Scribe notes

Davis Townsend, Zack Bilderback, Brooks Beckelman

July 28, 2016

Module 3: July 28, 2016

Main Discussion Points:

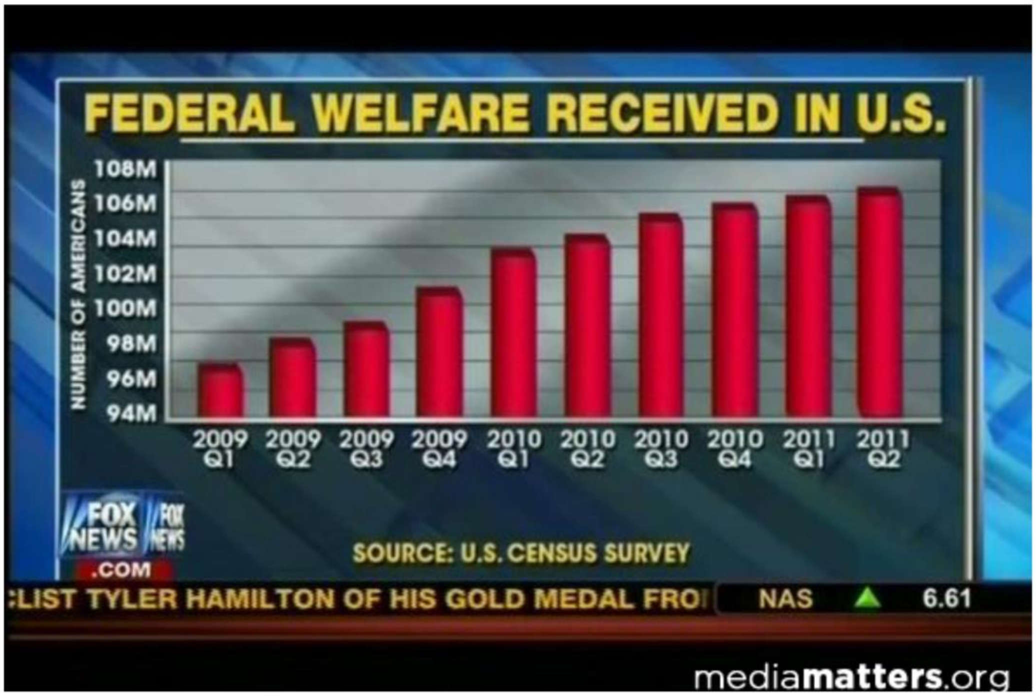
- Data Exploration
- Data Visualizations

Bad Plots

Things that consititute bad plot design

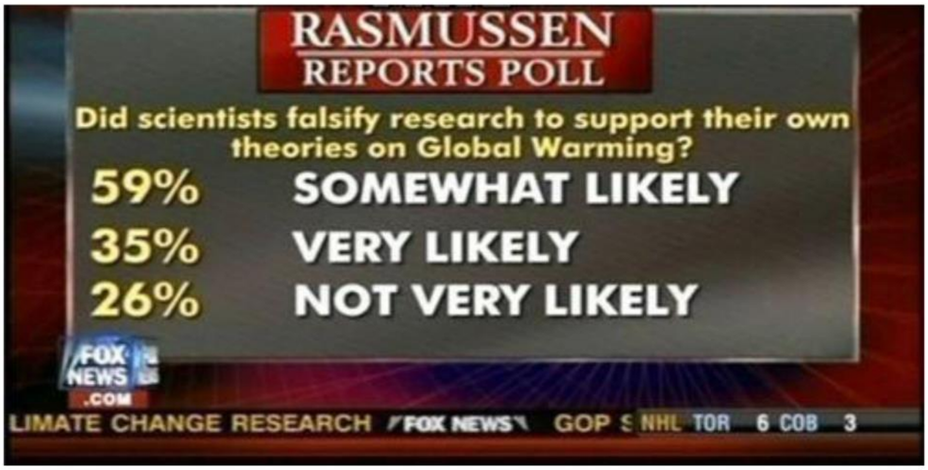
- Truncated y-axis
- Percentages that don't add to 100
- Visual magnitude doesn't map to numerical magnitude
- Distorting relative sizes
- Low information density
- Plots should never look ridiculous ("junk charts")
- Weird perspective
- Wrong choice of display
- Axis absurdities

