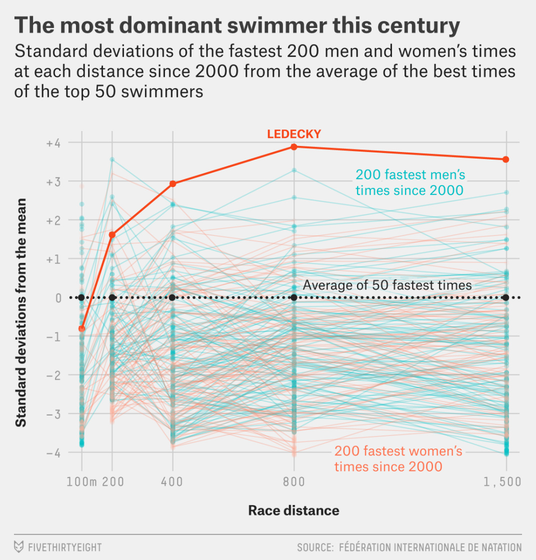
Intro to Visualization

Introduction

- Today we will talk about communicating your analysis to others
- The most effective way in almost all cases is visually.
- Please note: For more information on today's lecture look at chapter 3 of R for Data Science at r4ds.had.co.nz.

Katie Ledecky is Amazing



The most dominant swimmer this century

Standard deviations of the fastest 200 men and women's times at each distance since 2000 from the average of the best times of the top 50 swimmers

FIVETHIRTYEIGHT

SOURCE: FÉDÉRATION INTERNATIONALE DE NATATION

Overview of today's lecture

- Review how to read in data
- How to examine data
- Line and scatter plots with ggplot.
- Other types of plots with ggplot

Financial Question

- Interest rates on government bonds
- The main way in which the federal government raises money
- We want to know:
  - How do interest rates for bonds of different payback periods behave
    - relative to each other
    - over time

What is a Bond?

- I owe you
  - You give the government money today with the promise of getting a fixed amount in the future
  - Additionally you get a annual payment specified by the interest rate
- Tenor is the length of the payback period
- A 5-year bond worth \$10,000 that pays \$1,000 per year has a 5-year tenor and 10% interest rate

Why does the Federal Reserve Care?

- Dual mandate - Inflation and Unemployment
- Interest rates = price of money
- Fluctuations help heat/cool the economy

Why is this Important?

- Interest rates affect how much money you can borrow
  - Home-loan
  - Car-loan
  - Student-loan
- Interest rates affect how quickly your money or debt grows

Review of Reading in Data

- Our first step is getting our data into R
  - This data is stored in Data/treasuries.csv.
- We read in this data with `read.csv()`
  - What arguments does this function take?
  - What is the output?

Reading in Data: read.csv()

```r
treasuries <- read.csv("Data/treasuries.csv",
                       stringsAsFactors = F,
                       header = T)
```

- file path
- Logicals: `T/TRUE` and `F/FALSE`
- Why am I using `<-`?

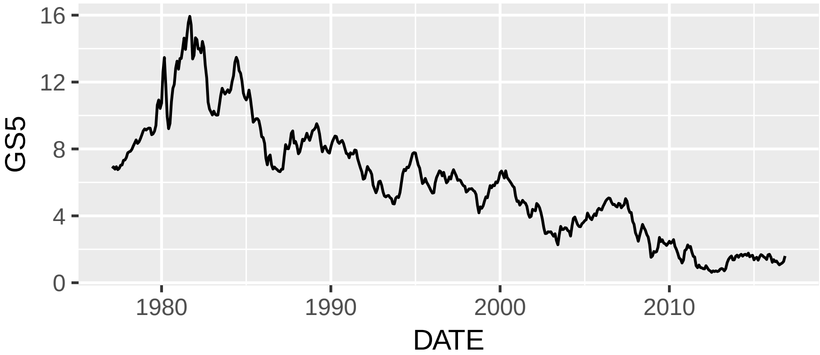
Review of data.frames

- Data.frames are similar to an excel spreadsheet
    - Every column in a data.frame must have the same number of rows
        - If a datapoint is missing in a row, an `NA` value will be placed there instead

Accessing data in your data.frames

- `data.frame$column`
- Subset rows and columns using `[]` notation
    - Try it: `treasuries[3, 4]`
- Can also use column names:
    - `treasuries[3, c("DATE", "GS10")]`

Goal Chart



Examining our Data

- First step: examine data
- `head()` and `tail()`
- Here are the first 3 rows of our data:

```
head(treasuries, 3)
```

```
Output:          DATE  GS1 GS10  GS3 GS30  GS5 UNRATE
Output:  1 1977-02-01 5.47 7.39 6.44 7.75 6.83    7.6
Output:  2 1977-03-01 5.50 7.46 6.47 7.80 6.93    7.4
Output:  3 1977-04-01 5.44 7.37 6.31 7.73 6.79    7.2
```

Exercise: Examining our Data

- Ok, now it's your turn: using the `tail()` function, I want to see the last 4 rows of treasury data, your answer should look like this.

```
Output:            DATE  GS1 GS10  GS3 GS30  GS5 UNRATE
Output:  475 2016-08-01 0.57 1.56 0.85 2.26 1.13    4.9
Output:  476 2016-09-01 0.59 1.63 0.90 2.35 1.18    5.0
Output:  477 2016-10-01 0.66 1.76 0.99 2.50 1.27    4.9
Output:  478 2016-11-01 0.74 2.14 1.22 2.86 1.60    4.6
```

- What do you think are the appropriate types for this data?

Treasury Data definitions

- Date is the date, in month/day/year format.
- GS1, GS10, GS3, GS5, GS30 are all interst rates for government bonds of different lengths
  - The number corresponds to the number of years
  - So entry on 11/1/2016 in the GS10 column of 2.14 means that a 10 year bond released on 11/1/2016 had an interest rate of 2.14%
- UNRATE corresponds to the unemployment rate, as a percent of the labor force population.

