Week 1 Code & Assignment

REDA1-CE1000 - Week 1
The power of modern, open-source environments Getting Accustomed to R and the R Studio IDE

Creating a notebook chunk - on a mac: 'control' + 'option', then 'i' - on a pc: 'control' + 'alt', then 'i'

Install these packages using install.packages("")

```
r = getOption("repos")
r["CRAN"] = "http://cran.us.r-project.org"
options(repos = r)

install.packages(c("PASWR2", "MASS", "repmis", "latex2exp",
                   "devtools","tidyverse", "stargazer", "quantmod"))
```

```
## Installing packages into '/Users/petermattingly/Library/R/3.5/library'
## (as 'lib' is unspecified)

##
##   There are binary versions available but the source versions are
##   later:
##            binary source needs_compilation
## MASS     7.3-51.5 7.3-53              TRUE
## devtools    2.2.2  2.3.2             FALSE
## quantmod   0.4-16 0.4.17             FALSE
##
##
## The downloaded binary packages are in
##  /var/folders/fz/3h3gq4bj0ks2mm14xx3fl92w0000gn/T//RtmpHLsriW/downloaded_packages

## installing the source packages 'MASS', 'devtools', 'quantmod'
```

Load the libraries

```
library(PASWR2)
```

```
## Loading required package: lattice

## Loading required package: ggplot2
```

```
library(MASS)
library(repmis)
library(latex2exp)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- tidyverse

## v tibble  3.0.3     v purrr   0.3.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------------------------ tidyverse_confli
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```
library(stargazer)
```

```
##
## Please cite as:

##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.

##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last
```

```
## Loading required package: TTR
```

```
## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
devtools::install_github("sboysel/fredr")
```

```
## Skipping install of 'fredr' from a github remote, the SHA1 (97b244ed) has not changed since last ins
##   Use 'force = TRUE' to force installation
```

```
library(fredr)
```

Identify and set the working directory.

```
getwd()
```

```
## [1] "/Users/petermattingly/Desktop/NYU Schack/Fall 2020/Real Estate Data Analytics - November"
```

```
#setwd("/Users/timothysavage/Desktop/REDA")
```

Some example code to get used to

```
set.seed(1492) # Set seed makes results reproducible.
ruv = runif(n = 20, min = 0, max = 1) # Generate a uniform[0, 1] RV with 20 draws.
round(ruv, 4) # Round answers to 4 decimals places.
```

```
##  [1] 0.2776 0.2161 0.1844 0.1105 0.0522 0.0082 0.8527 0.5104 0.3904 0.7691
## [11] 0.6415 0.6386 0.1949 0.5221 0.5216 0.7921 0.1234 0.3437 0.6608 0.9165
```

Basic summary statistics

```
wn = rnorm(1000, mean=0, sd=1) # Sample 1,000 draw from N(0, 1)
```

```
mean(wn)
```

```
## [1] 0.06648864
```

```
var(wn)
```

```
## [1] 1.028402
```

```
summary(wn)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -3.35728 -0.64821  0.10088  0.06649  0.76168  3.08726
```

```r
sqrt(var(wn))
```

```
## [1] 1.014101
```

```r
(7 * 3) + 12/2 - 7^2 + sqrt(4) # R can act as basic calculator.
```

```
## [1] -20
```

Graphing

```r
age = c(1, 3, 5, 2, 11, 9, 3, 9, 12, 3) # Generate some fake data.
weight = c(4.4, 5.3, 7.2, 5.2, 8.5, 7.3, 6, 10.4,
           10.2, 6.1)
```

```r
mean(weight)
```

```
## [1] 7.06
```

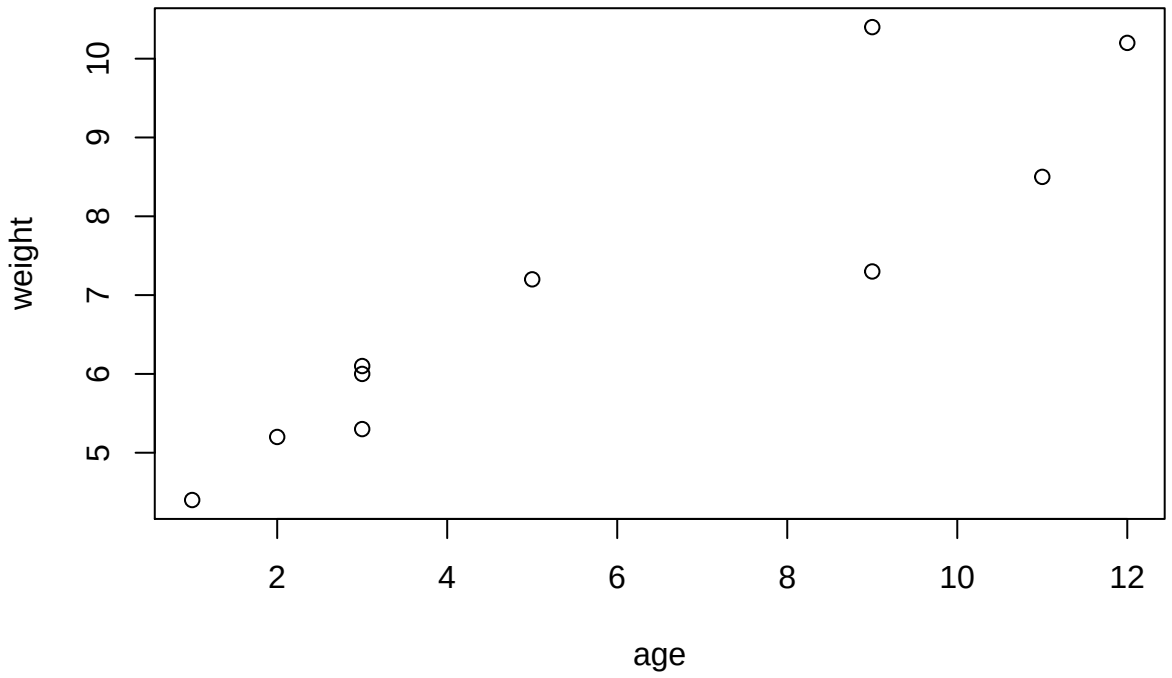
```r
sd(weight)
```

```
## [1] 2.077498
```

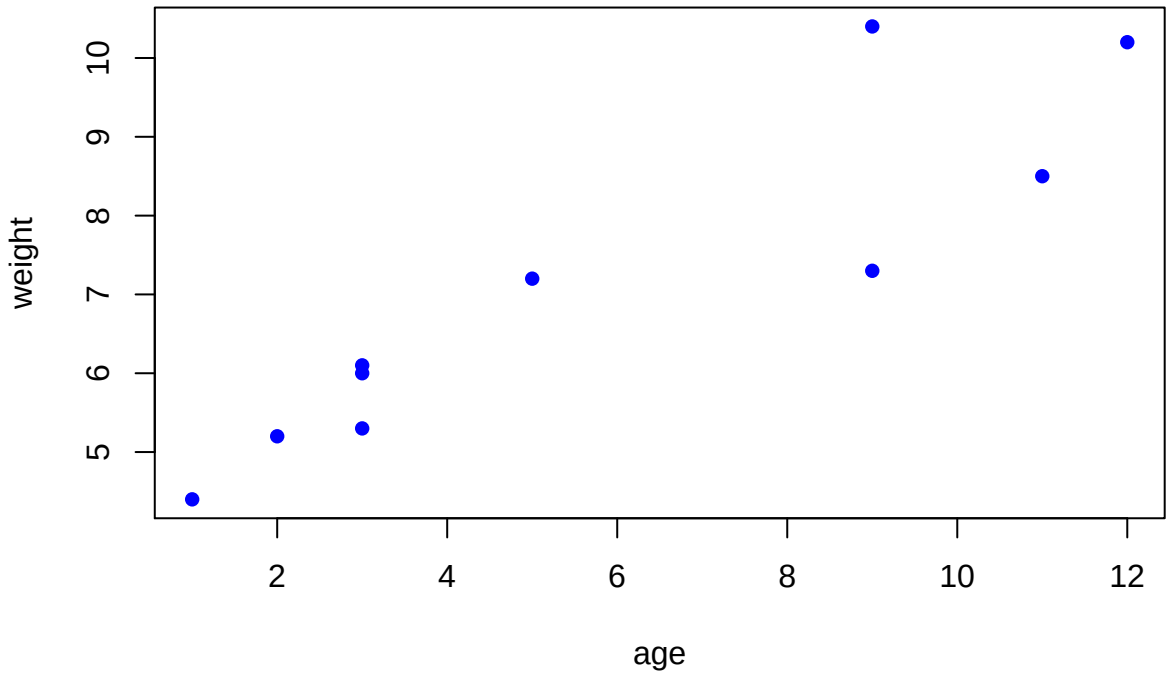
```r
cor(age, weight)
```

```
## [1] 0.9075655
```

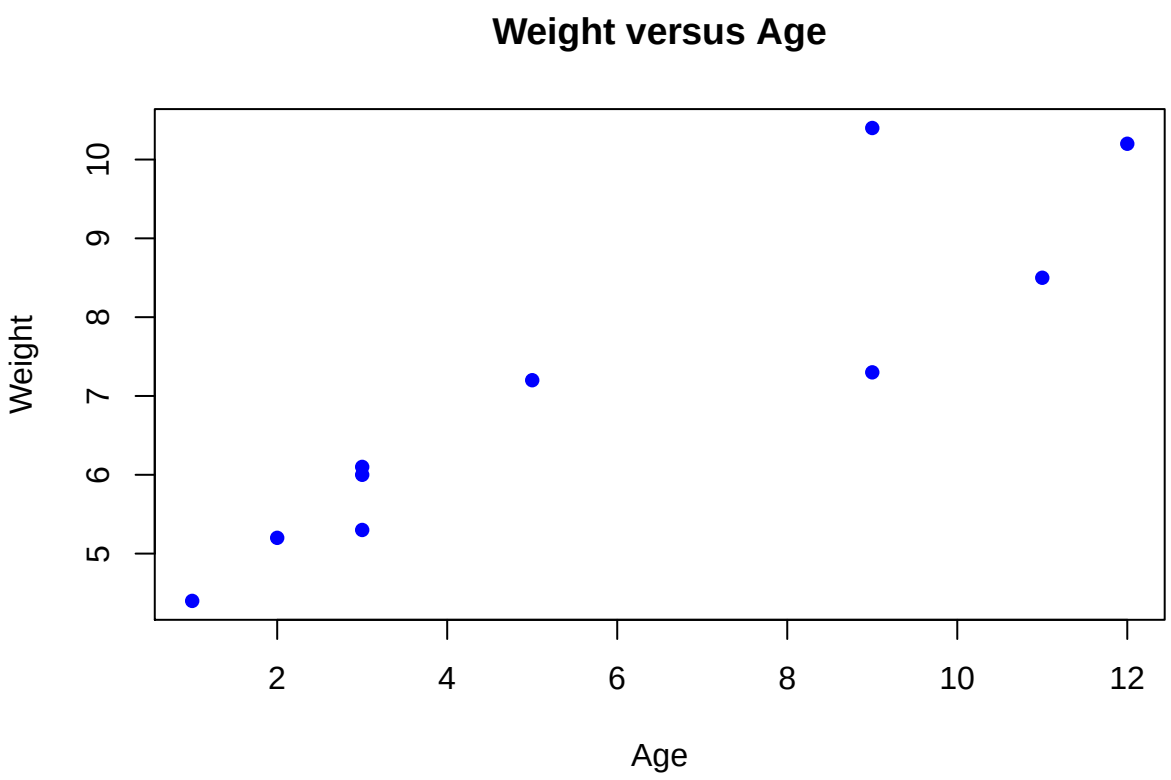
```r
plot(age, weight)
```



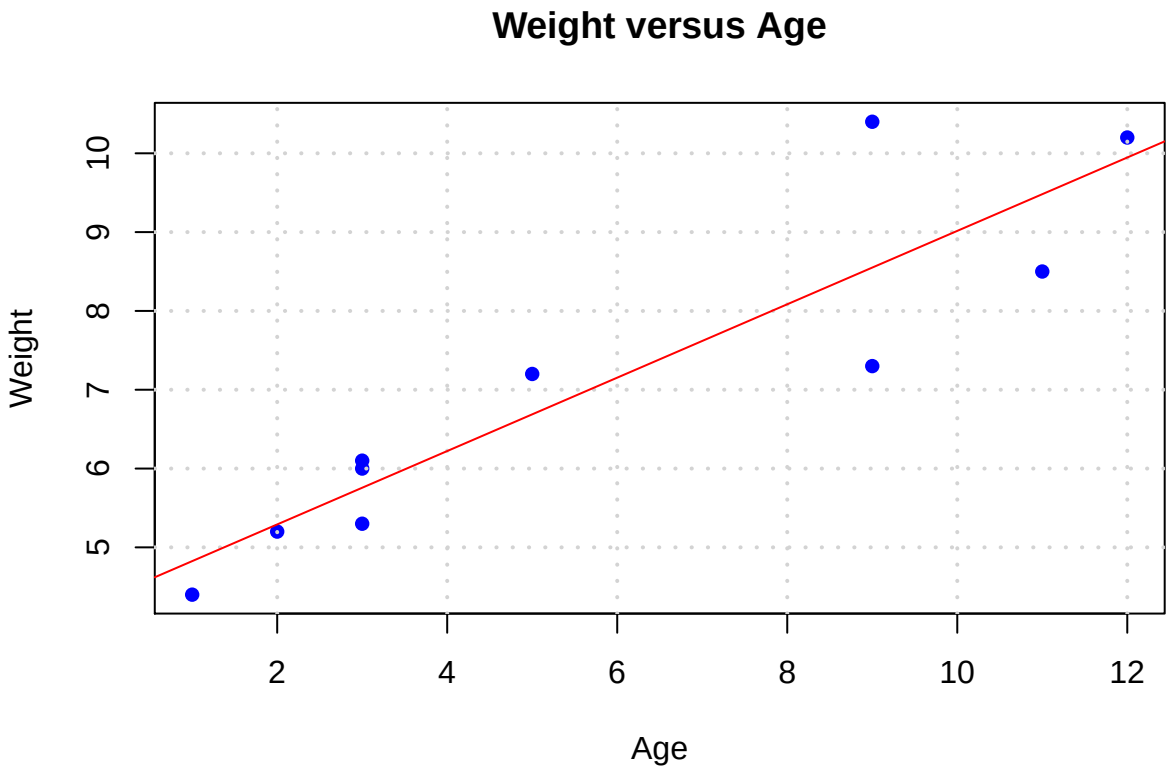
```
plot(age, weight, pch=16, col="blue")
```




```
plot(age, weight, pch=16, col="blue",
     main="Weight versus Age", xlab="Age", ylab="Weight")
```



```r
plot(age, weight, pch=16, col="blue",
     main="Weight versus Age", xlab="Age", ylab="Weight")
grid(lw=2)
abline(lm(weight~age), col="red")
```




```r
x = 1:50 # Generate a list from 1 to 50.
y = x + rnorm(n = 50, mean = 0, sd = 0.5) # Add normal errors.
model = lm(y ~ x) # Regress y on x.
model # Print results
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Coefficients:
## (Intercept)            x
##     -0.2031       1.0086
```

```r
stargazer(model, type="text", title="Random Model", single.row=TRUE,
          ci=TRUE, ci.level=0.95) # Print a nice layout
```

```
##
## Random Model
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                                   y
```

```
## ------------------------------------------------
## x                      1.009*** (1.001, 1.017)
## Constant               -0.203* (-0.436, 0.030)
## ------------------------------------------------
## Observations                    50
## R2                             0.999
## Adjusted R2                    0.999
## Residual Std. Error       0.414 (df = 48)
## F Statistic         61,928.170*** (df = 1; 48)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

R as Excel: The R Dataframe