Example of bad plot designs:



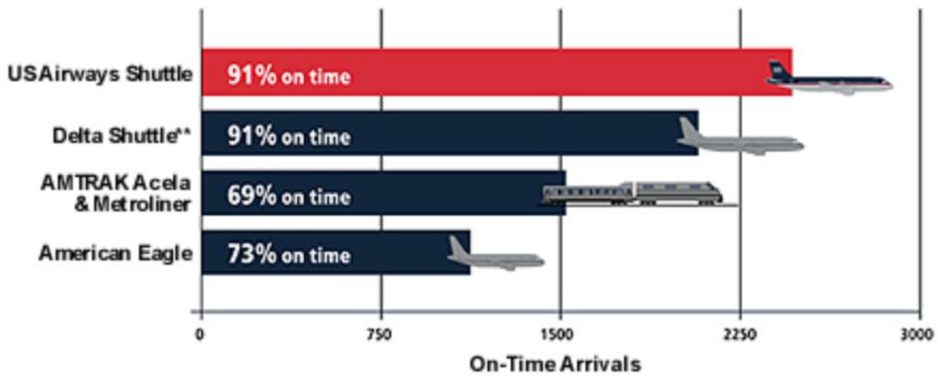
Truncating the y axis

Size of bars in graph misrepresents magnitude of change



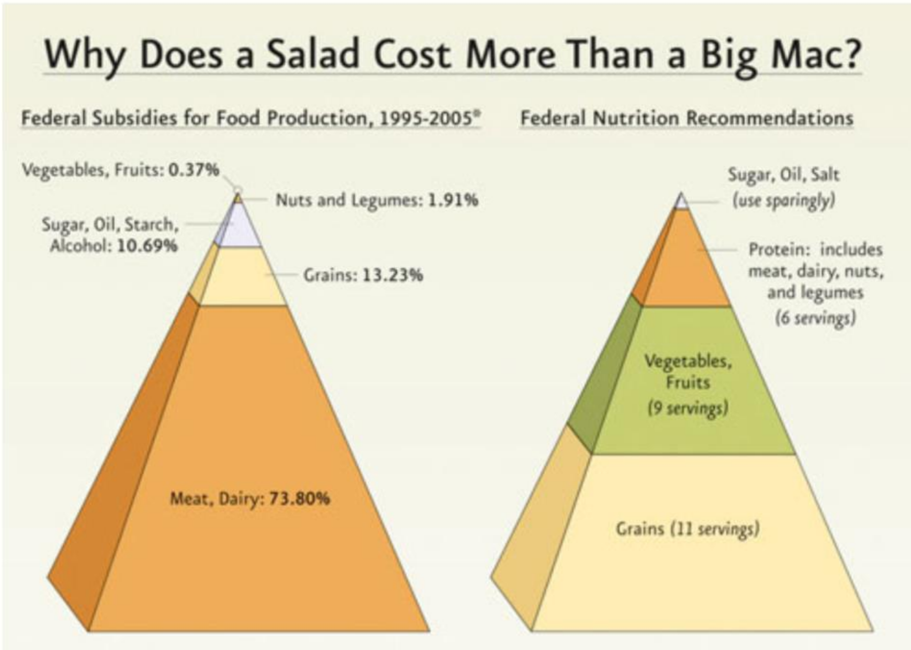
Percentages that don't add to 100

Percentages don't sum to 100



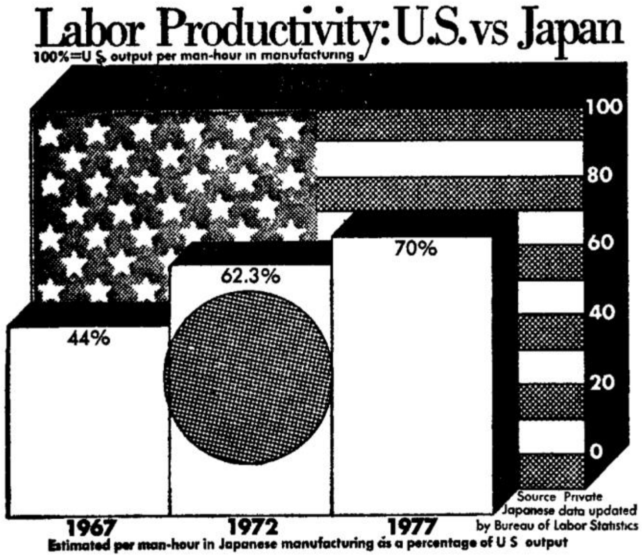
91 > 91?

Visual magnitude does not map to numerical magnitude



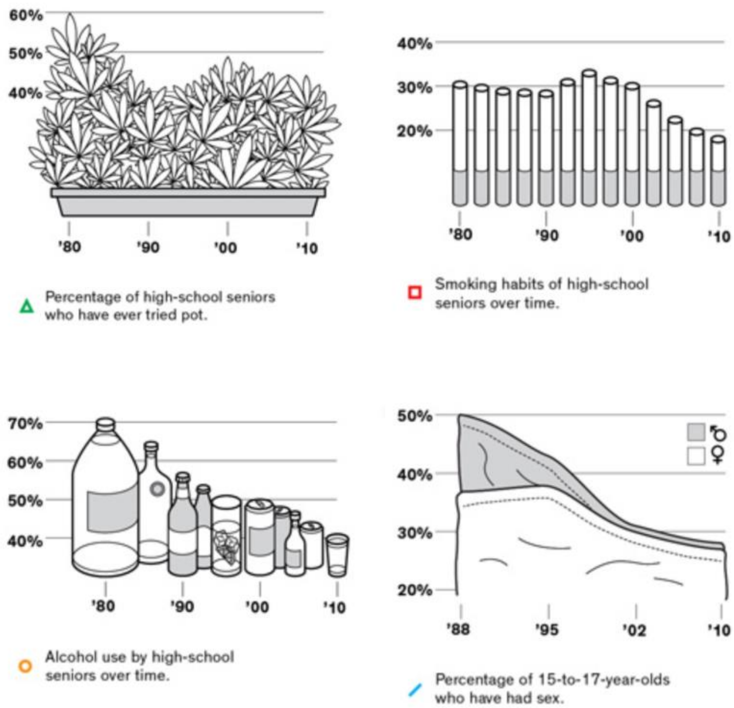