Checking all of Our Classes

- want numeric data for plotting
- the `class()` function

```
class(treasuries$GS1)
```

```
Output:  [1] "numeric"
```

Checking Types Exercise

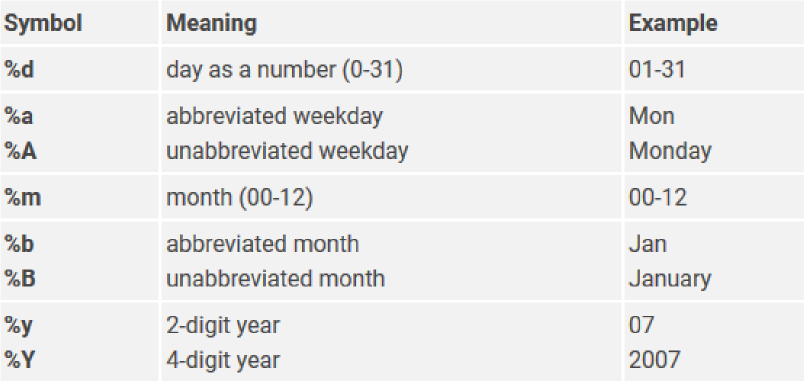
- Using the `class()` function and the $ notation check the type for the rest of our columns.
- What is the class of your “DATE” column?
- What is the class of your UNRATE column?
- Which columns are not numeric?

Converting our values to dates

- `as.Date()`
- Let's take a look at the help for `as.Date()`
  - It looks like we need to provide an object, `x` and a `format` to the function.
  - http://www.statmethods.net/input/dates.html

```
treasuries$DATE <- as.Date(treasuries$DATE,
                           format = "%Y-%m-%d")
```

Common date formats



Symbol	Meaning	Example
%d	day as a number (0-31)	01-31
%a %A	abbreviated weekday unabbreviated weekday	Mon Monday
%m	month (00-12)	00-12
%b %B	abbreviated month unabbreviated month	Jan January
%y %Y	2-digit year 4-digit year	07 2007

Figure 1:

Review

- Always examine your data
- Convert to proper types
- Also, don't forget to always examine your data
- The next step is to select the data that we want to plot and then graph it.

Packages in R

- ggplot2 package for all plotting
- people can write code and share it with other people
- To see the complete list of packages available, check out CRAN at https://cran.r-project.org/

Installing a package

- `install.packages()`
  - (download it from the internet)
  - need to supply the name of the wanted packaged in quotes
- https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf
  - Also on the Help tab in Rstudio

```r
install.packages("ggplot2")
```

Attaching a package

- Make the functions usable
- `library()`

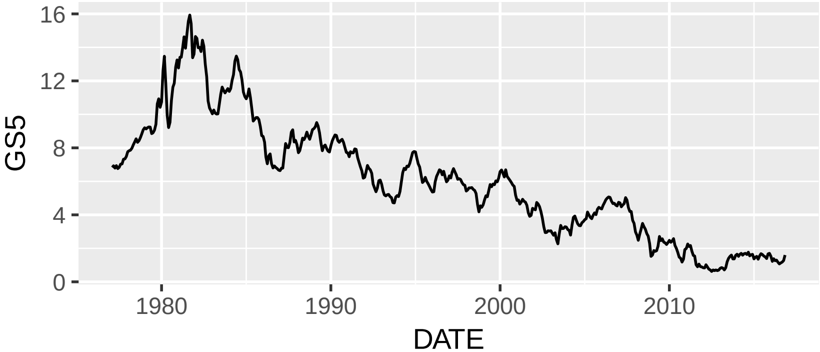
```r
library(ggplot2)
```

- `install.package()` once
- `library()` every time

First plot with ggplot

- Interest rate on 5 year bonds over time:

```
GS5_data <- treasuries[, c("DATE", "GS5")]
ggplot(data = GS5_data,
       aes(x = DATE, y = GS5)) +
    geom_line()
```

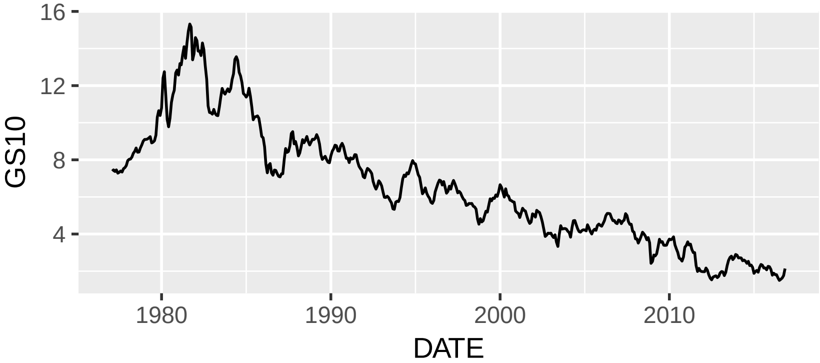


First Chart: Analysis

- Ggplot works in the following way:
  - You start your plot with a call to `ggplot()`
  - assign a dataset using the `data =` argument
  - set your `x` and `y` variables using `mapping = aes(...)`
  - Using the + symbol you can combine multiple layers for your plot

First plot: exercise

- Using the code from the previous slide, make a chart showing the interest rate of 10 year bonds over time



Treasury rates

- 10-year and 5-year bonds are really similar
  - Why?
- More recent data?
  - 2016 only and zoom in on the recent data
- multiple lines on the same chart?

Selecting our Data

- Our goal is to graph only data from the year 2016.
- Using the `tail()` function, find out what the last row of data shows, what is the date for the last row and what is the date for the second to last row?
  - frequency of our data?

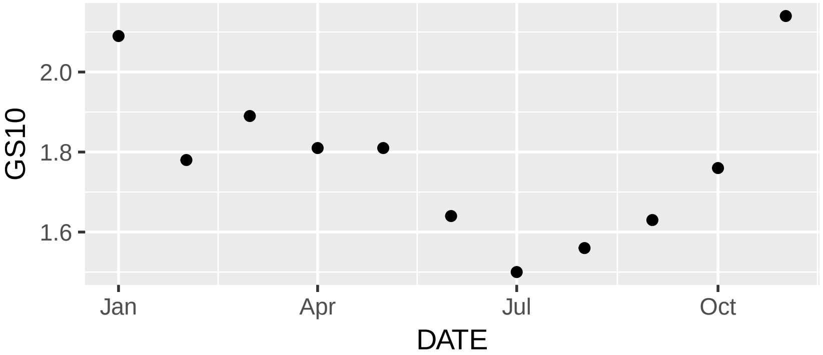
Selecting our Data part II

- last date November 1, 2016, and the second to last date is October 1, 2016.
  - monthly data and 11 observations in 2016
- Could use the `[` with our data.frame.
- Or could use `tail()`

```r
data_2016 <- tail(treasuries, 11)
```