```
nv = c(1, 3, 6, 8) # Numeric list
cv = c("a", "d", "f", "p") # Character list
lv = c(TRUE, FALSE, FALSE, TRUE) # Logical list
DF1 = data.frame(nv, cv, lv) # Create an R dataframe
head(DF1) # Print out the dataframe.
```

```
##   nv cv    lv
## 1  1  a  TRUE
## 2  3  d FALSE
## 3  6  f FALSE
## 4  8  p  TRUE
```

```
str(DF1) # Describe its contents.
```

```
## 'data.frame':    4 obs. of  3 variables:
##  $ nv: num  1 3 6 8
##  $ cv: Factor w/ 4 levels "a","d","f","p": 1 2 3 4
##  $ lv: logi  TRUE FALSE FALSE TRUE
```

```
DF1$nv # Dollar sign prefix links dataframe to column name.
```

```
## [1] 1 3 6 8
```

```
DF1$cv # Again.
```

```
## [1] a d f p
## Levels: a d f p
```

The power of R, the CRAN Repository, and Library Vignettes

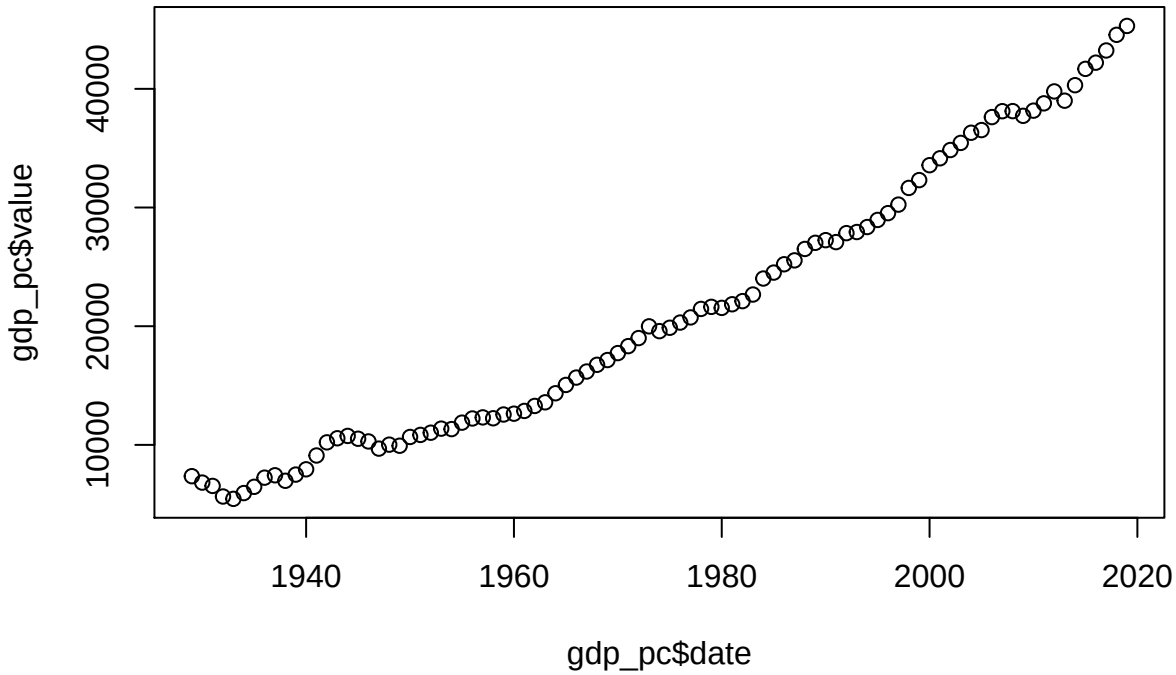
The power of the Application Protocol Interface (API)

```
fredr_set_key('fd7c2810b87f970f3d03b94e5b2ccb26') # My key, please don't abuse.
```

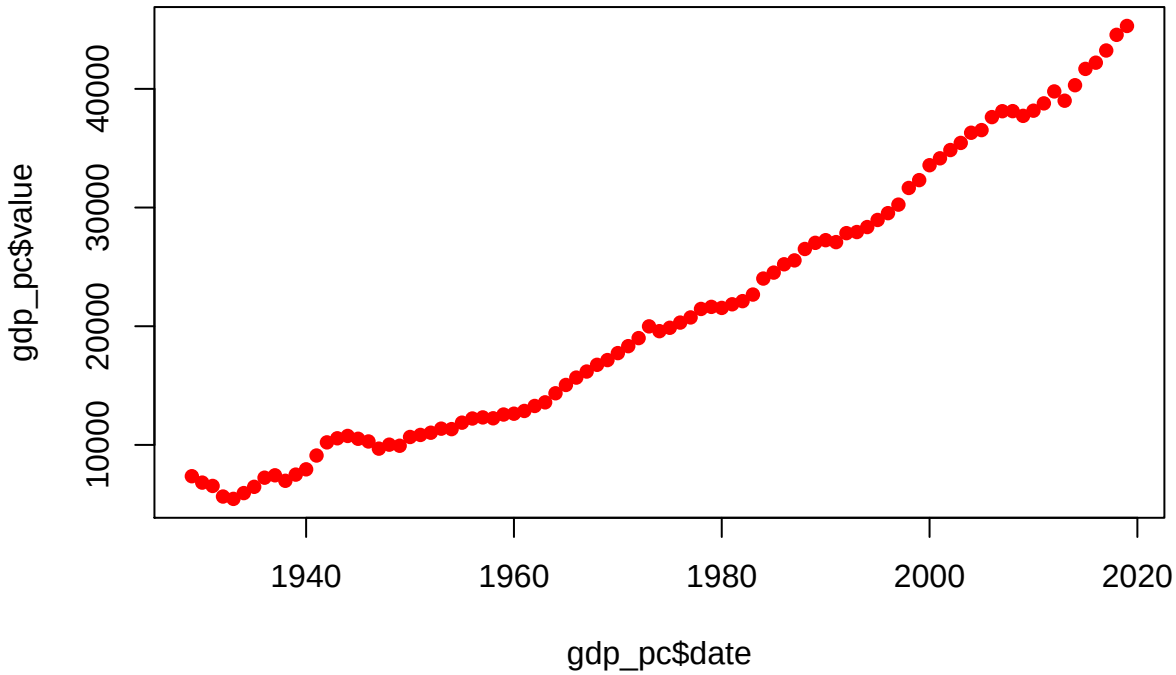
GDP Per Capita

```
gdp_pc = fredr('A229RX0A048NBEA') # Grab GDP per capita
```

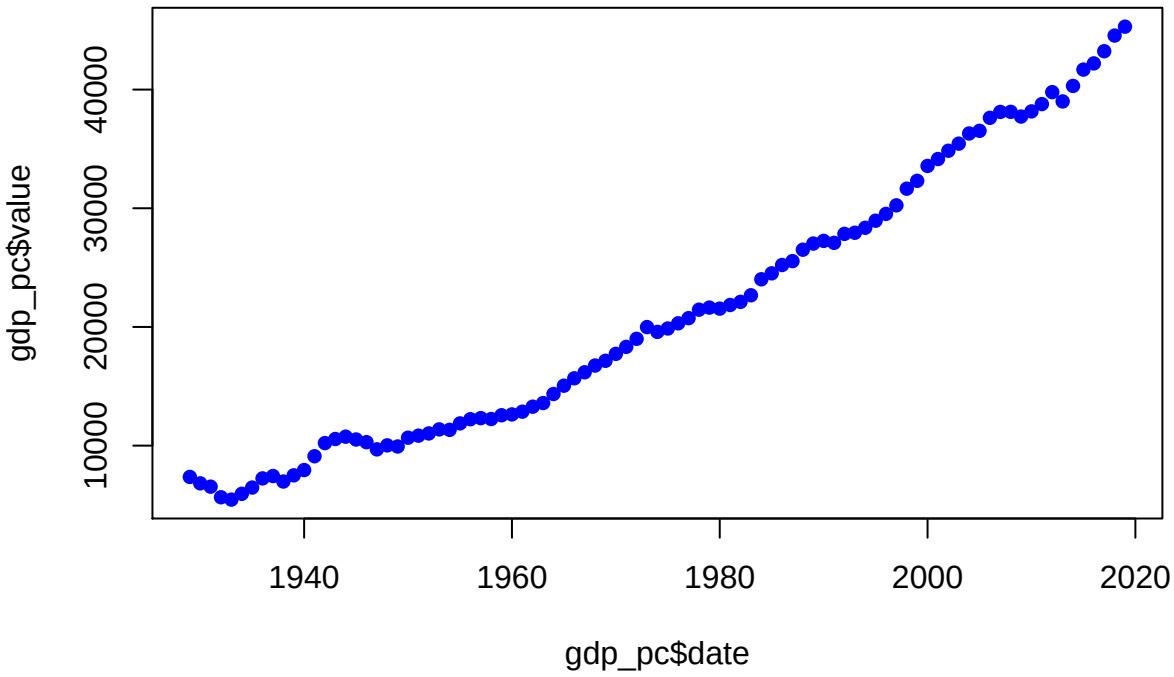
```
plot(gdp_pc$date, gdp_pc$value) # Basic graphs in R.
```



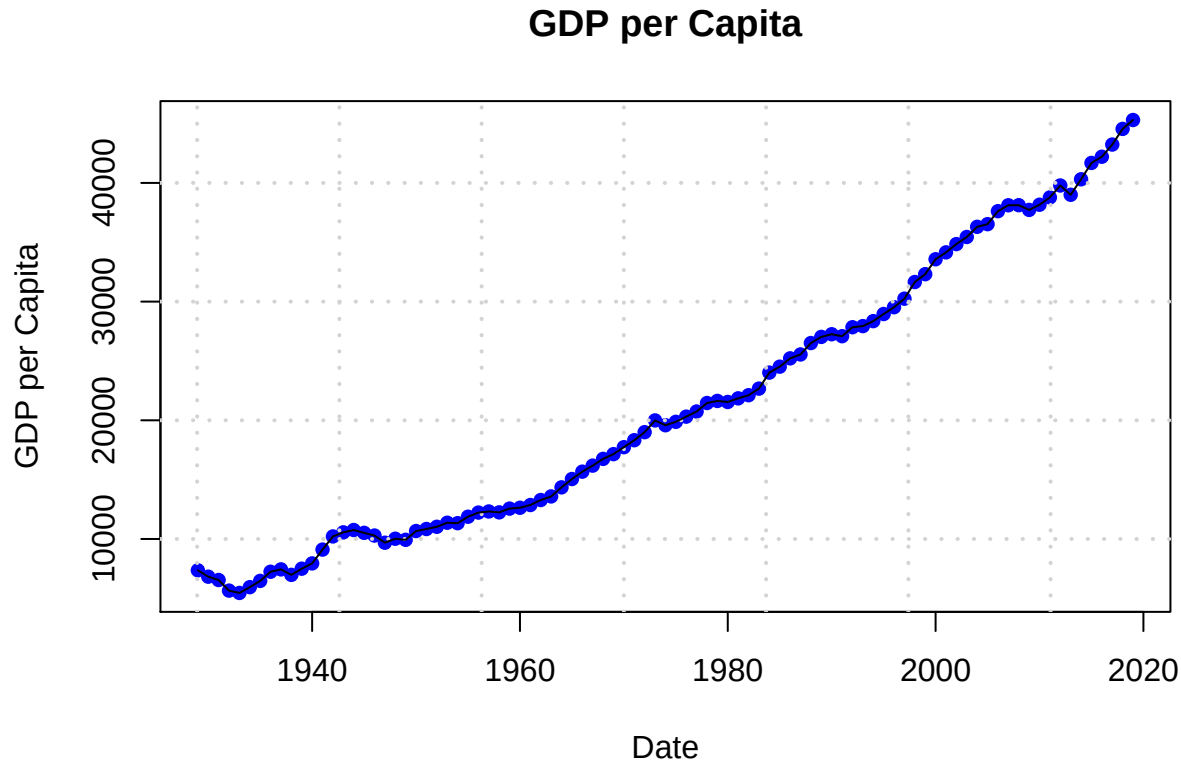

```
plot(gdp_pc$date, gdp_pc$value, col = 'red', pch=16) # In red.
```




```
plot(gdp_pc$date, gdp_pc$value, col = 'blue', pch=16) # In blue.
```



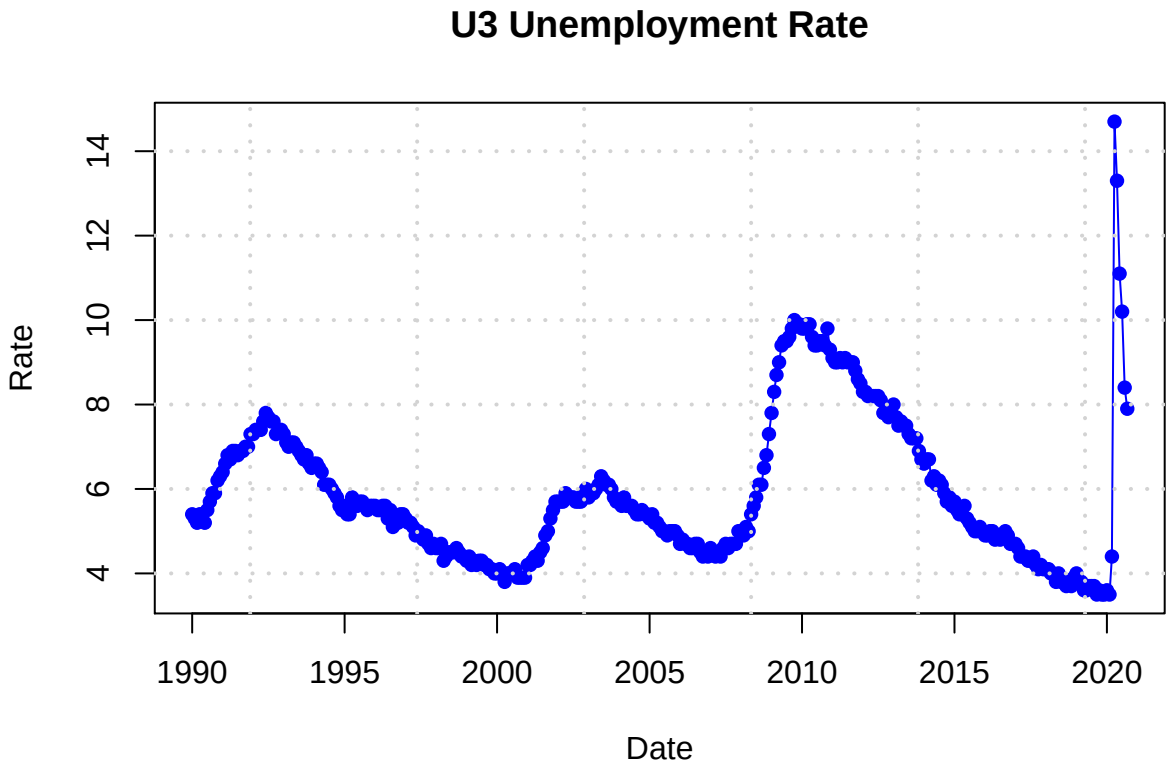

