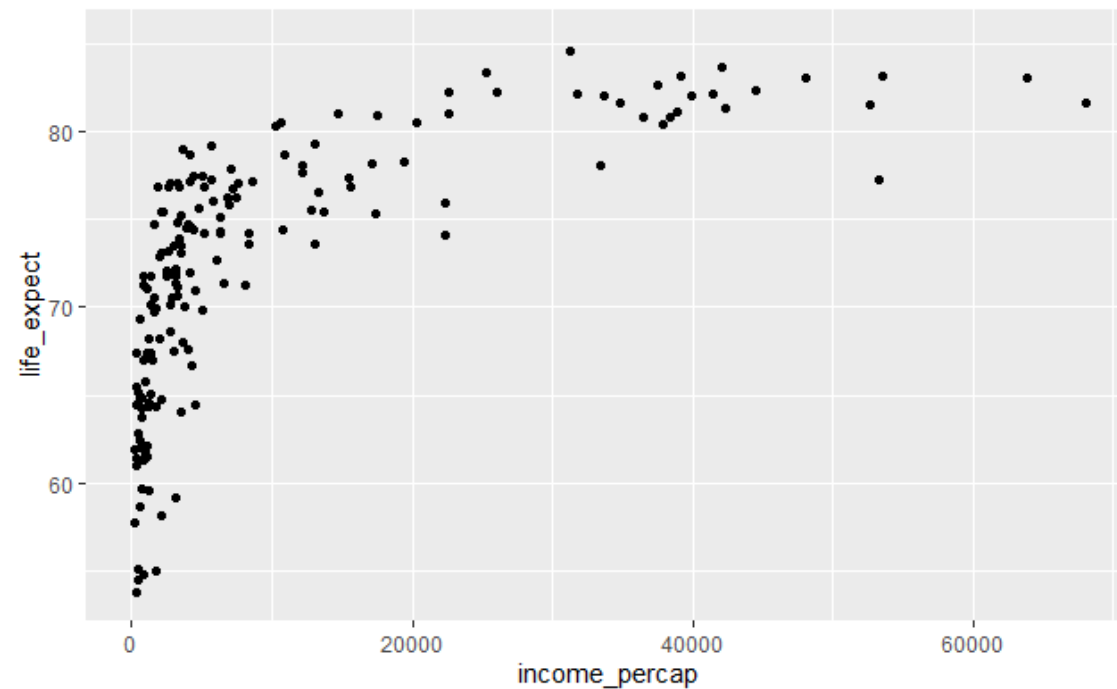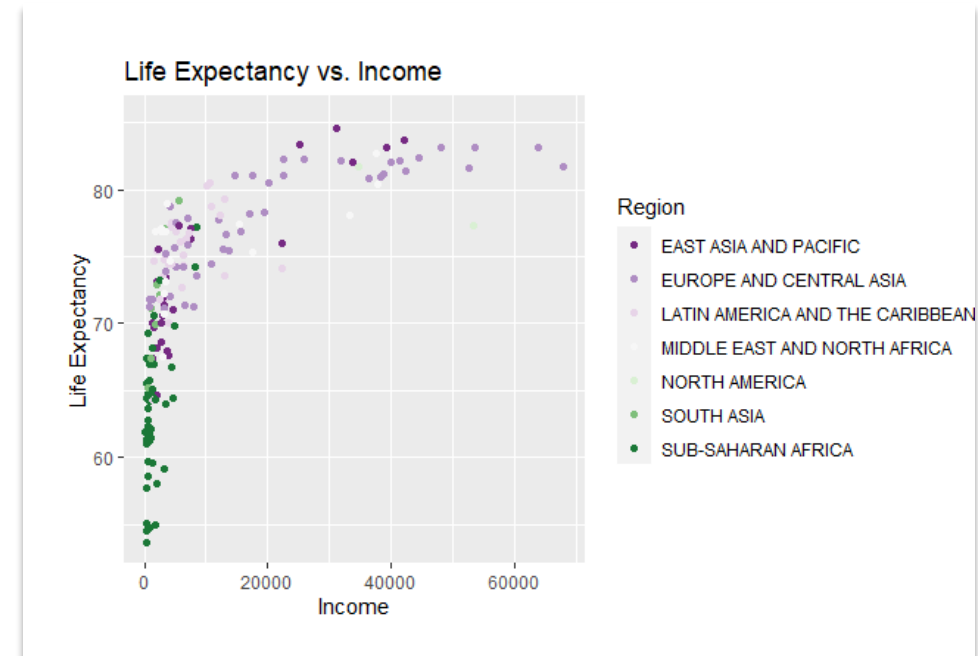
Visualizing Data: Fancier Scatterplot

Gives the graph a name so you can call it up later (and add to it etc). Repeating "country scatter" at the end calls up the actual graphic.