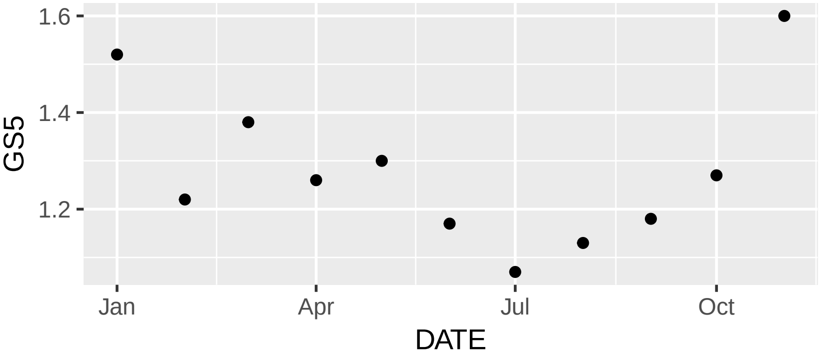
First scatterplot

```
ggplot(data = data_2016,
       mapping = aes(x = DATE, y = GS10)) +
    geom_point()
```



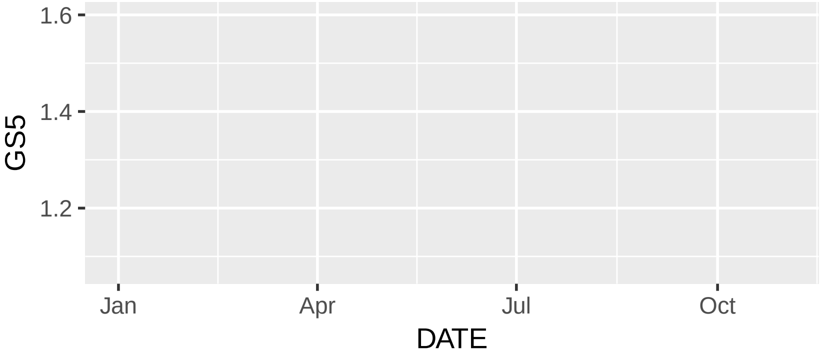
In Class Exercise: GS5

- Make a scatterplot for the 2016 values of the `GS5` column in the treasuries data.frame.



Forgetting your geom

- What would happen if you did not use `+ geom_point()`?
  - Try omitting `geom_point()`



Intro to Layering

- Empty plot:
  - build the plot with layers
  - Easier to make changes
- Start with empty plot and add layers
- Line chart instead of a scatter-plot.

–>

Storing our plots

- assigning the plot to a variable
  - The plot itself is an object
  - Calling the plot causes it to be printed
  - Can add more layers to an existing plot, we will see this later.
- Save your plots using `ggsave()`.
  - file destination and a plot
- Using `ggsave()` create “five_year_line_plot.png”

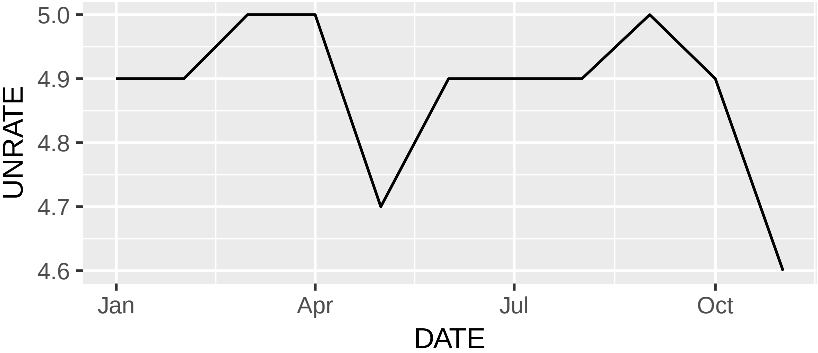
Filtering date values

- Can select rows based on DATE values instead of using `tail()`
- Logical comparisons:
  - less than, less than or equal to, greater than, greater than or equal to etc

Plotting with date filters

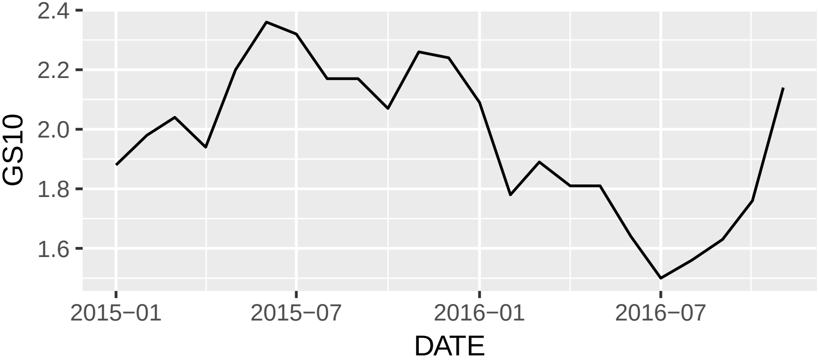
- Plot 2016 without tail()

```
ggplot(data = treasuries[treasuries$DATE >=
                                    as.Date("2016-01-01"),],
       aes(x = DATE, y = UNRATE)) + geom_line()
```



In Class Exercise: Line Plot

- Make a line plot for the 10 year treasury data for 2015 and 2016. After creating your plot, save it as "GS10_2015_2016.pdf"
  - Note that you should save it as a pdf
  - Your plot should look like this:



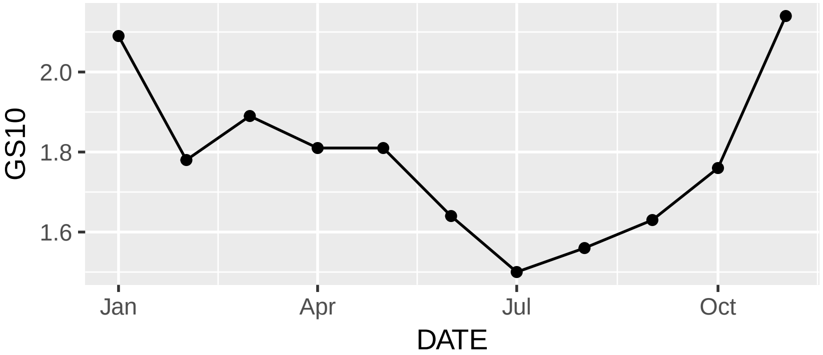
Why do you think the rate went down in 2016 only to go back up immediately after?