```
plot(gdp_pc$date, gdp_pc$value, col = 'blue', pch=16, xlab="Date", ylab="GDP per Capita",
     main="GDP per Capita")
grid(lw=2)
lines(gdp_pc$date, gdp_pc$value) # As good as Excel.
```



Unemployment Rate

```
unrate = fredr(series_id = "UNRATE", observation_start = as.Date("1990-01-01"))
```

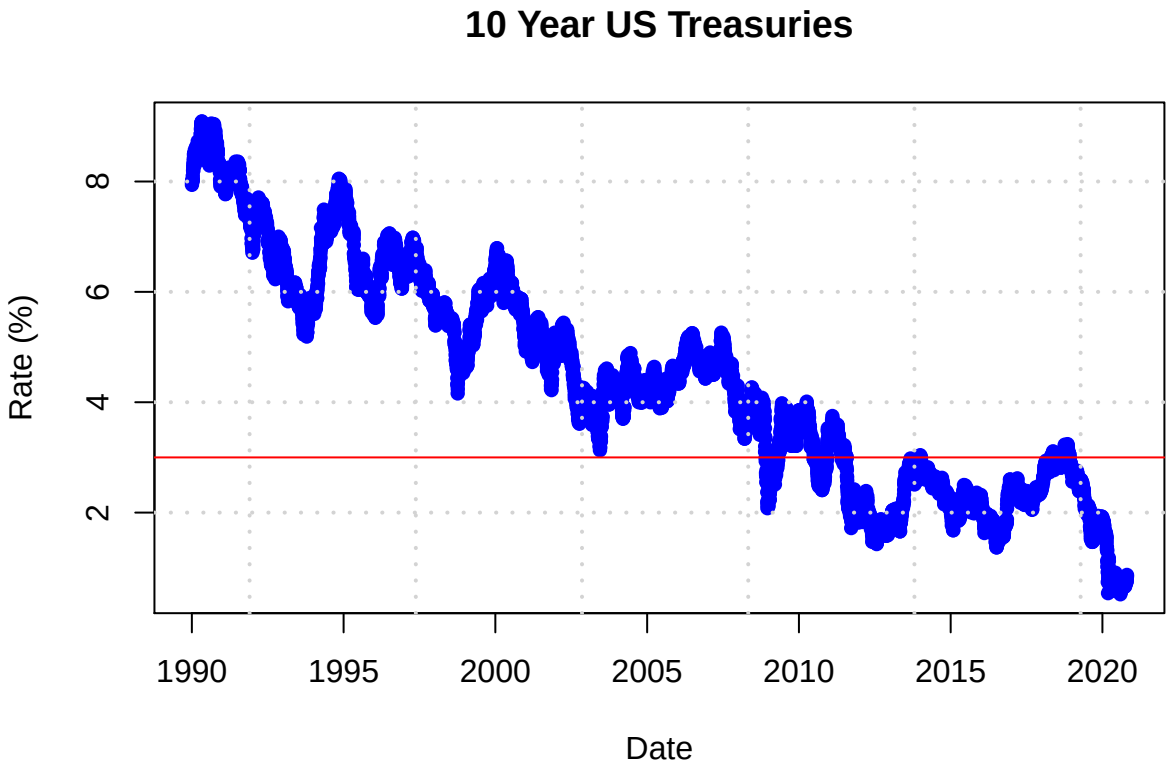
```
plot(unrate$date, unrate$value, col = 'blue', pch=16, ylab = "Rate", xlab = "Date",
     main="U3 Unemployment Rate")
lines(unrate$date, unrate$value, col = 'blue')
grid(lw=2)
```



10 Year US Treasuries

```
tenyear = fredr(series_id = "DGS10", observation_start = as.Date("1990-01-01"))
```

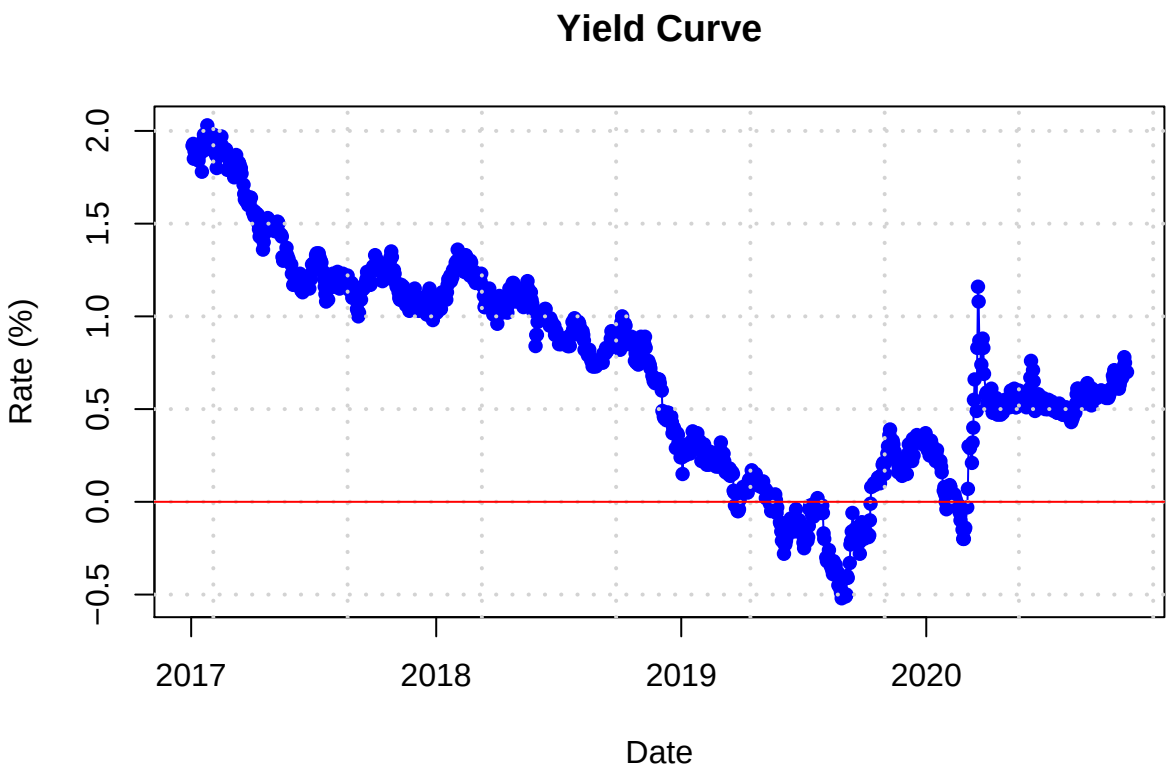
```
plot(tenyear$date, tenyear$value, col = 'blue', pch=16, ylab = "Rate (%)",
     xlab = "Date", main="10 Year US Treasuries")
lines(tenyear$date, tenyear$value, col = 'blue')
grid(lw=2)
abline(h=3, col='red')
```



Yield Curve

```
yieldcurve = fredr(series_id = "T10Y3M", observation_start = as.Date("2017-01-01"))
```

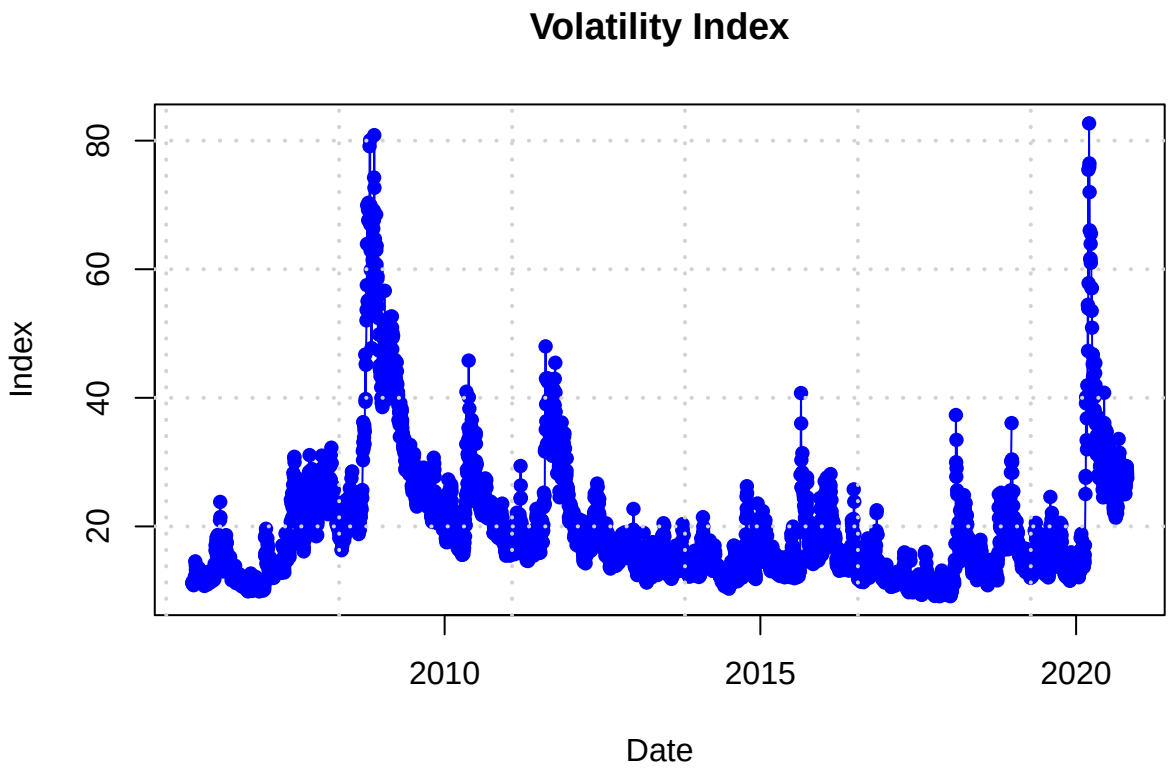
```
plot(yieldcurve$date, yieldcurve$value, col = 'blue', pch=16, ylab = "Rate (%)",
     xlab = "Date", main="Yield Curve")
lines(yieldcurve$date, yieldcurve$value, col = 'blue')
grid(lw=2)
abline(h=0, col='red')
```



Volatility Index

```
vix = fredr(series_id = "VIXCLS", observation_start = as.Date("2006-01-01"))
```

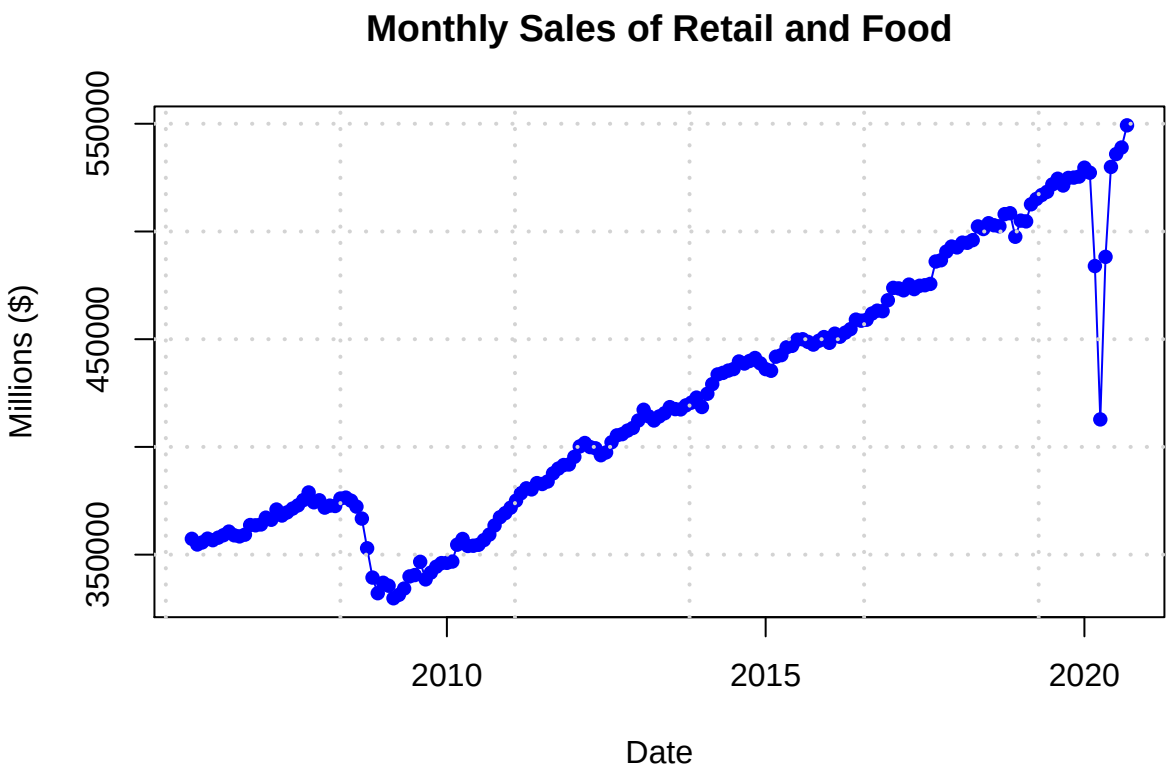
```
plot(vix$date, vix$value, col = 'blue', pch=16, ylab = "Index",
     xlab = "Date", main="Volatility Index")
lines(vix$date, vix$value, col = 'blue')
grid(lw=2)
```



Retail/Food Sales