Distorting relative sizes

The size of each food group not proportional to servings due to 3-d aspect of graph. Humans are bad at thinking in terms of volumes



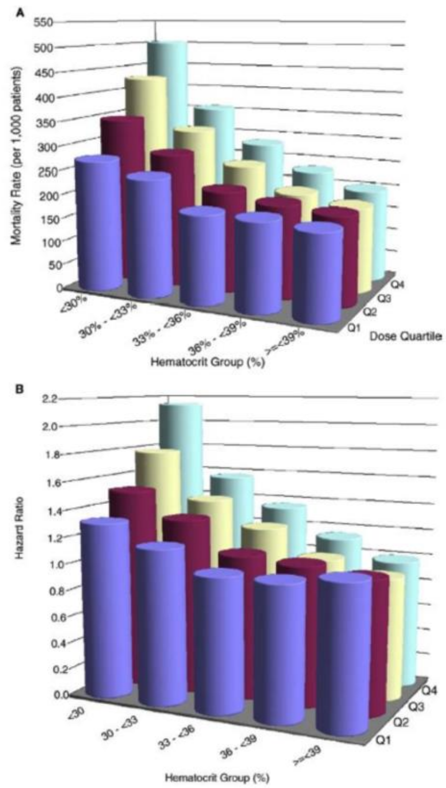
Low information density

This graph would be better represented as a list or a table due to the low amount of information. The graph is also too artsy which interferes with the interpretation