Review

- Remember, always examine your data
  - Check for types
- `ggplot()` creates and empty plot
  - takes `data =` and `mapping =` arguments.
  - mapping with the `aes()` function.
  - Build layers with +
  - Let's see this in action now.

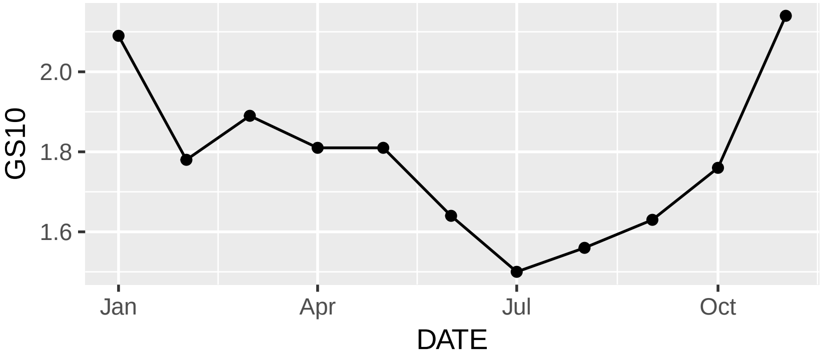
Layering with ggplot

- Add points to a line chart
- Save plot into a variable
- Add the layers



You could make the chart all at once

```
ggplot(data = data_2016,
       aes(x = DATE, y = GS10)) +
    geom_line() + geom_point()
```

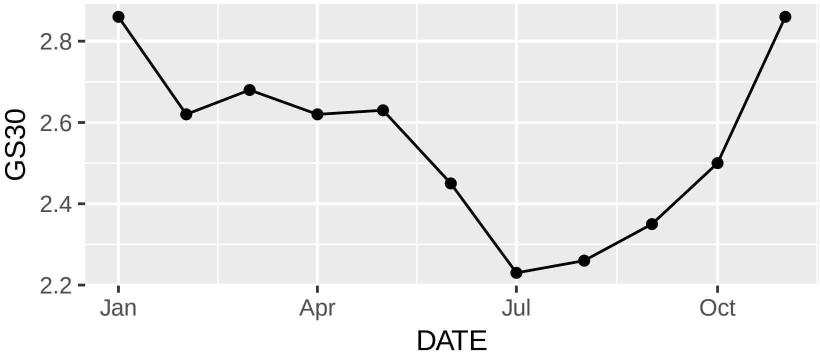


Behind the Scenes

- For layer_two how did ggplot know what data to use?
  - `x` and `y`?
- `ggplot()` takes the data for the chart and the mapping for the chart.
  - Subsequent calls inherit the data and the mapping from the `ggplot()` function.
    - It is possible to override

In Class Exercise: Layering

- Create a layered line and point chart for the **GS30** data for the year 2016.



What does it all mean?

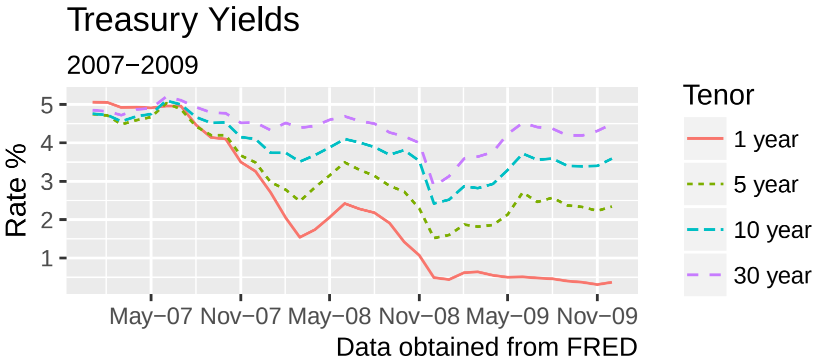
- What do these lines mean?
- The values are treasury **yields** on bonds of different **tenor**.
  - A bond is similar to a loan
    - The length of time until the bond is paid back is called the "tenor."
    - So a $100,000 bond with a 30 year tenor means that in 30 years the bond holder is due $100,000

Yield on Bonds

- But what about inflation?
  - $100,000 today worth more than $100,000 in 30 years.
- bonds also pay interest
  - Percentage of the face-value
  - A $100,000 bond with a 10% interest rate would pay the bondholder $10,000 per year.
- What do you think are some reasons that an interest rate on a bond would fall?
- What are some reasons that an interest rate on a bond would rise?
- How do you think the interest rates on bonds changed during the most recent recession?

Bond yields by tenor over time

- What do you notice about how the interest rates move
  - Why might this be?



Let's learn how to make this chart

Clarifying your graphics

- Multiple lines on the same chart
- Add titles and labels
  - check out how the dates are formatted!
- add a caption describing the source of our data, always a good practice.
  - FRED is Federal Reserve Economic Data
  - A public database you should definitely check out!

Plotting multiple lines: Selecting your data

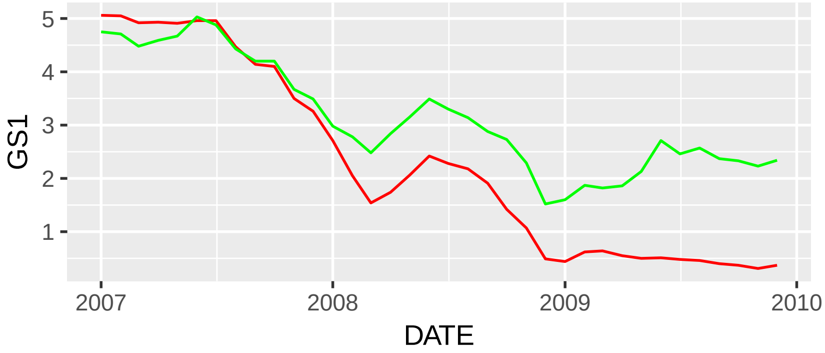
- Select data
  - Use logical filtering on it

```
recession_data <- treasuries[treasuries$DATE >=
                            as.Date("2007-01-01")
              & treasuries$DATE < as.Date("2010-01-01"),
              c("DATE", "GS1", "GS5", "GS10", "GS30")]
head(recession_data, 3)
```

```
Output:            DATE  GS1  GS5 GS10 GS30
Output:  360 2007-01-01 5.06 4.75 4.76 4.85
Output:  361 2007-02-01 5.05 4.71 4.72 4.82
Output:  362 2007-03-01 4.92 4.48 4.56 4.72
```