```
sales = fredr(series_id = "RSAFS", observation_start = as.Date("2006-01-01"))
```

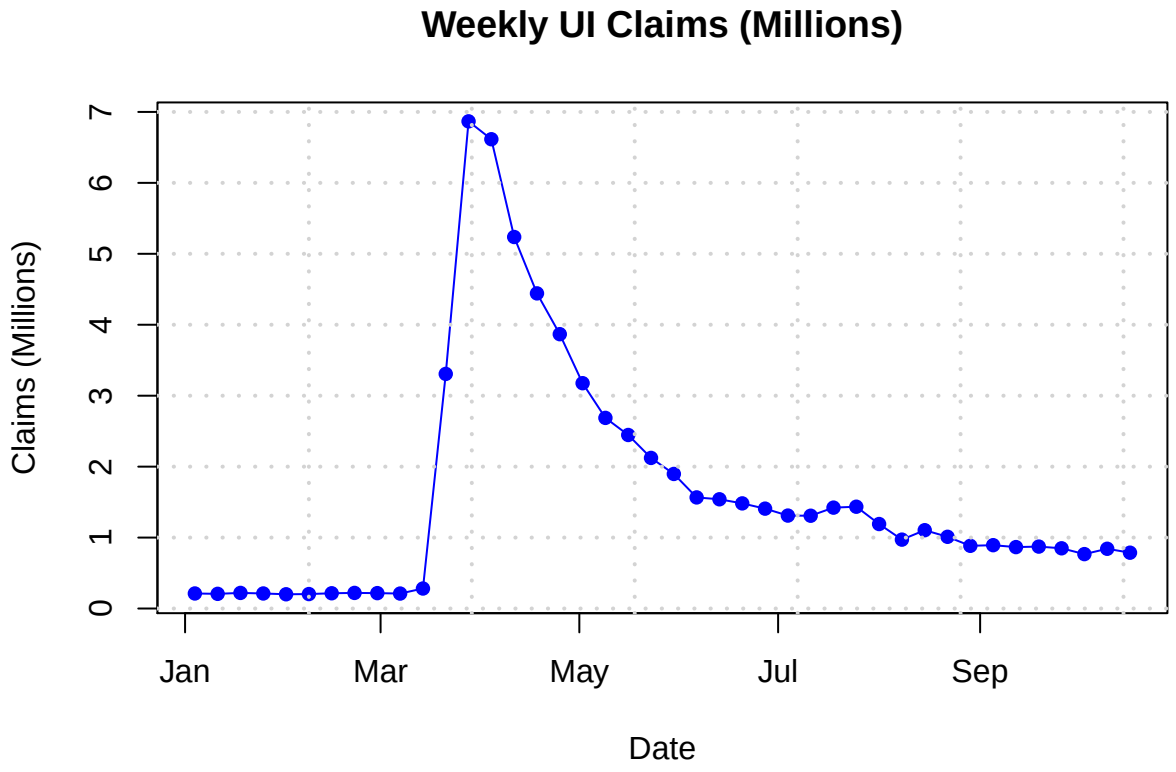
```
plot(sales$date, sales$value, col = 'blue', pch=16, ylab = "Millions ($)",
     xlab = "Date", main="Monthly Sales of Retail and Food")
lines(sales$date, sales$value, col = 'blue')
grid(lw=2)
```



Unemployment Insurance Claims

```
claims = fredr(series_id = "ICSA", observation_start = as.Date("2020-01-01"))
claims$value = claims$value / 1000000 # create new variable
```

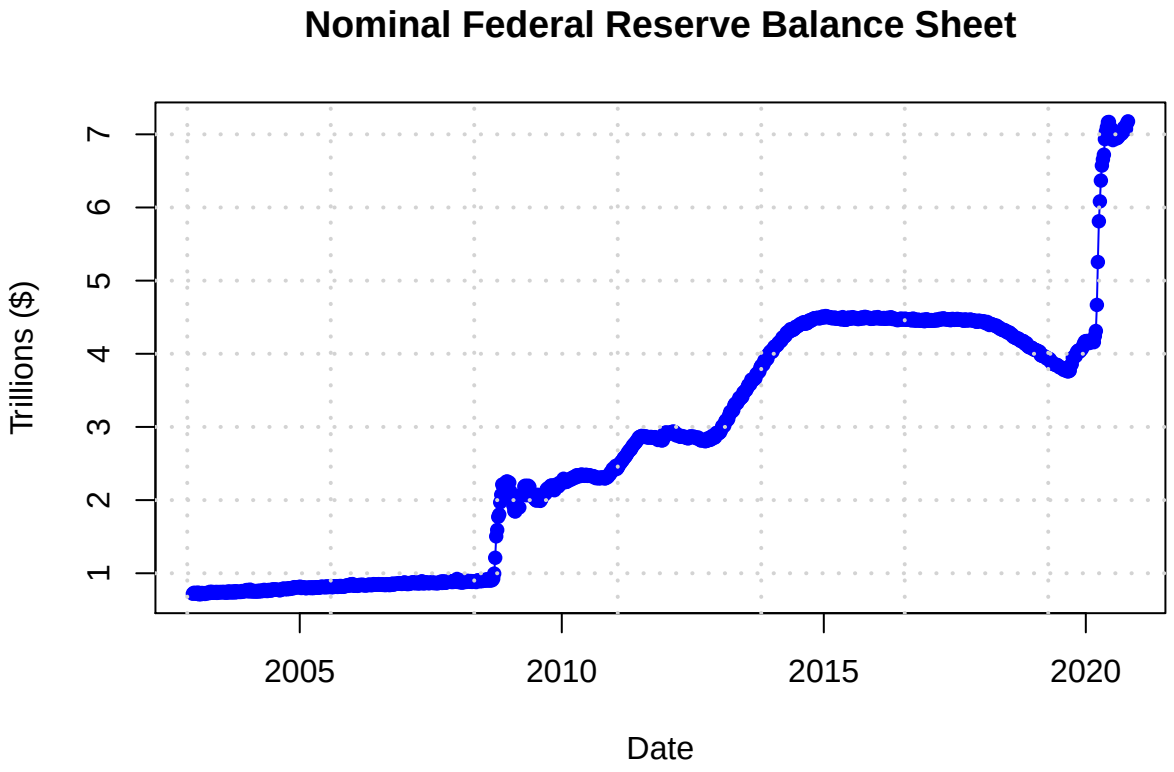
```
plot(claims$date, claims$value, col = 'blue', pch=16, ylab = "Claims (Millions)",
     xlab = "Date", main="Weekly UI Claims (Millions)")
lines(claims$date, claims$value, col = 'blue')
grid(lw=2)
```



Federal Reserve Balance

```
balance_sheet = fredr(series_id = "WALCL", observation_start = as.Date("2000-01-01"))
balance_sheet$value = balance_sheet$value / 1000000
```

```
plot(balance_sheet$date, balance_sheet$value, col = 'blue', pch=16, ylab = "Trillions ($)",
     xlab = "Date", main="Nominal Federal Reserve Balance Sheet")
lines(balance_sheet$date, balance_sheet$value, col = 'blue')
grid(lw=2)
```



Searching for the most popular series

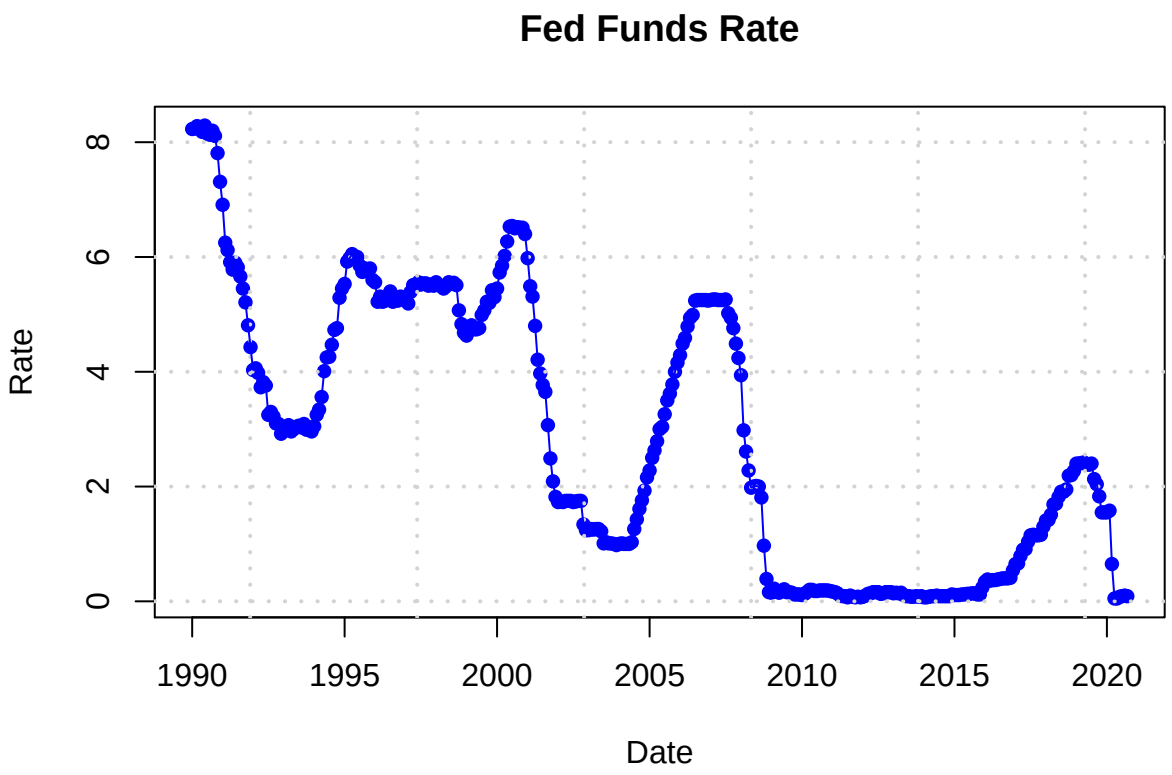
```
fredr_series_search_text(
  search_text = "federal funds",
  order_by = "popularity",
  sort_order = "desc",
  limit = 1) %>%
  pull(id) %>%
  fredr(series_id = .)
```

```
## # A tibble: 795 x 3
##    date       series_id value
##    <date>     <chr>     <dbl>
##  1 1954-07-01 FEDFUNDS   0.8
##  2 1954-08-01 FEDFUNDS   1.22
##  3 1954-09-01 FEDFUNDS   1.07
##  4 1954-10-01 FEDFUNDS   0.85
##  5 1954-11-01 FEDFUNDS   0.83
##  6 1954-12-01 FEDFUNDS   1.28
##  7 1955-01-01 FEDFUNDS   1.39
##  8 1955-02-01 FEDFUNDS   1.29
##  9 1955-03-01 FEDFUNDS   1.35
## 10 1955-04-01 FEDFUNDS   1.43
## # ... with 785 more rows
```

Federal Funds Rate

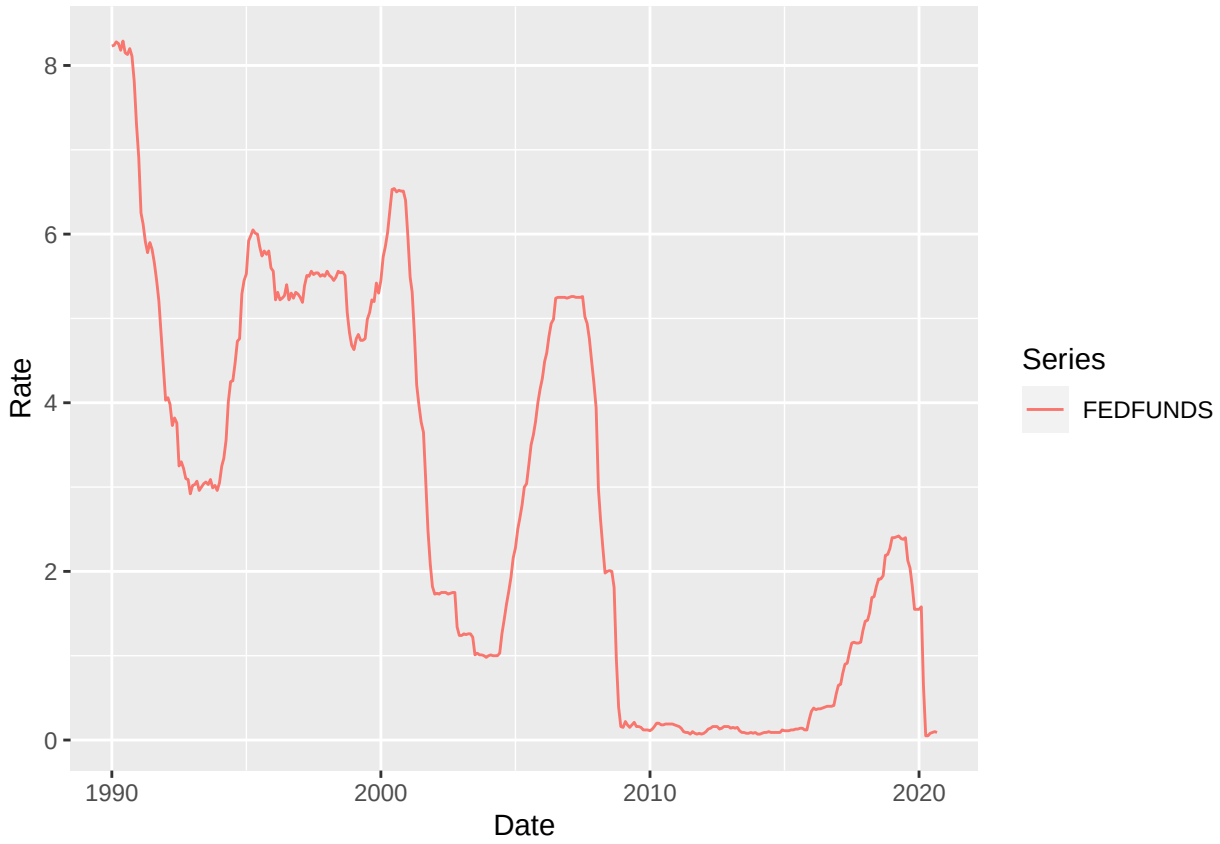
```
fedfunds = fredr(series_id = "FEDFUNDS", observation_start = as.Date("1990-01-01"))
```

```
plot(fedfunds$date, fedfunds$value, col = 'blue', pch=16, ylab = "Rate", xlab = "Date",
     main="Fed Funds Rate")
grid(lw=2)
lines(fedfunds$date, fedfunds$value, col = 'blue')
```



Grammar of Graphics, or ggplot

```
funds_graph <- ggplot(data = fedfunds, mapping = aes(x = date, y = value, color = series_id)) +
  geom_line() + labs(x = "Date", y = "Rate", color = "Series")
ggsave("funds_graph.png", funds_graph, width = 7, height = 5, device = "png")
funds_graph
```



Data scrapping

```
griliches = read.csv("https://vincentarelbundock.github.io/Rdatasets/csv/Ecdat/Griliches.csv")
```