Can list variable name then pipe it instead of including "countries" in the ggplot command

Assigning axis labels and a title here

```
country_scatter <- countries %>%
    ggplot(aes(x=income_percap,y=life_expect)) +
    labs(title = "Life Expectancy vs. Income",y="Life Expectancy", x="Income") +
    geom_point(aes(col=Region)) +
    scale_color_brewer(palette="PRGn")
country_scatter
```

Col=Region tells R to assign color to points based on region. The scale_color_brewer is a special way to add built in color palettes using R.

Life Expectancy vs. Income

Region
- EAST ASIA AND PACIFIC
- EUROPE AND CENTRAL ASIA
- LATIN AMERICA AND THE CARIBBEAN
- MIDDLE EAST AND NORTH AFRICA
- NORTH AMERICA
- SOUTH ASIA
- SUB-SAHARAN AFRICA

Visualizing Data: Line Graph

```
life_line <- gapminder %>%
    ggplot(aes(x=year, y=lifeExp, by=country)) +
    geom_line()
life_line
```

geom_line for a line graph
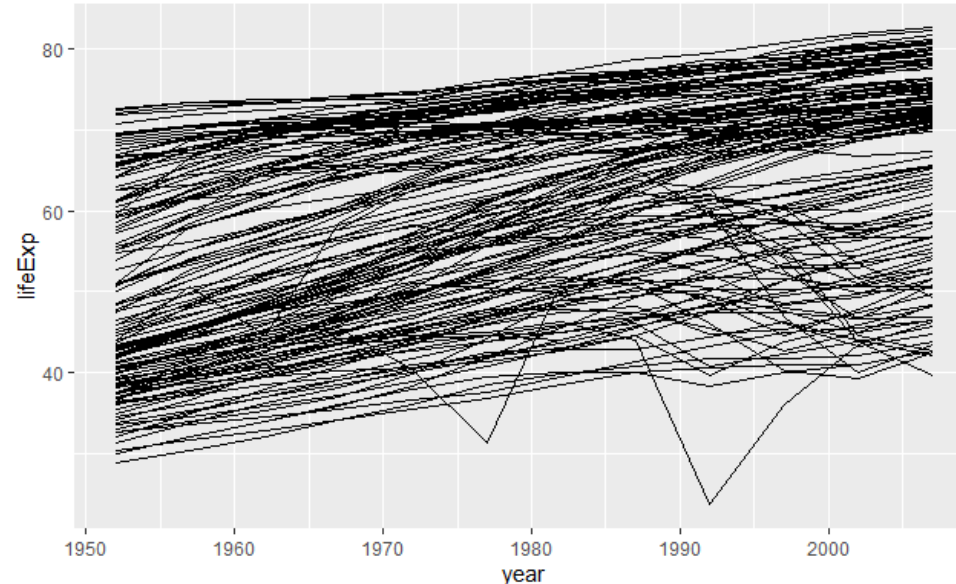
Tells ggplot to create a line
for each country
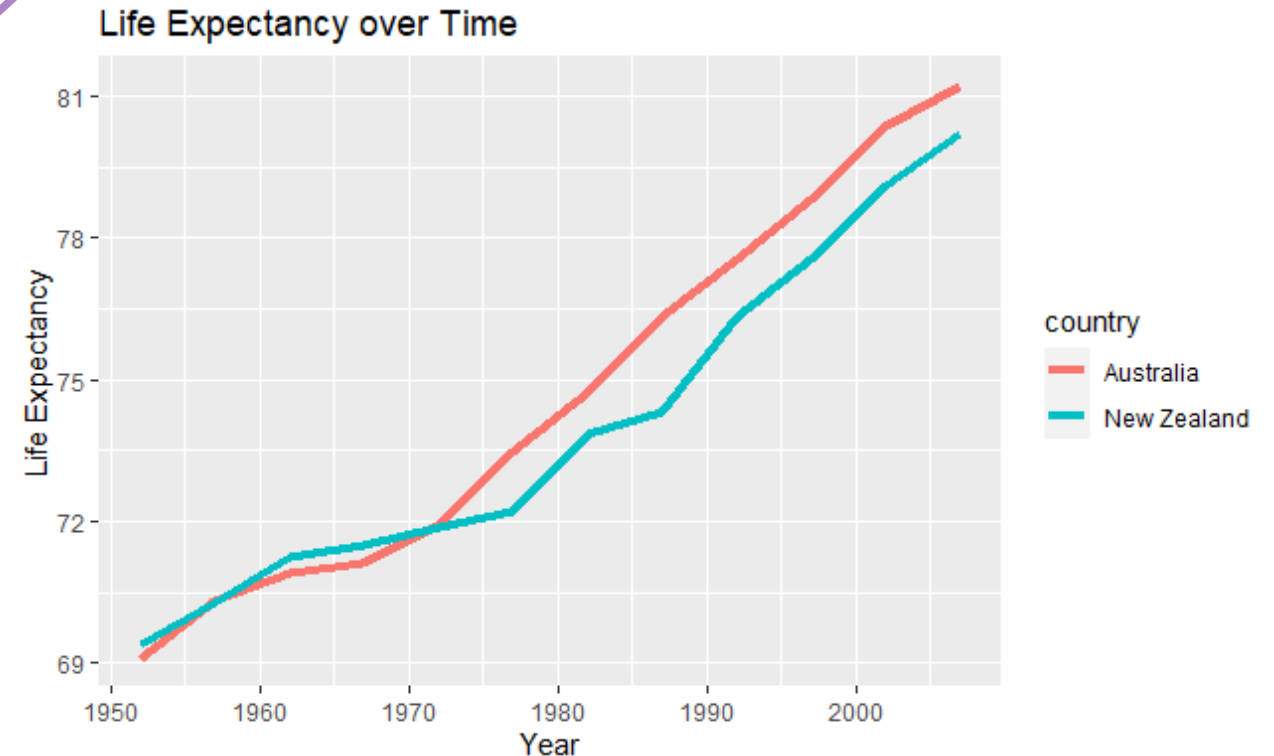
Visualizing Data: Filtered Line Graph

```
life_line <- gapminder %>%
    filter(continent == "Oceania") %>%
    ggplot(aes(x=year, y=lifeExp, by=country)) +
    labs(title = "Life Expectancy over Time",y="Life Expectancy", x="Year") +
    geom_line(aes(col=country), size=1.5)
life_line
```
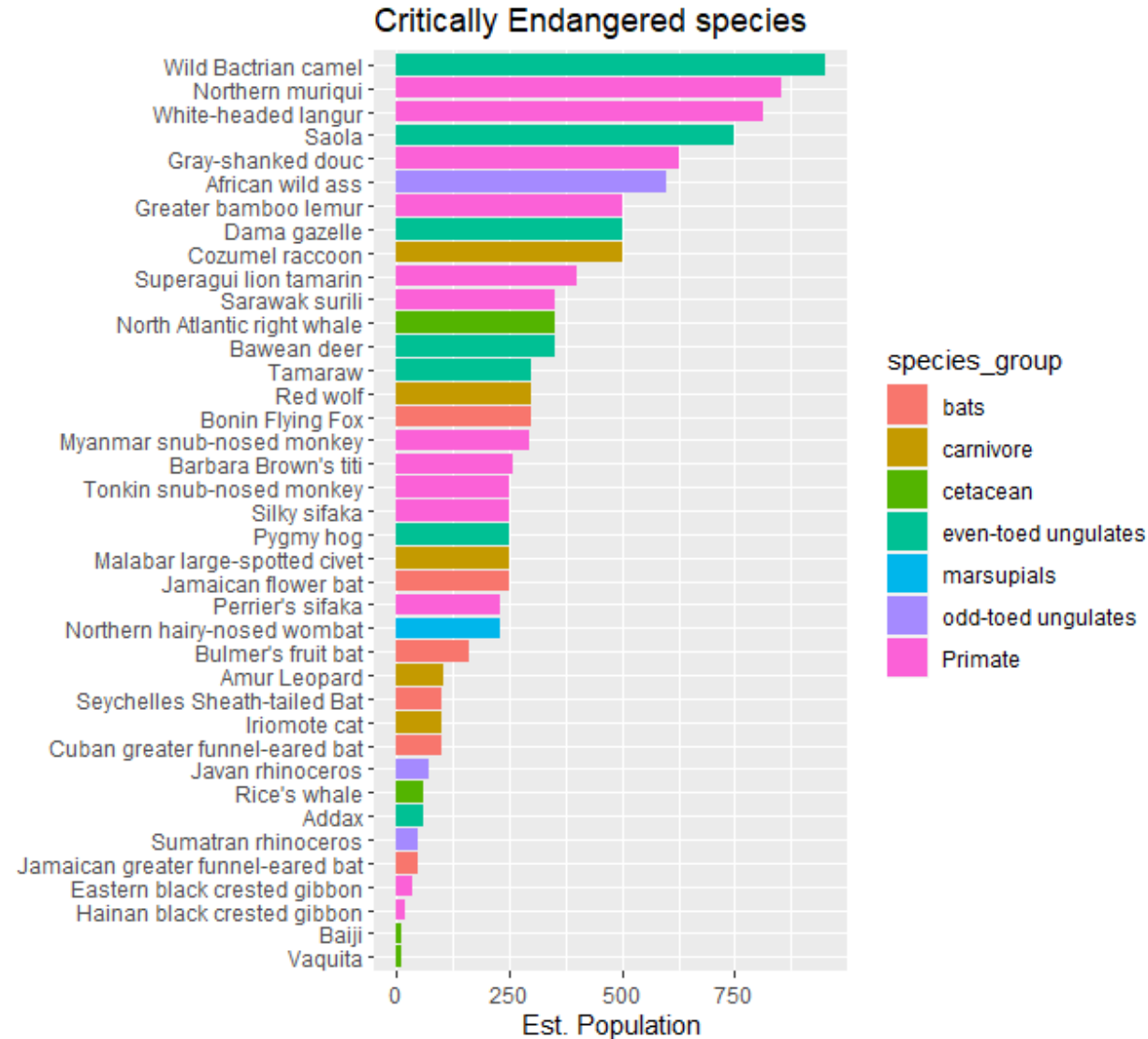
Telling R to only include countries in Oceania