Plotting multiple lines: First two lines

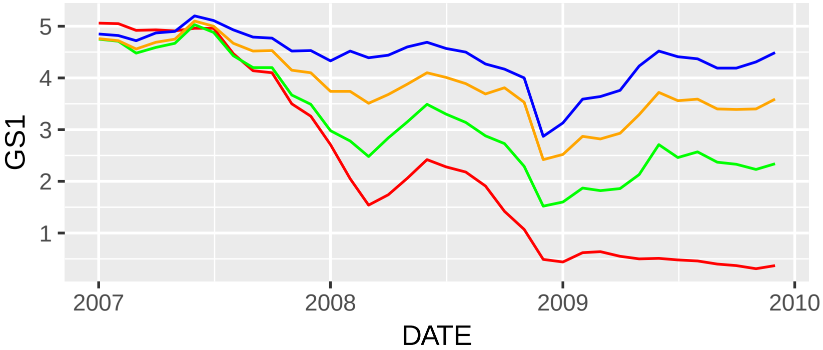
```
ggplot(data = recession_data) +
    geom_line(mapping = aes(x = DATE, y = GS1),
              color = "red") +
    geom_line(mapping = aes(x = DATE, y = GS5),
              color = "green")
```



What's going on here with the 1 and 5 year interest rates?

In class exercise: plotting multiple lines

- Add the lines for `GS10` and `GS30`, they should be orange and blue respectively.



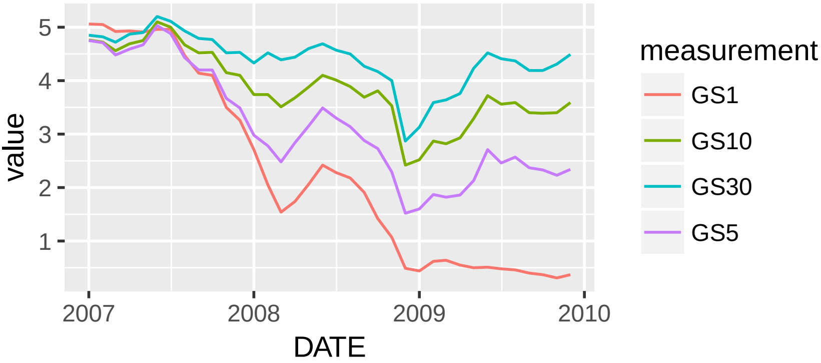
Plotting multiple lines: Wide vs. long data

- manually adding the lines and points
  - This is time consuming and tedious
  - There is a better way
- Wide vs long data
  - Instead of having a column for each different tenor bond (4 columns total), we could have only 2 columns
  - One column would have the length of the tenor and the other would have the value
- **Read in the data we will need for this exercise: treasuries_long.csv**

Plotting multiple lines: the power of aes()

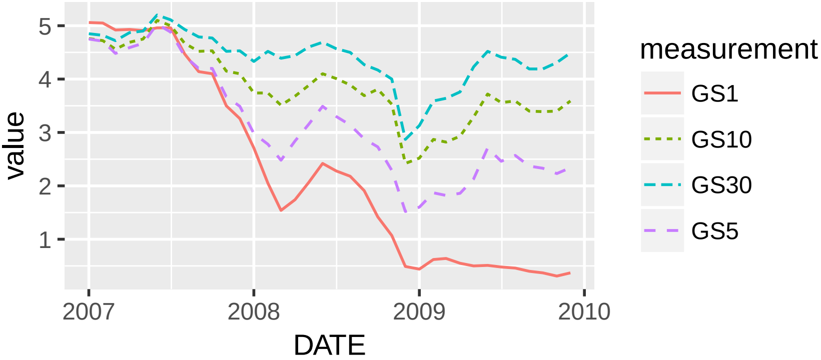
- `aes()` function maps data to an aesthetic.
  - mappings other than x and y are possible
    - color

```
ggplot(data = melted_data,
       aes(x = DATE, y = value, color = measurement)) +
    geom_line()
```



Let's map the linetype.

```
ggplot(data = melted_data,
       aes(x = DATE, y = value, linetype = measurement,
           color = measurement)) +
    geom_line()
```



Single guide for our color/linetype.

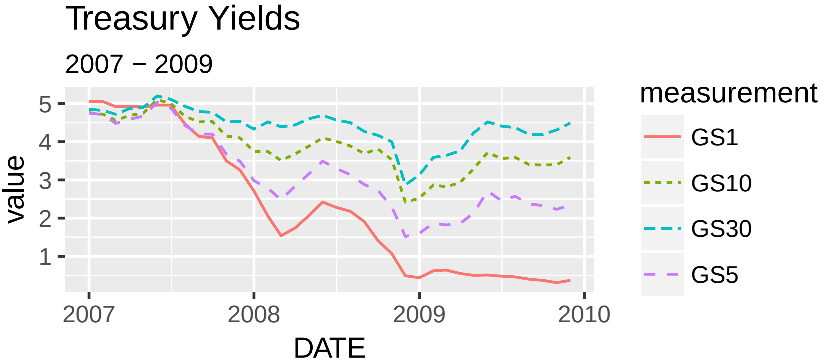
Plotting multiple lines: titles and labels

- Let's add on the points
    - Assign our plot as first_layer

```
first_layer <- ggplot(data = melted_data,
    aes(x = DATE, y = value, linetype = measurement,
        color = measurement)) +
    geom_line()
```

- `ggtitle()`
    - Can also add a subtitle

```
second_layer <- first_layer +
    ggtitle("Treasury Yields",
            subtitle = "2007 - 2009")
second_layer
```



Plotting multiple lines: Customizing our scales

- Data displayed on scales.
  - x and y
  - A scale maps a datapoint to the visual representation of the datapoint.
    - x and y values these are locations
    - Color scale takes a text string, such as "GS1," and maps it to a value, "red."
- Scale_ functions control the scales
  - scale_y_continuos, etc...
- The first argument of the scale_ functions is `name =`
  - Give y-axis the name "Rate %"

Cleaning up our color scale

- Using discrete value for color scale
- Use `scale_color_discrete`
- Why did the guides separate?