```
str(griliches)
```

```
## 'data.frame':    758 obs. of  21 variables:
##  $ X       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ rns     : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ rns80   : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ mrt     : Factor w/ 2 levels "no","yes": 1 1 1 1 2 1 2 2 2 2 ...
##  $ mrt80   : Factor w/ 2 levels "no","yes": 2 2 2 2 2 1 2 2 2 2 ...
##  $ smsa    : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 1 2 1 ...
##  $ smsa80  : Factor w/ 2 levels "no","yes": 2 2 2 2 2 2 2 1 2 1 ...
##  $ med     : int  8 14 14 12 6 8 8 14 12 13 ...
##  $ iq      : int  93 119 108 96 74 91 114 111 95 132 ...
##  $ kww     : int  35 41 46 32 27 24 50 37 44 44 ...
##  $ year    : int  68 66 67 66 73 66 73 67 66 73 ...
##  $ age     : int  19 23 20 18 26 16 30 23 22 30 ...
##  $ age80   : int  31 37 33 32 34 30 38 36 36 38 ...
##  $ school  : int  12 16 14 12 9 9 18 15 12 18 ...
##  $ school80: int  12 18 14 12 11 10 18 15 12 18 ...
##  $ expr    : num  0.462 0 0.423 0.333 9.013 ...
##  $ expr80  : num  10.6 11.4 11 13.1 14.4 ...
##  $ tenure  : int  0 2 1 1 3 1 6 1 2 5 ...
##  $ tenure80: int  2 16 9 7 5 0 14 1 16 13 ...
```

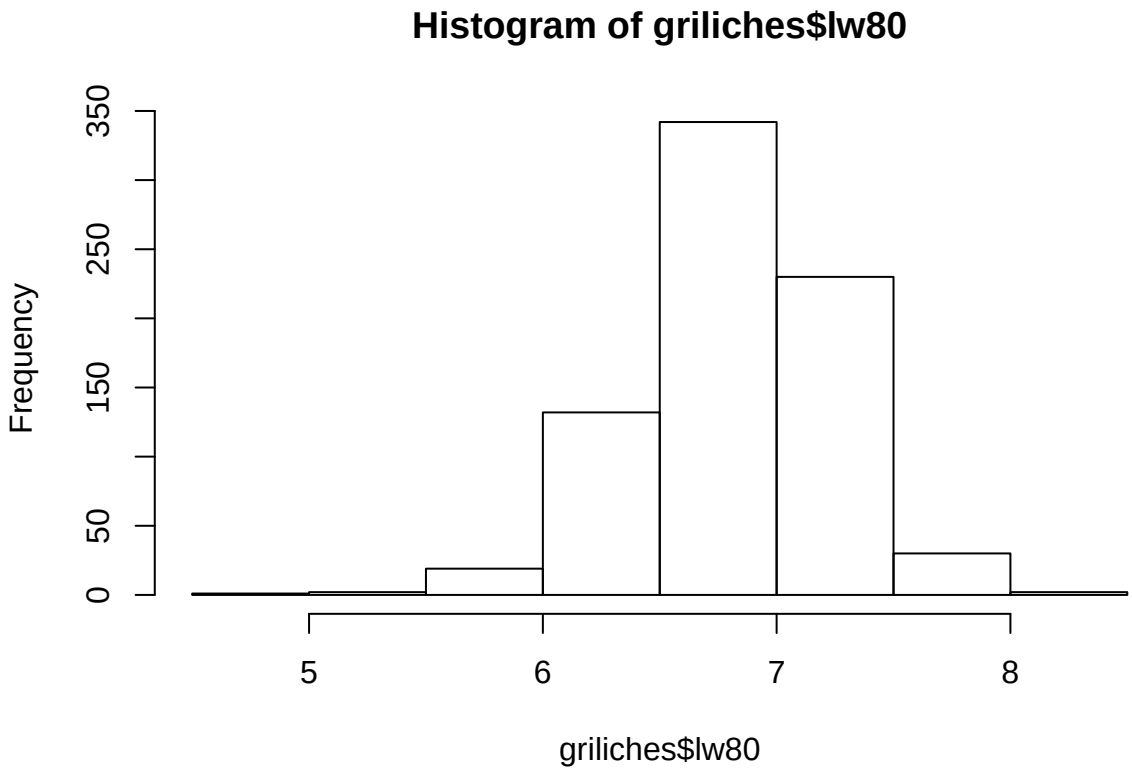
```
##  $ lw      : num  5.9 5.44 5.71 5.48 5.93 ...
##  $ lw80    : num  6.64 6.69 6.71 6.48 6.33 ...
```

```
summary(griliches)
```

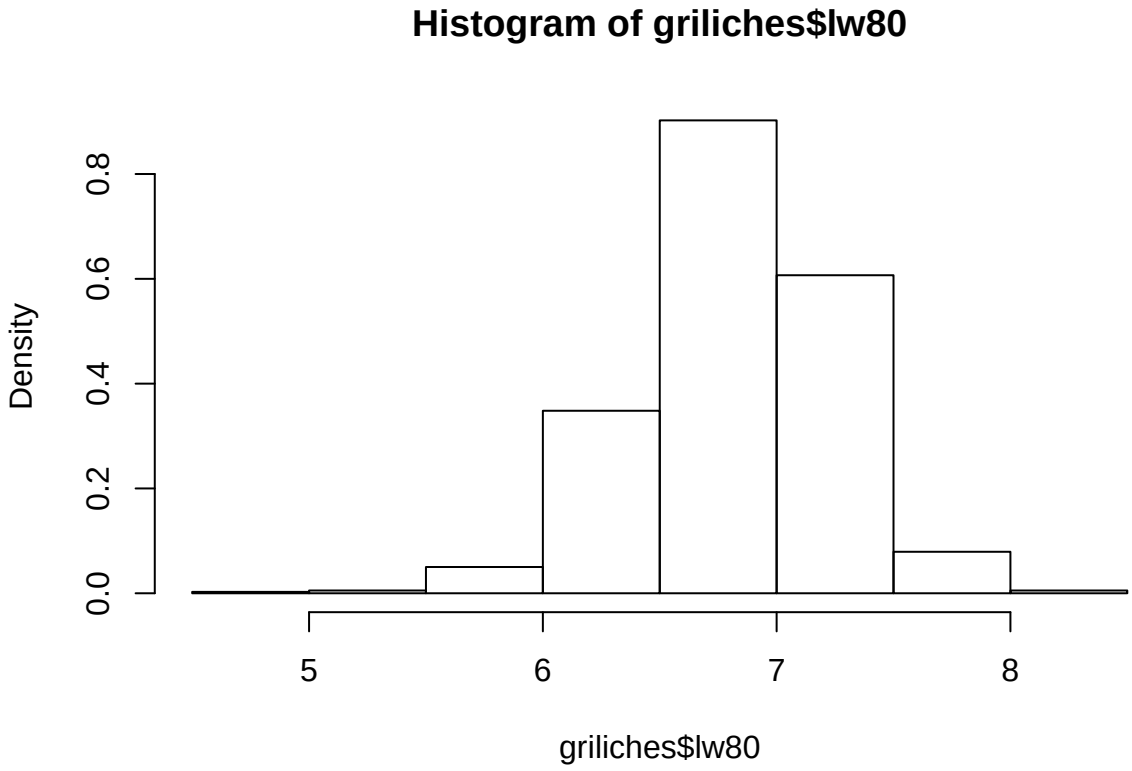
```
##        X          rns      rns80      mrt      mrt80      smsa
##  Min.   :  1.0   no :554   no :536   no :368   no : 77   no :224
##  1st Qu.:190.2   yes:204   yes:222   yes:390   yes:681   yes:534
##  Median :379.5
##  Mean   :379.5
##  3rd Qu.:568.8
##  Max.   :758.0
##  smsa80         med              iq              kww
##  no :218   Min.   : 0.00   Min.   : 54.00   Min.   :12.00
##  yes:540   1st Qu.: 9.00   1st Qu.: 95.25   1st Qu.:32.00
##            Median :12.00   Median :104.00   Median :37.00
##            Mean   :10.91   Mean   :103.86   Mean   :36.57
##            3rd Qu.:12.00   3rd Qu.:113.75   3rd Qu.:41.00
##            Max.   :18.00   Max.   :145.00   Max.   :56.00
##       year            age            age80           school
##  Min.   :66.00   Min.   :16.00   Min.   :28.00   Min.   : 9.00
##  1st Qu.:66.00   1st Qu.:20.00   1st Qu.:30.00   1st Qu.:12.00
##  Median :69.00   Median :22.00   Median :33.00   Median :12.00
##  Mean   :69.03   Mean   :21.84   Mean   :33.01   Mean   :13.41
##  3rd Qu.:71.00   3rd Qu.:24.00   3rd Qu.:36.00   3rd Qu.:16.00
##  Max.   :73.00   Max.   :30.00   Max.   :38.00   Max.   :18.00
##     school80          expr            expr80          tenure
##  Min.   : 9.00   Min.   : 0.0000   Min.   : 0.692   Min.   : 0.000
##  1st Qu.:12.00   1st Qu.: 0.2815   1st Qu.: 8.388   1st Qu.: 1.000
##  Median :13.00   Median : 0.9600   Median :11.059   Median : 1.000
##  Mean   :13.71   Mean   : 1.7354   Mean   :11.394   Mean   : 1.831
##  3rd Qu.:16.00   3rd Qu.: 2.4400   3rd Qu.:14.671   3rd Qu.: 2.000
##  Max.   :18.00   Max.   :11.4440   Max.   :22.045   Max.   :10.000
##     tenure80           lw              lw80
##  Min.   : 0.000   Min.   :4.605   Min.   :4.749
##  1st Qu.: 3.000   1st Qu.:5.380   1st Qu.:6.571
##  Median : 7.000   Median :5.684   Median :6.854
##  Mean   : 7.363   Mean   :5.687   Mean   :6.827
##  3rd Qu.:11.000   3rd Qu.:5.991   3rd Qu.:7.092
##  Max.   :22.000   Max.   :7.051   Max.   :8.032
```

Histograms

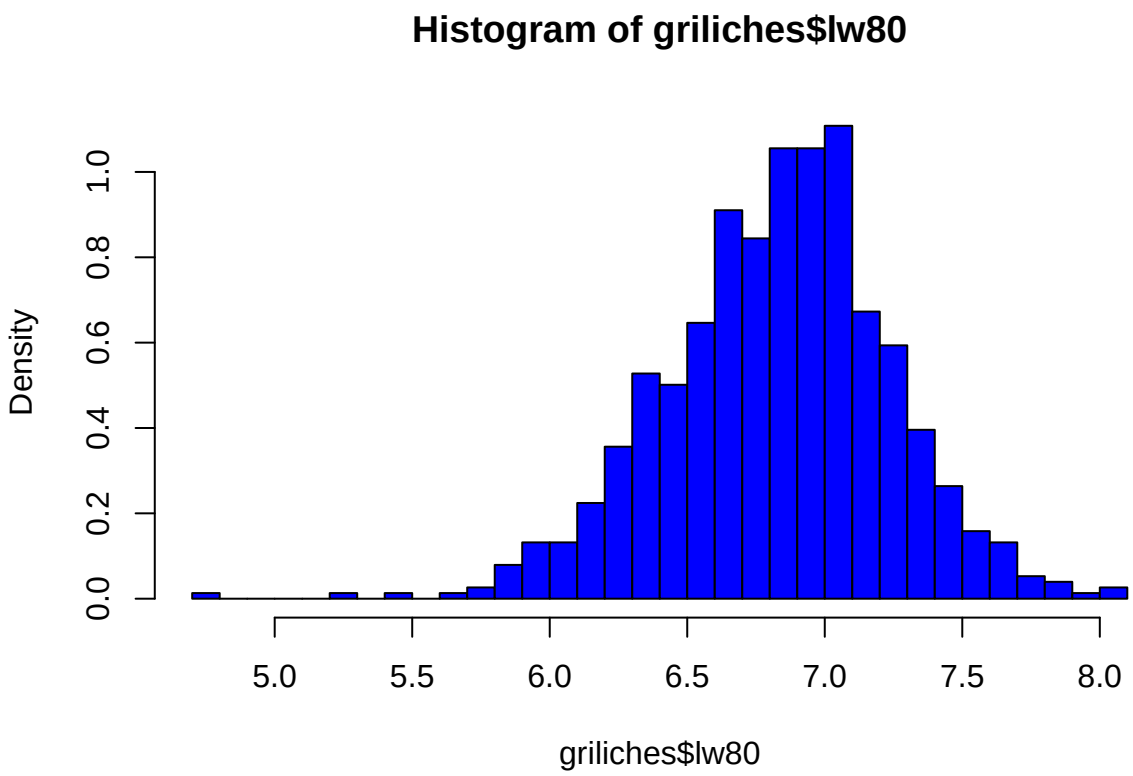
```
hist(griliches$lw80) # Histograms
```



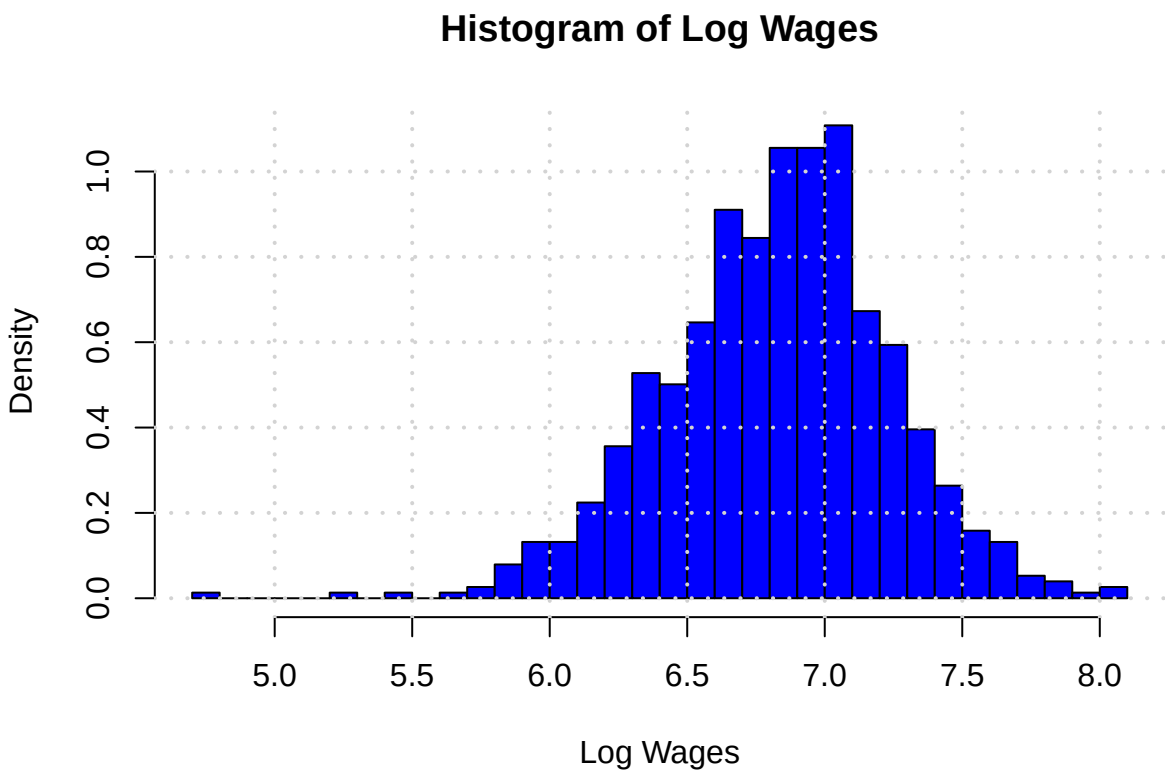