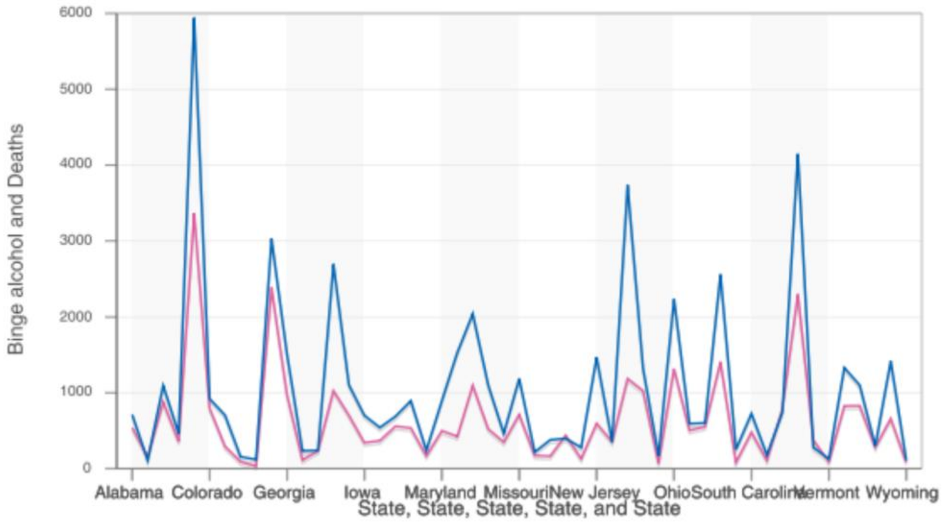
Plots should never look ridiculous.

These charts are known as "junk charts" because they contain useful insights but the visuals distract from the main point of the plot



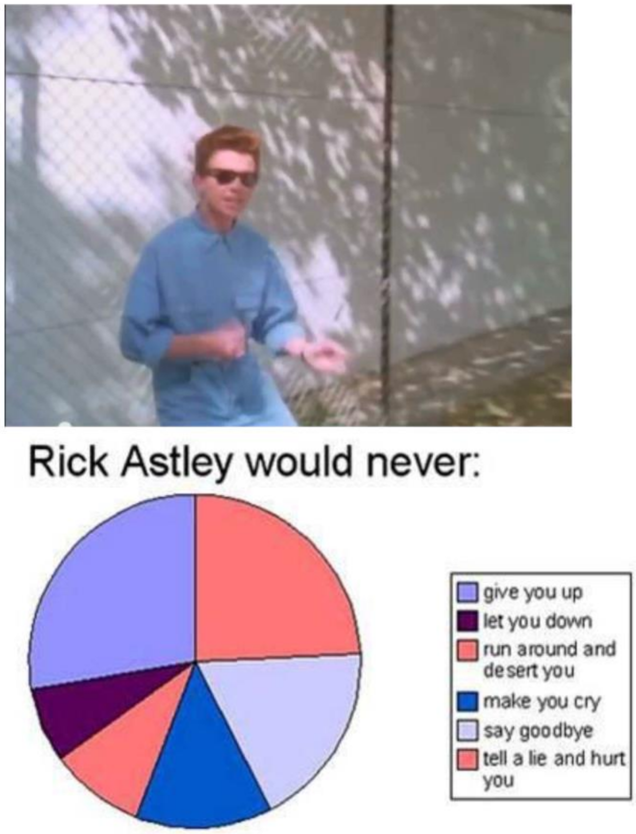
Weird perspective

It is hard to determine which bars are taller from this perspective of the 3-D space



Axis absurdities; wrong choice of display
No adjustment for population

Line chart should not have been used for this representation since there is no relation between the alphabetical order of the states. The data is also not normalized for the population



Avoid pie charts because people are better at estimating numbers when looking at a bar plot when compared to a pie chart. The information comes across more clear with bar plots.