Assigning axis labels and a title here

Col= country tells R to assign color to points based on country. The size=1.5 makes the lines a bit thicker

Life Expectancy over Time

Visualizing Data: Bar Graph

Critically Endangered species

```
most_endangered <- species %>%
    arrange(est_pop) %>%
    filter(IUCN_abbr == "CR", est_pop <1000) %>%
    ggplot(aes(x=reorder(common_name, est_pop), y=est_pop)) +
    labs(title = "Critically Endangered species", y="Est. Population", x=NULL) +
    geom_col(aes(fill=species_group))
most_endangered + coord_flip()
```
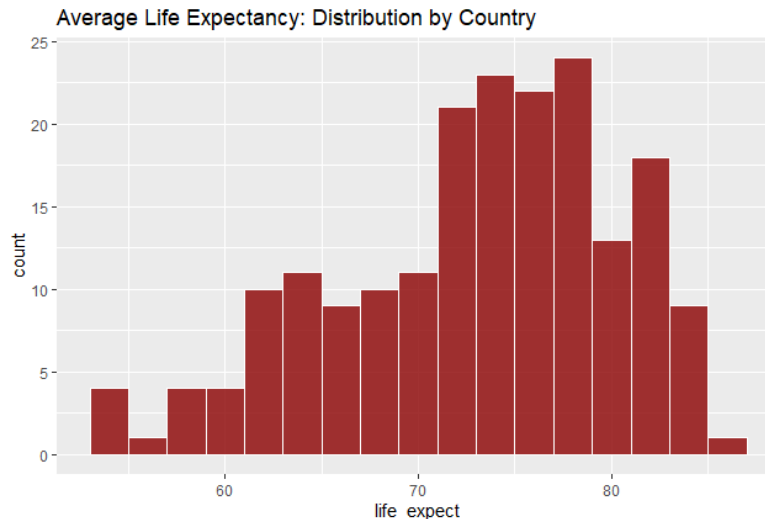
Visualizing Data: Histogram

```
life_dist <- countries %>%
    ggplot(aes(x=life_expect)) +
    geom_histogram(fill="dark red", color="white", binwidth=2, alpha=0.8) +
    ggtitle("Average Life Expectancy: Distribution by Country")
life_dist
```

Creates a new bar for each group of 2 years

Makes bars 80% opaque

Makes the borders of the bars white

Average Life Expectancy: Distribution by Country

Exercise 3

1. Filter the countries data set to include only data from South Asia. With this subset, create a bar chart that reports average rainfall in mm for each country.

2. Create a scatter plot with the countries data set that shows the relationship between tree cover and rainfall. Color the dots by region, label the x & y axes, and give the chart a title.

Thank You

QUESTIONS? PLEASE CONTACT ME AT
ATROST@MONTGOMERYCOLLEGE.EDU