```
hist(griliches$lw80, freq = F)
```



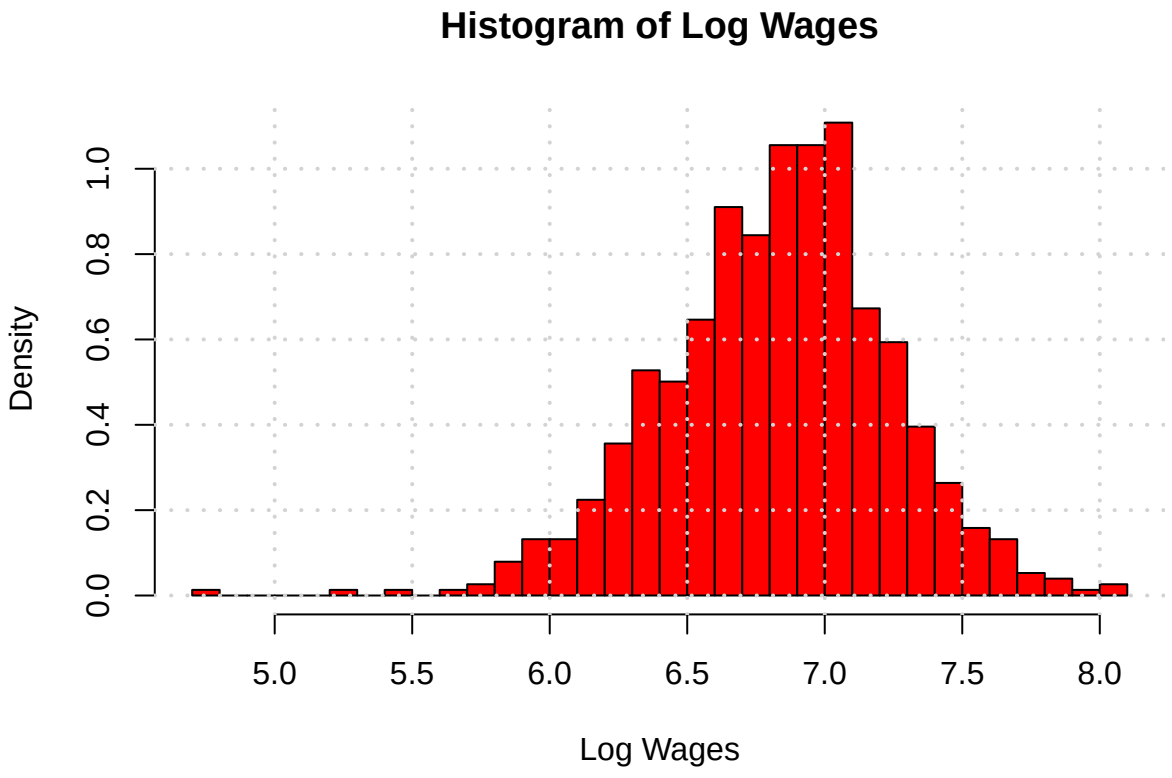
```
hist(griliches$lw80, freq = F, breaks=40, col = 'blue')
```




```
hist(griliches$lw80, freq = F, breaks=40, col = 'blue',
     main='Histogram of Log Wages', xlab='Log Wages')
grid(lw=2)
```



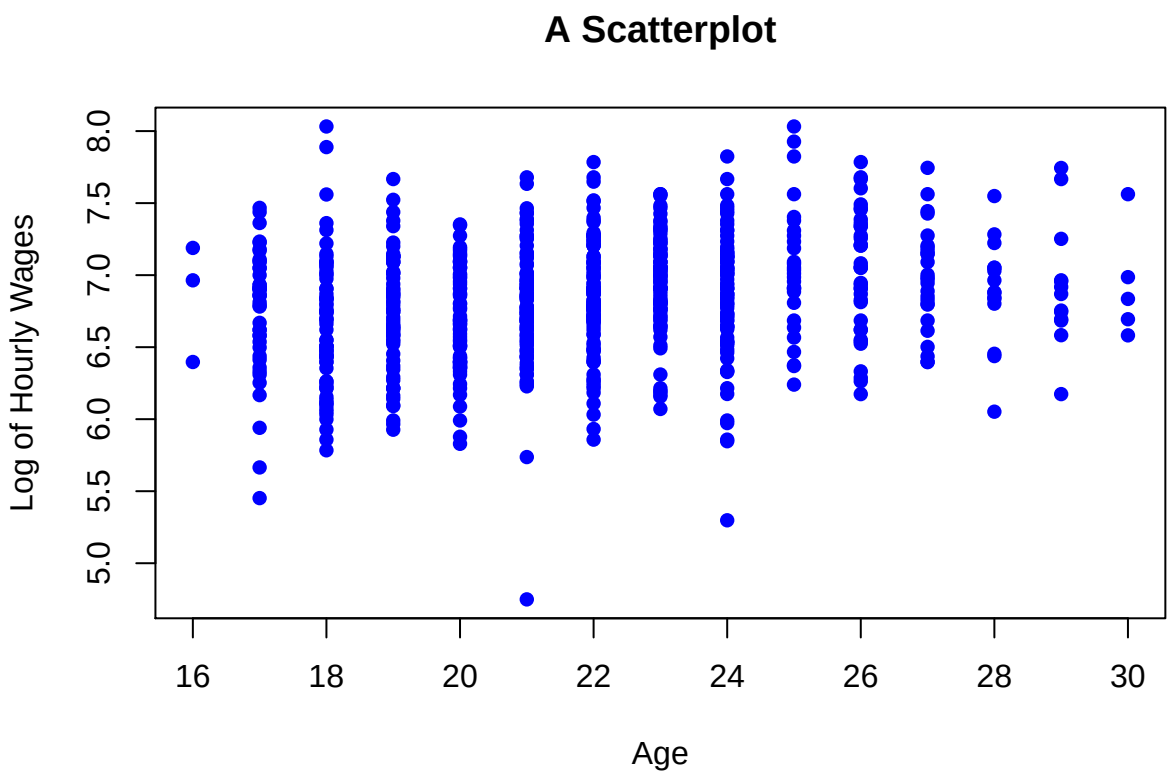

```
hist(griliches$lw80, freq = F, breaks=40, col = 'red',
     main='Histogram of Log Wages', xlab='Log Wages')
grid(lw=2)
```



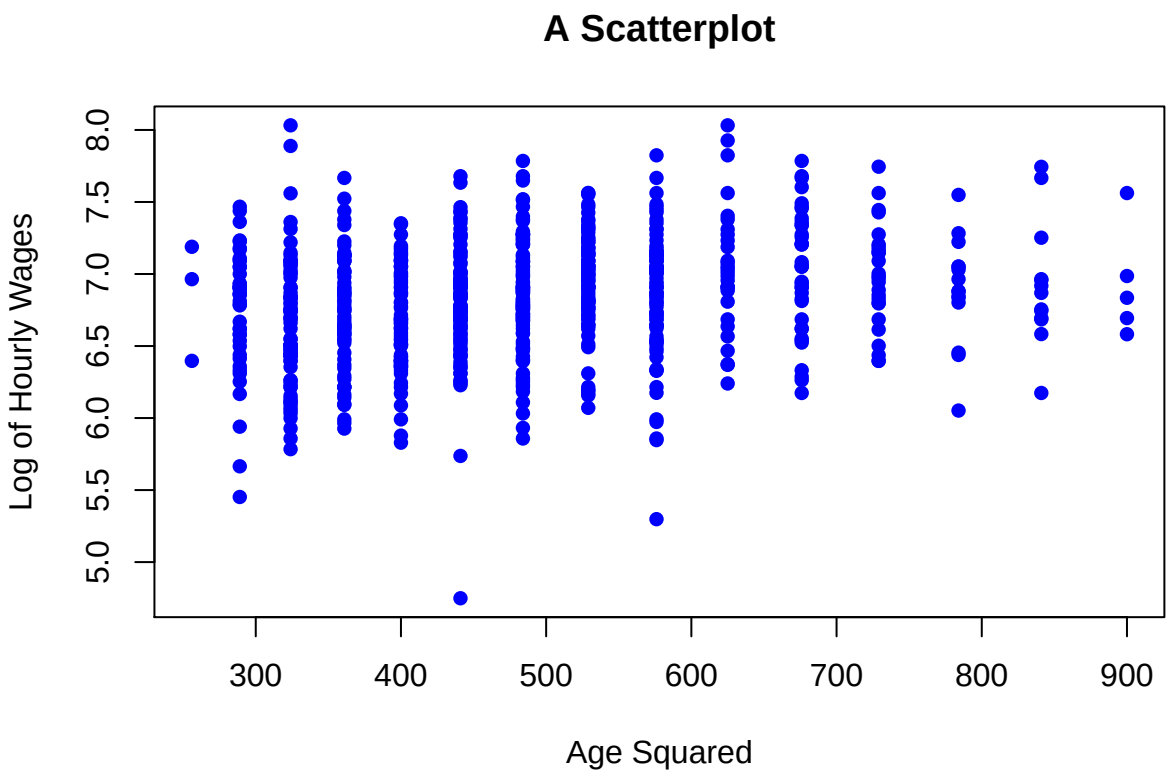
Data Manipulation

```
griliches$age2 = (griliches$age)^2 # Generate an additional variable, the square of age.
```

```
plot(griliches$age, griliches$lw80, col='blue', pch=16, xlab="Age",
     ylab="Log of Hourly Wages", main = "A Scatterplot")
```




```
plot(griliches$age2, griliches$lw80, col='blue', pch=16,
     xlab="Age Squared", ylab="Log of Hourly Wages", main = "A Scatterplot")
```



NY Census

```
#url = "http://www2.census.gov/geo/docs/maps-data/data/gazetteer/census_tracts_list_36.txt"
#data = read.csv(url, header=TRUE, sep='\t')
```

```
#names = c('usps', 'geo', 'pop', 'hu', 'land', 'water', 'landSqmi', 'waterSqmi', 'lat', 'long')
#colnames(data) = names
```

```
#plot(data$long, data$lat, pch=16, col="blue",
#     main="The Empire State by Census Centroid", xlab="Longitude", ylab="Latitude")
#grid(lw=2)
```

Scrapping curated data

```
d = read.csv("https://stats.idre.ucla.edu/stat/data/hsbraw.csv")
head(d)
```

```
##    id female    ses schtyp     prog read write math science socst
## 1  45 female    low public vocation   34    35   41      29    26
## 2 108   male middle public  general   34    33   41      36    36
## 3  15   male   high public vocation   39    39   44      26    42
## 4  67   male    low public vocation   37    37   42      33    32
## 5 153   male middle public vocation   39    31   40      39    51
## 6  51 female   high public  general   42    36   42      31    39
##         honors awards cid
## 1 not enrolled      0   1
## 2 not enrolled      0   1
## 3 not enrolled      0   1
```

```
## 4 not enrolled      0   1
## 5 not enrolled      0   1
## 6 not enrolled      0   1
```

Summary Statistics with dplyr

```
d_summary <- d %>%
  summarise(read_average = mean(read),
         read_median = median(read),
         read_var = var(read),
         read_sd = sd(read))
head(d_summary)
```

```
##   read_average read_median read_var  read_sd
## 1        52.23          50 105.1227 10.25294
```

```
d_gender_summary <- d %>%
  group_by(ses) %>%
  summarise(read_average = mean(read),
         read_median = median(read),
         read_var = var(read),
         read_sd = sd(read))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

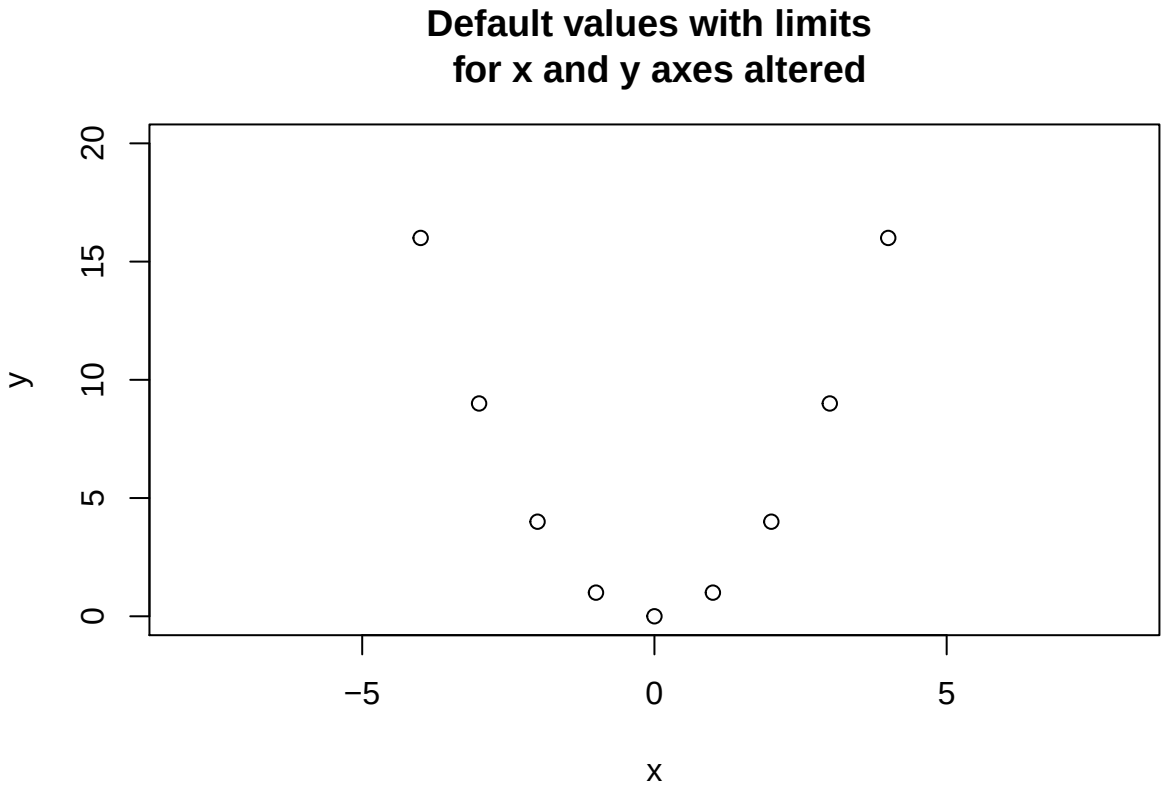
```
head(d_gender_summary)
```

```
## # A tibble: 3 x 5
##   ses    read_average read_median read_var read_sd
##   <fct>         <dbl>       <dbl>    <dbl>   <dbl>
## 1 high           56.5        57.5    118.    10.9
## 2 low            48.3        47       87.3    9.34
## 3 middle         51.6        50       88.8    9.43
```