Plotting multiple lines: Harmonizing scales

- Using same data for color and linetype scales
  - Changed one but not the other
  - Update labels to fix

Plotting multiple lines: Date scales

- Time series data: dates displayed on x-axis
  - Use scale_x_date.

```r
sixth_layer <- fifth_layer +
    scale_x_date(name=NULL,
            date_breaks = "6 months",
            date_labels = "%b-%y")
```

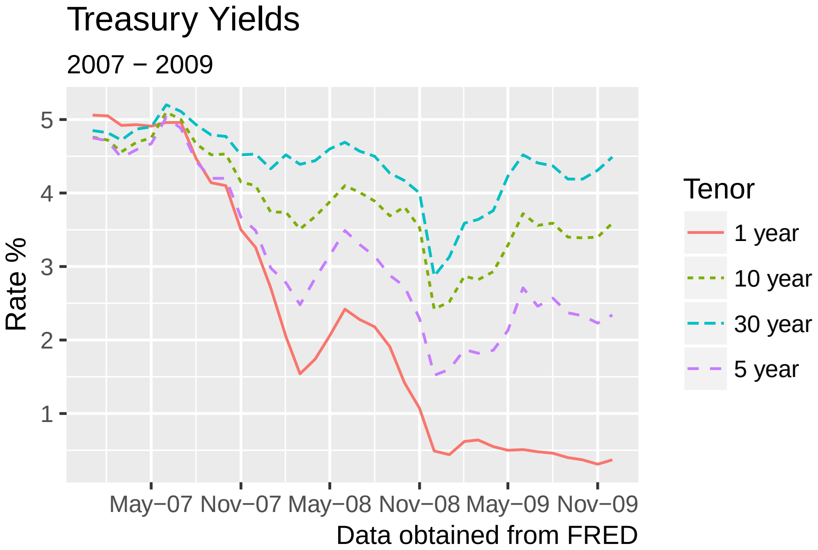
Woah, what are `NULL`, `date_breaks =` and `date_labels =`?

- `?scale_x_date`
- No need for a name
  - Use `NULL` to indicate something has no value in R
- The `date_breaks =` argument takes a text string
  - number and range combinations:
    - "1 year", "2 weeks", "3 months", "5 days", "2 hours", etc. . .
- The `date_labels =` argument same type as the `format =` argument for `as.Date()`
  - What do "%b" and "%y" mean?

Plotting multiple lines: adding a caption

- Add sources of your data to your graphics
  - Helps you and your reader!
- `labs()`
  - This functions is short for “labels”
  - can set all labels in the plot (title, x-axis, etc. . .)
  - `caption =` argument

Final Plot



Treasury Yields

2007 − 2009

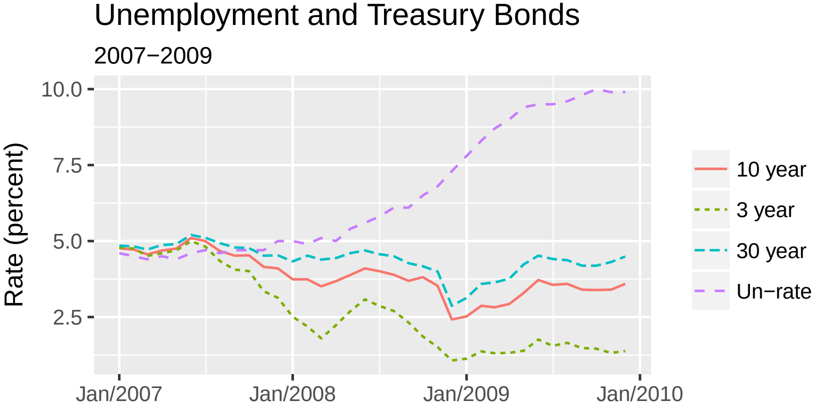
Data obtained from FRED

Plotting Multiple lines: Recap

- Using long data
  - Fully utilizes `aes()`
- scale functions customize your chart
- Unify your labels to get unified guides
- Using clear titles and labels helps communicate your data
  - Other people understand what you did
  - You understand what you did

In Class Exercise: Plotting Multiple Lines

- Now it's your turn, make a plot of the unemployment rate, 3, 30, and 10 year treasury yields for 2007-2009.
- Use the `multiple_line_exercise_data.csv` file



Correlation between the unemployment rate and the interest rate on Treasury bonds?

Is your data special?

- We've seen lots of data on government bonds
- Is the recession different?
- Measures of spread in our data
  - Compare 2008-2009 with the rest of our data

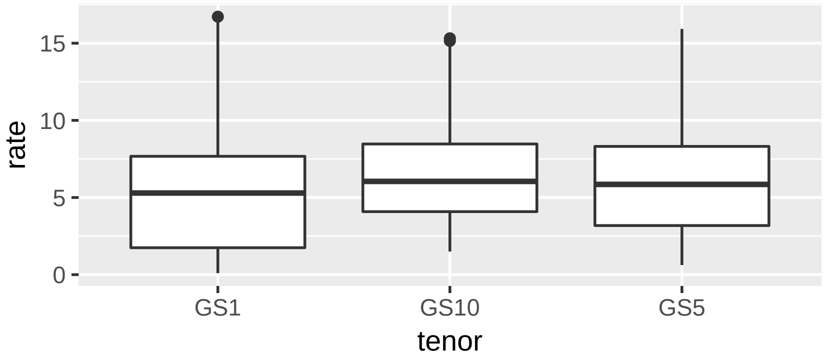
Communicating different aspects of your data: Geoms

- Line and scatter plots:
  - Good for distinct data
  - Less than $\sim$ 6 clearly different line
  - Not good with lots of data - message gets lost in overlap