Good Plots

Things that constitute good plot design

- Vehicles for comparison
- Multivariate
- Truthful about magnitude
- Usually not for small datasets
- Not pie charts

Examples of Good Plot designs:

We discussed the figure skater graph in class. This graph summarizes 5 dimensions (time, difficulty, base score, execution, rankings) of data into an interactive and easy to interpret format. You can quickly see the multivariate comparisons between the different skaters, and the circles are truthful about the magnitude.

Another example was the chart that showed all the sectors of the economy during the recession. The graph uses color coding and interactiveness to quickly show comparisons in certain sectors, as well as providing in-depth interpretation

The last example in class was the birth control example. The slope of the lines quickly summarized the comparison, and the interactive feature showed comparative changes between all the graphs, providing quick comparisons of effectiveness. While the data in this graph extrapolated after year 1 is wrong because it assumes independence between failure rates in successive years, the graph is still a good example of data visualizaton and providing effective interactive comparisons.

The graphs we discuss can be found at this link

Goodgraphics

Setup for Scripts

Load libraries

```
library(mosaic)
library(foreach)
```

Read in data: Make sure to check your working directory or set it using setwd()

```
TitanicSurvival = read.csv('TitanicSurvival.csv')
gdpgrowth = read.csv('gdpgrowth.csv', header=TRUE)
```

Titanic scripts

This creates a table of frequencies(i.e. a contingency table) using xtab(cross-tabulation). The data included consists of gender and whether the person survived or not and stores it in the variable t1

```
t1 = xtabs(~survived + sex, data=TitanicSurvival)
t1

##         sex
## survived female male
##      no     127  682
##      yes    339  161
```

This creates a table of proportions from the frequencies we just found and stores it in the p1 variable. Margin=1 makes the rows sum to 1. We see here that if we change margin=2, then we simply make the columns sum to 1 instead of the rows.

```
p1 = prop.table(t1, margin=1)
p1

##         sex
## survived    female      male
##      no  0.1569839 0.8430161
##      yes 0.6780000 0.3220000
```

```
p1 = prop.table(t1, margin=2)
p1

##         sex
## survived    female      male
##      no  0.2725322 0.8090154
##      yes 0.7274678 0.1909846
```

risk table is simply the same command as we have just done. Here we just show that you can explicitly refer to one of the cells in this table by table_name[row #, column #]. So in this case we get the value from row 1, column 2 which is the risk for males (i.e. proportion of men who did not survive)

```
risk_table = prop.table(t1, margin=2)
risk_men = risk_table[1, 2]
risk_men

## [1] 0.8090154
```

Now we will calculate the relative risk of dying for both men and women in terms of the individual cells of the table This value can be thought of like a correlation coeffient for a binary variable. Since it is positive we see that males had a higher chance of not surviving than females. They were about three times as likely to die on the Titanic as women.

```
risk_female = risk_table[1,1]
risk_male = risk_table[1,2]
relative_risk = risk_male/risk_female
relative_risk

## [1] 2.968513
```