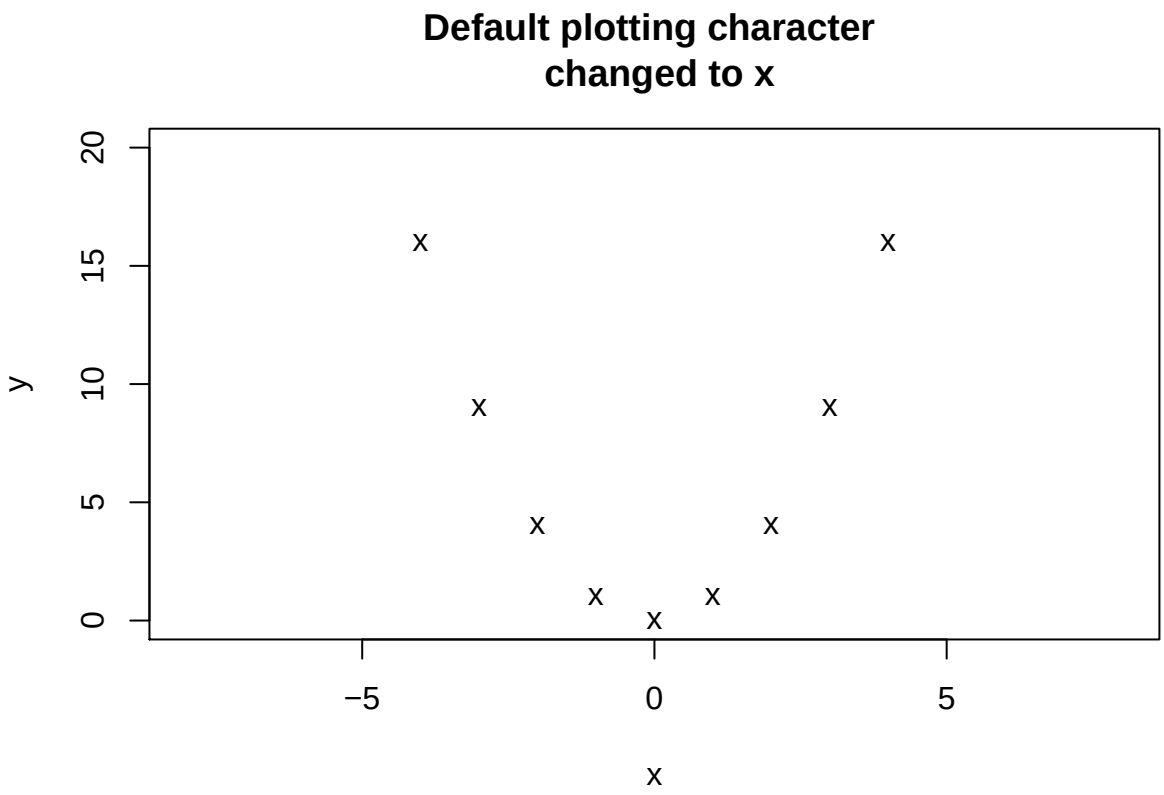
Graphical Power

```
par(mfrow=c(3, 3), pty = "m")  # 3 by 3 layout
x = -4:4
y = x^2
```

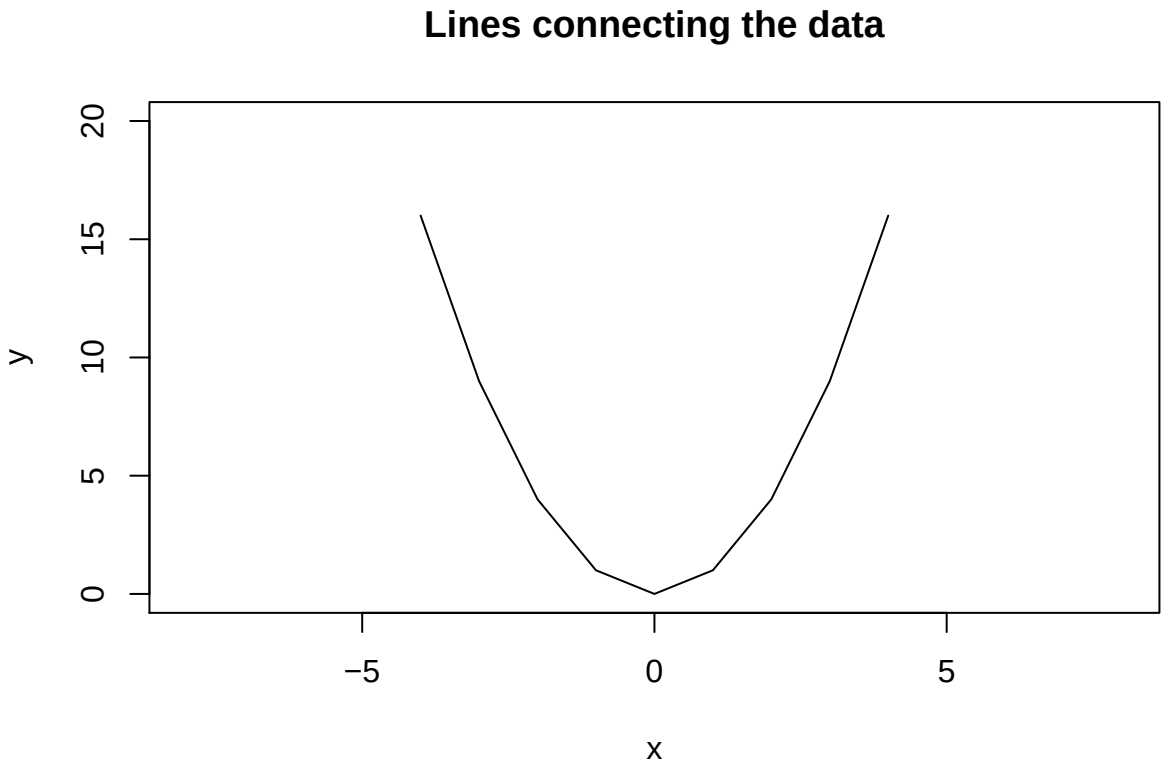
```
plot(x, y, xlim=c(-8, 8), ylim = c(0, 20), main ="")
title(main = "Default values with limits \n for x and y axes altered")
```



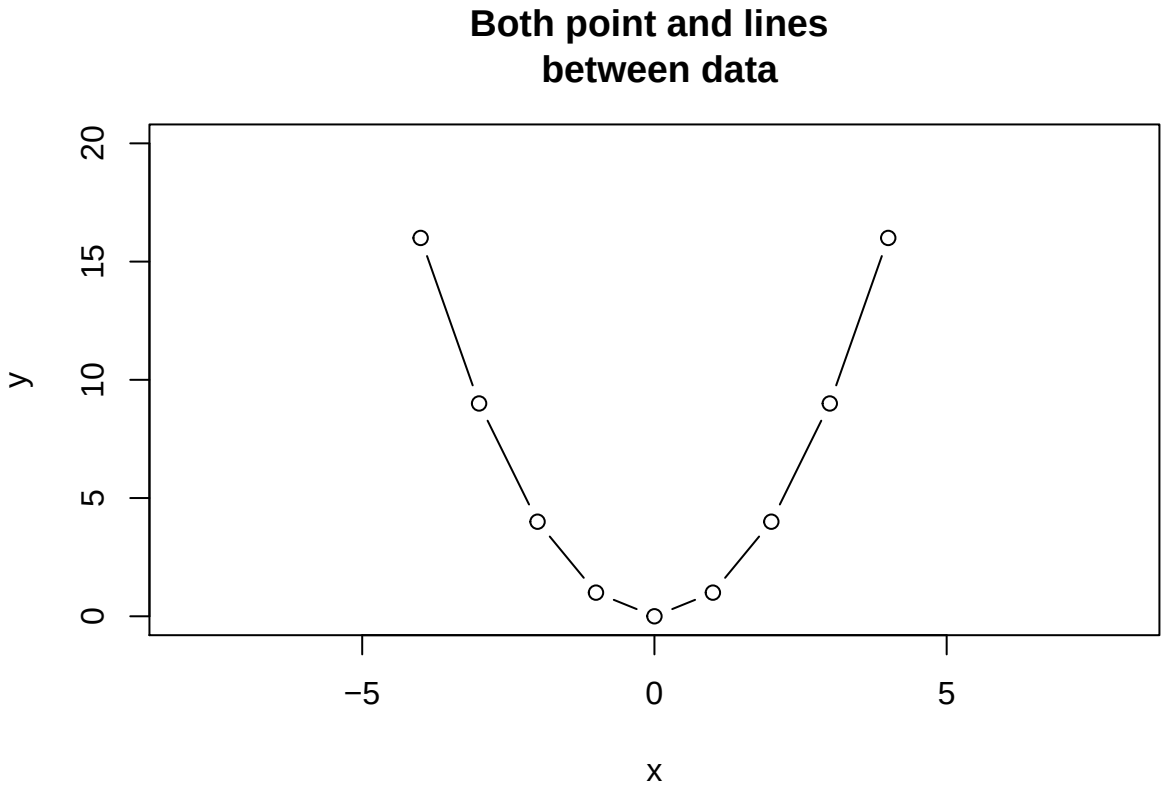

```
plot(x, y, pch = "x", xlim=c(-8, 8), ylim = c(0, 20), main="")
title(main = "Default plotting character \n changed to x")
```



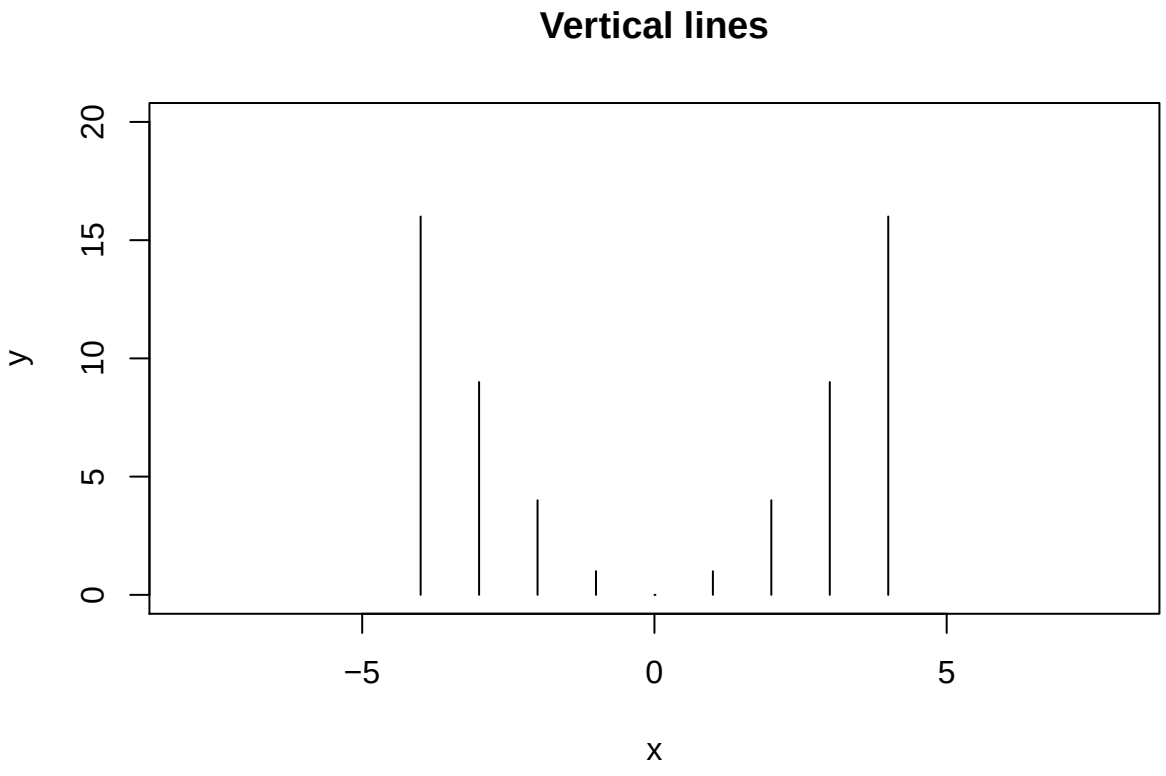
```r
plot(x, y, type = "l", xlim = c(-8, 8), ylim = c(0, 20), main="")
title(main = "Lines connecting the data")
```



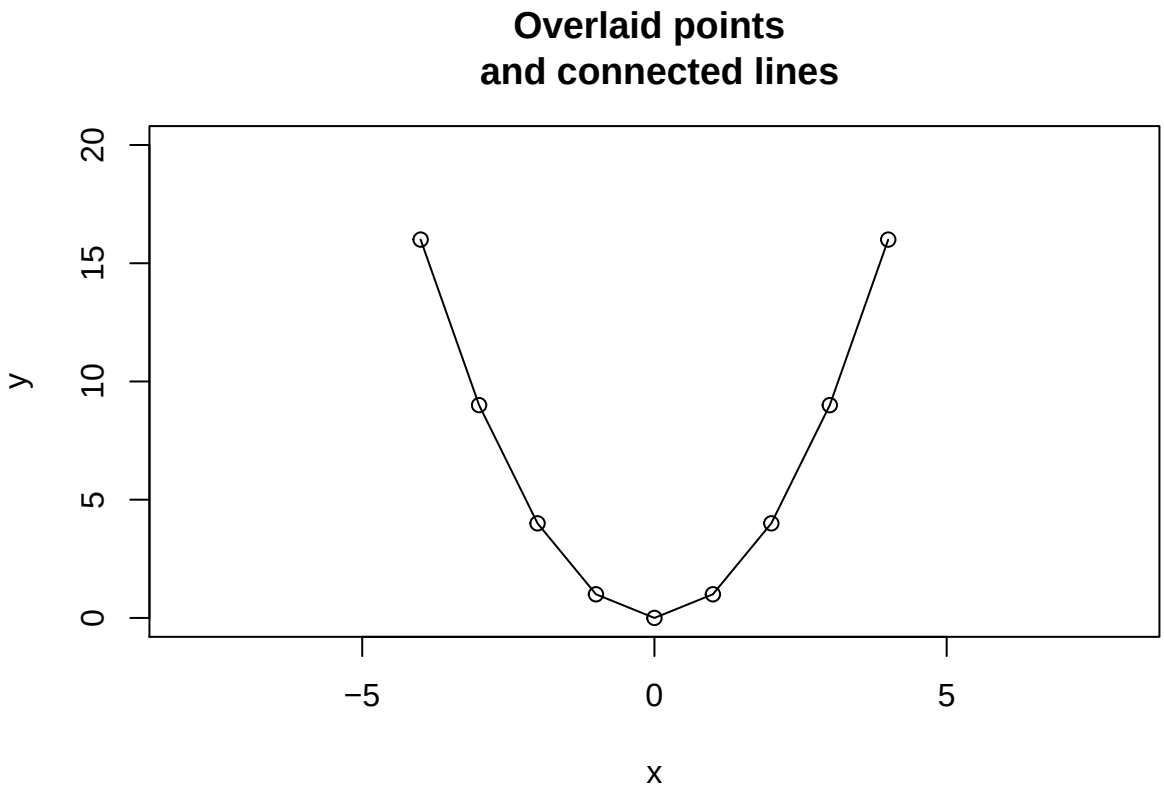

```r
plot(x, y, type = "b", xlim = c(-8, 8), ylim = c(0, 20), main="")
title(main = "Both point and lines \n between data")
```