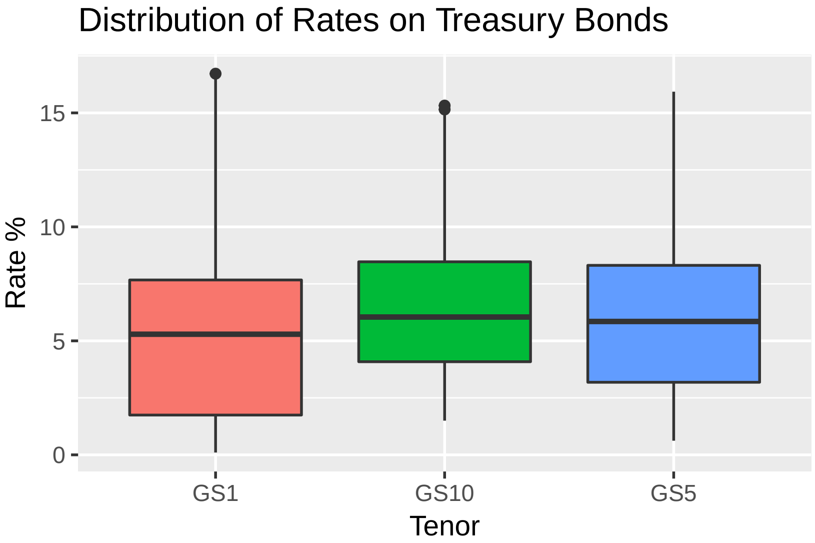
Box Plots

- Showing the spread of your data.
  - Box plots show the median, 25th and 75th percentiles, the inter-quartile range of your data, the minimum and maximum, and outliers
  - The inter-quartile range, (IQR) = area between the 25th and 75th percentiles

```r
box_data <- read.csv("Data/boxplot_data.csv",
                     stringsAsFactors = F,
                     header = T)
box_data$DATE <- as.Date(box_data$DATE)
ggplot(box_data, aes(x = tenor, y = rate)) +
    geom_boxplot()
```

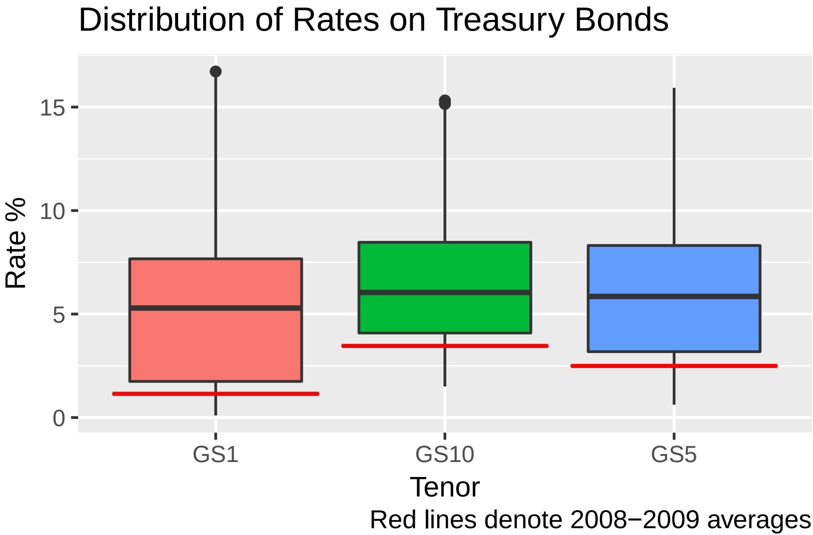


- control the color of our boxes:



Comparing to our Recession values

- I calculated the 2008-2009 average values
  - rec_data
- Add these lines to our bar chart
- `geom_crossbar()`


Distribution of Rates on Treasury Bonds
Rate %
15
10
5
0
GS1
GS10
GS5
Tenor
Red lines denote 2008−2009 averages

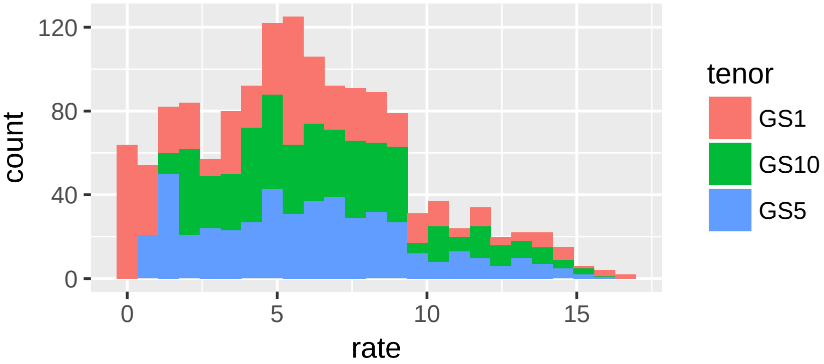
Analyzing our bar charts

- Values during the recession are very low compared with the overall distribution
- Area for statistical analysis
- Power of visualization
  - Easy to grasp the idea

Histograms

- Box plots show range
- Histograms show distribution
  - For example, if you take the sets: [1, 2, 2, 2, 5] and [1, 3, 4, 5, 5] the range of each set is the same, (1-5), but the distribution of the data is different.
- ggplot2 has two main methods
  - `geom_histogram`
  - `geom_density()` function
    - smoothed version of the histogram

```r
histogram <- ggplot(box_data,
                    aes(x = rate, fill = tenor))
histogram + geom_histogram(bins = 25)
```

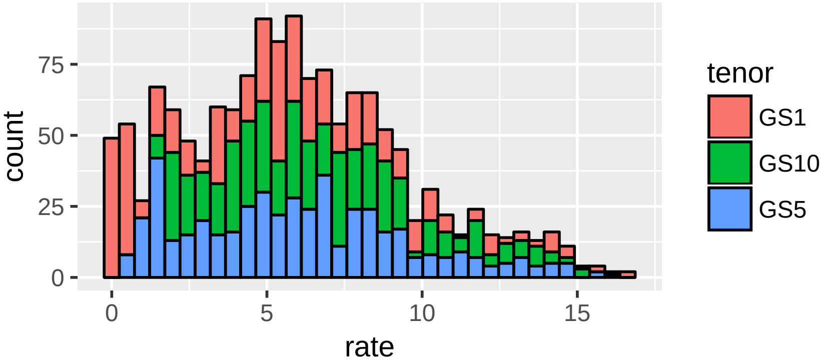


Seeing the outline of our shapes

- Bar outlines?
- `color =` adds a border
  - Remember: `fill =` controls the color inside the shape while `color =` controls the border color of a shape
  - Don't use `aes()` this time
    - Are not varying color

Using the color option to make borders

```
histogram +
    geom_histogram(bins = 35, color = "black")
```



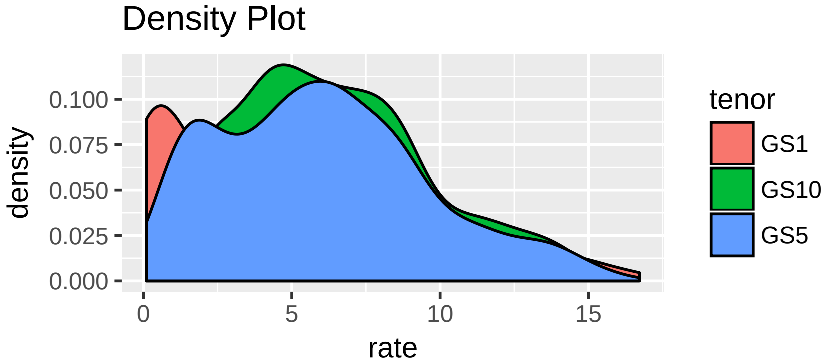
I changed the number of bins to 35 from 25, what effect did this have?

Showing data distributions: Density plots

- The density plot has a y-axis which shows the probability distribution of your data.
- Note that y-axis values greater than 1 are possible but the **area** under the curve must always be 1

Basic Density Plot

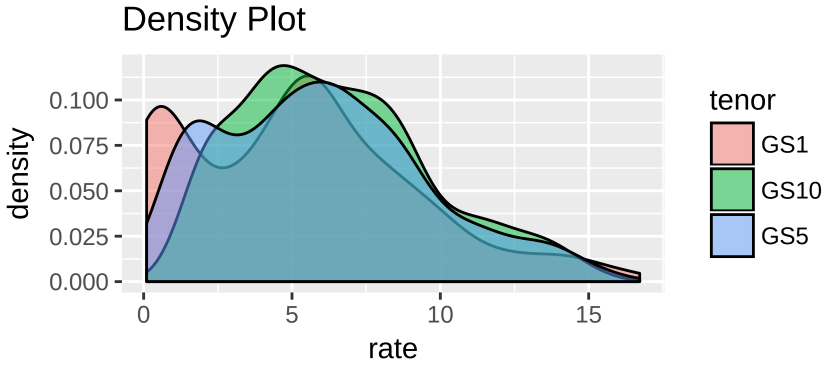
```
density_plot <- ggplot(box_data,
                       aes(x = rate, fill = tenor)) +
    labs(title = "Density Plot")
density_plot + geom_density()
```



Dealing with Overlap: alpha

- Values hover around 5
  - Distribution is skewed
- Shapes overlap
  - Control transparency
    - `alpha` = takes a value between 0 (see through), and 1 (opaque)

```r
density_plot + geom_density(alpha = 0.5)
```

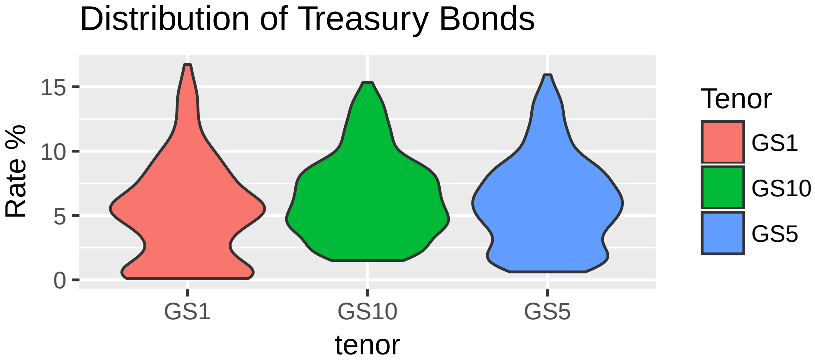


Showing distribution and Density

- Show distribution and spread at once?
- Violin plots!
  - Width shows distribution
  - Length shows spread

Violin Plot

```
violin <- ggplot(box_data, aes(x = tenor, y = rate,
                               fill = tenor)) +
    geom_violin() +
    labs(title = "Distribution of Treasury Bonds",
         fill = "Tenor", y = "Rate %")
violin
```

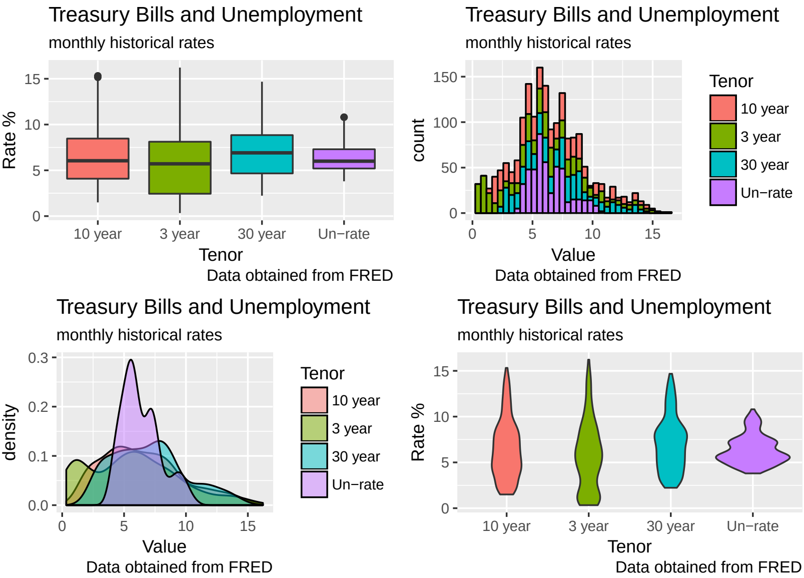


What does it all mean?

- How do the interest rates from our recent past compare with historical averages for rates on bonds of the same tenor?
  - Government is “risk-free”
  - What do low rates mean if you want a loan?
  - What do low rates mean if you are a bank?
  - What do low rates mean for you?

In class exercise: Data distribution

- For the rest of the class work on creating 4 charts like the ones we just went over: boxplot, histogram, density, and violin plots
- Use the data in `distribution_exercise.csv`
- You should look at the following 4 values: GS3, GS10, GS30, and UNRATE
- Use the scale functions
- Your charts should look like the following:
  - You should make yours separately, do not worry about displaying all 4 at once
  - When you are finished save your plots to .png files



Treasury Bills and Unemployment

monthly historical rates

Data obtained from FRED

Treasury Bills and Unemployment

monthly historical rates

Data obtained from FRED

Treasury Bills and Unemployment

monthly historical rates

Data obtained from FRED

Treasury Bills and Unemployment

monthly historical rates

Data obtained from FRED

Does the unemployment rate seem to differ from the Treasury rates?