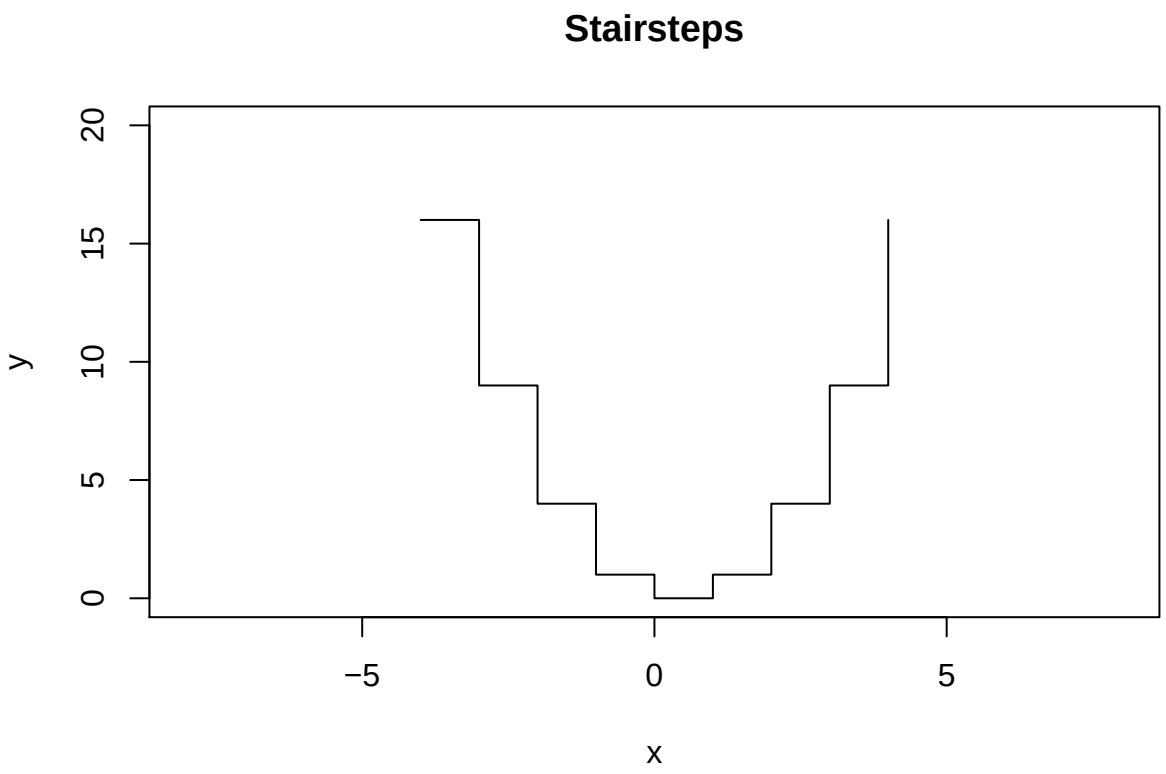

```
plot(x, y, type = "h", xlim = c(-8, 8), ylim = c(0, 20), main="")
title(main = "Vertical lines")
```



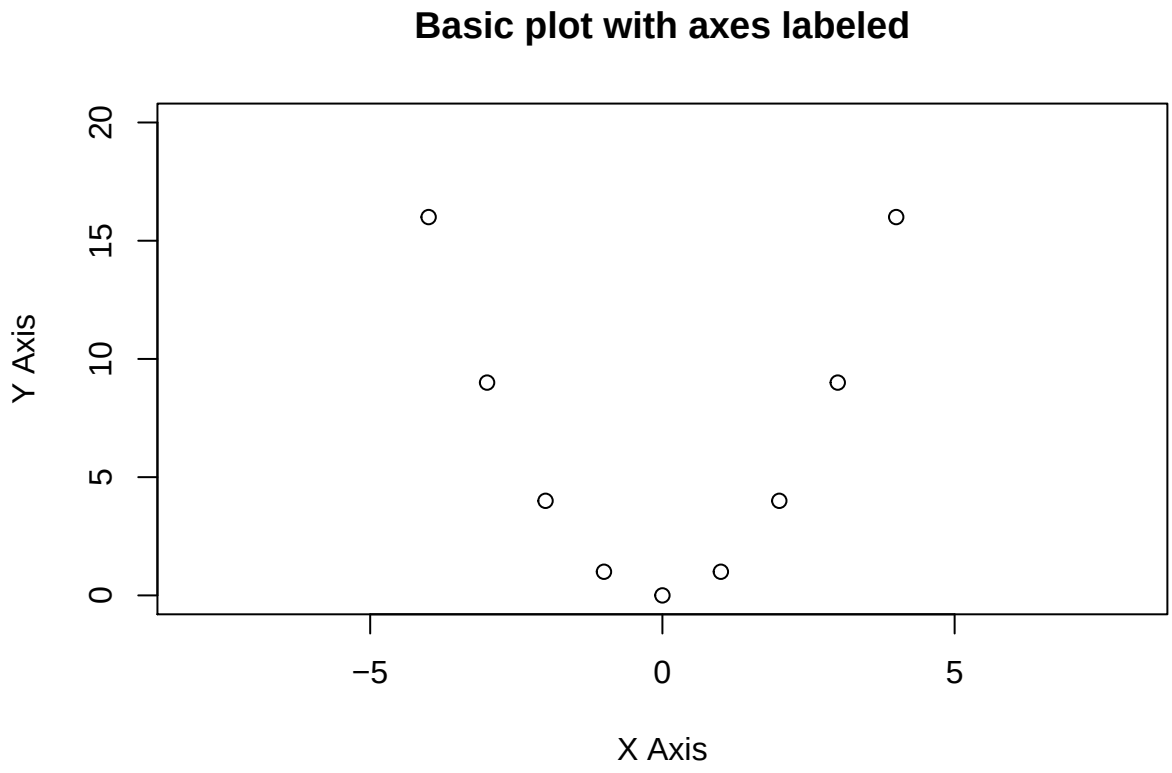
```
plot(x, y, type = "o", xlim = c(-8, 8), ylim = c(0, 20), main="")
title(main = "Overlaid points \n and connected lines")
```



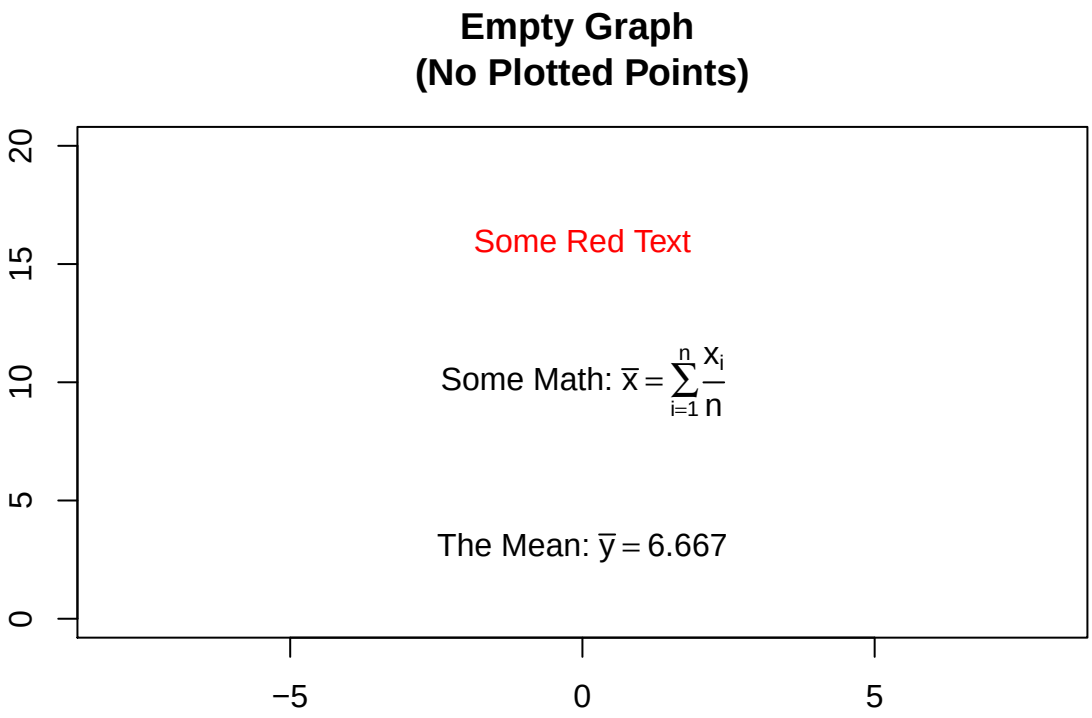

```
plot(x, y, type = "s", xlim = c(-8, 8), ylim = c(0, 20), main="")
title(main = "Stairsteps")
```




```
plot(x, y, xlim = c(-8, 8), ylim = c(0, 20), main = "", xlab = "X Axis",
     ylab = "Y Axis")
title(main = "Basic plot with axes labeled")
```



```
plot(x, y, type = "n", xlim = c(-8, 8), ylim = c(0, 20), xlab = "",
     ylab = "", main = "")
title(main = "Empty Graph \n(No Plotted Points)")
text(0, 16, "Some Red Text", col = "red")
text(0, 10, expression(paste("Some Math: ", bar(x)==sum(frac(x[i],
                                                        n), i==1, n))))
Alpha = round(mean(y), 3)
text(0, 3, bquote(paste("The Mean: ", bar(y)==.(Alpha))))
```

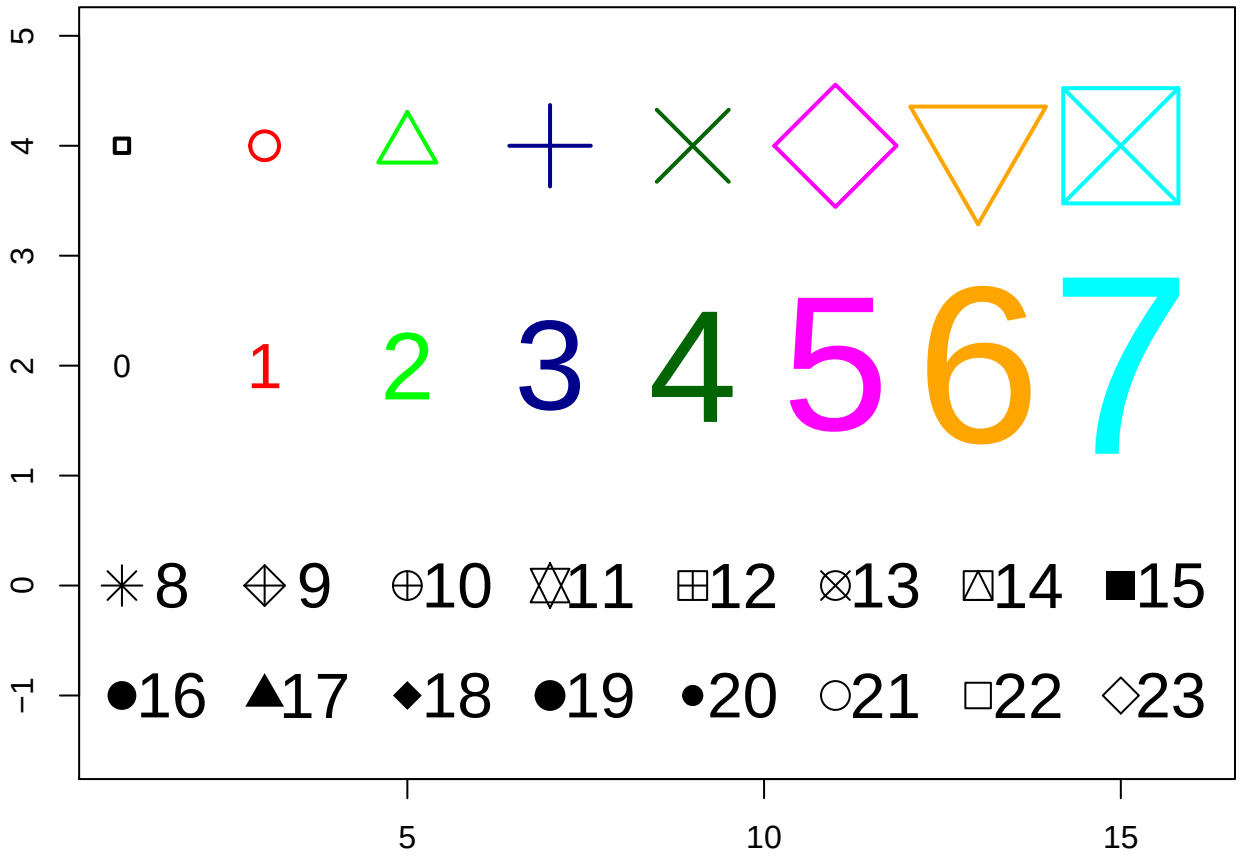



```
par(mfrow=c(1, 1))
```

Colors and points

```
# figure margins of 2.2, 2.2, 0.2, and 0.2 lines
par(mar=c(2, 2, 0, 0) + 0.2)
plot(x = 1, y = 1, xlim = c(1, 16), ylim = c(-1.5, 5), type = "n",
     xlab = "", ylab = "")  # create empty plot with x and y axes
COLORS = c("black", "red", "green", "darkblue", "darkgreen",
           "magenta", "orange", "cyan")  # vector of colors
# symbols (pch = 0:7) placed at (1, 4), (3, 4), ...(15, 4) with
# character expansion 1:8 with color specified in COLORS
points(x = seq(1, 15, 2), y = rep(4, 8), cex = 1:8, col = COLORS,
       pch = 0:7, lwd = 2)
# labels 0:7 placed at (1, 2), (3, 2),..., (15, 2) with
# character expansion 1:8 with color specified in COLORS
text(x = seq(1, 15, 2), y = rep(2, 8), labels = paste(0:7), cex = 1:8,
     col = COLORS)
# symbols (pch = 8:15) placed at (1, 0), (3, 0),..., (15, 0)
# with character expansion of 2
points(x = seq(1, 15, 2), y = rep(0, 8), pch = 8:15, cex = 2)
```

```
# labels 8:15 placed 0.7 to the right of (1, 0), (3, 0),..., (15, 0)
# with character expansion of 2
text(x = seq(1, 15, 2) + 0.7, y = rep(0, 8), labels = paste(8:15),
     cex = 2)
# symbols (pch = 16:23) placed at (1, -1), (3, -1),..., (15, -1)
# with character expansion of 2
points(x = seq(1, 15, 2), y = rep(-1, 8), pch = 16:23, cex = 2)
# labels 16:23 placed 0.7 to the right of (1, -1), (3, -1),..., (15, -1)
# with character expansion of 2
text(x = seq(1, 15, 2) + 0.7, y = rep(-1, 8), labels = paste(16:23),
     cex = 2)
```



ASSIGNMENT 1

1. Rerun all code above to ensure it works.
2. Import several economic data series of your choice from FRED.Generate a variety of plots using different colors and point shapes. Selecting one of the data series, write a one-paragraph narrative about how you would interpret the data series.