gdpgrowth scripts

We can start by simply looking at the first few lines of the data to get a feel for what our data is about.

```
head(gdpgrowth)

##   CODE          COUNTRY  GR6096   DENS60  COAST65  POPGR6090 EAST DEF60
## 1  DZA          Algeria  0.0110 5.396041 4.327307 0.02841708    0 0.030
## 2  BEN            Benin  0.0011 3.900966 4.607945 0.02396531    0 0.018
## 3  BDI          Burundi  0.0046 2.164587 0.000000 0.02027949    0 0.014
## 4  CMR         Cameroon  0.0024 4.475757 3.024604 0.02634325    0 0.024
## 5  CAF Cent'l Afr. Rep. -0.0252 6.006636 0.000000 0.02255507    0 0.009
## 6  COG            Congo  0.0151 5.845420 2.595199 0.02747309    0 0.025
##     LGDP60 EDUC60 LIFE60
## 1 7.451822 0.0297   47.3
## 2 7.003065 0.0248   38.9
## 3 6.461468 0.0183   41.8
## 4 6.463029 0.0221   43.4
```

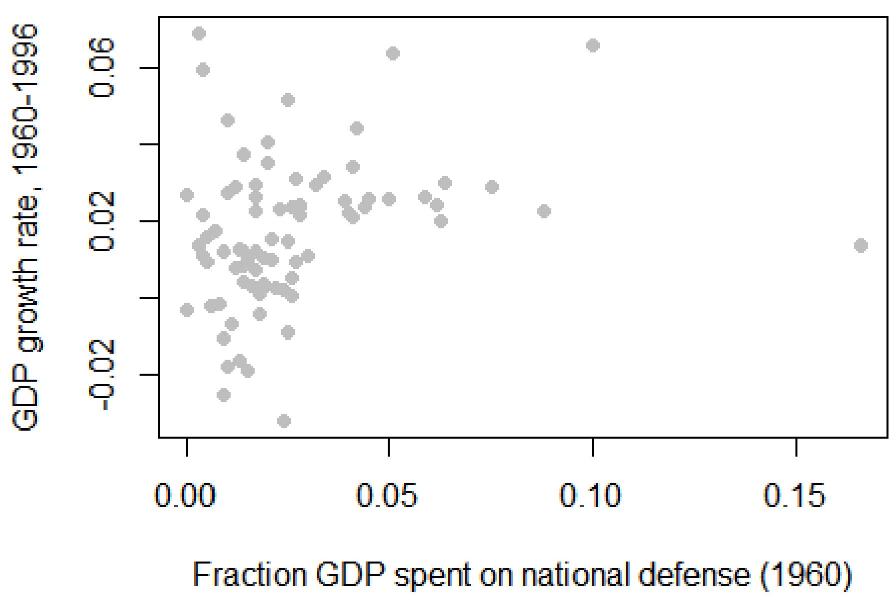
```
## 5 6.556778 0.0231   39.3
## 6 7.023759 0.0311   47.3
```

We can plot the relationship between GDP growth and defense spending. We see in the graph that there's one country that spends a lot more as a fraction of its GDP on defense than any other country.

If we wanted to see what this outlier was, we could run the identify function shown below. This function allows us to click on a point on the plot. (Note that this functionality works in R studio itself but not the markdown file, so in your scripts you should replace the number 54 with the word "outlier" to use the answer you get from the identify function in line 132 of the code)

Then by running the next line of code after the we get the row of the outlier (the country name) as well as the rest of the column details for this row. Xlab and Ylab are simply the labels, or headings for the x and y axis respectively. pch picks which symbol represents each data point on the plot and col is specifying what color the points of the plot should be.

```
plot(gdpgrowth$DEF60, gdpgrowth$GR6096,
    pch=19, col='grey',
    xlab='Fraction GDP spent on national defense (1960)',
    ylab='GDP growth rate, 1960-1996'
    )
outlier = identify(gdpgrowth$DEF60, gdpgrowth$GR6096, n=1)
```




```
gdpgrowth[54,]
```

```
##    CODE COUNTRY GR6096   DENS60  COAST65  POPGR6090 EAST DEF60   LGDP60
## 54  JOR  Jordan  0.014 3.960167 3.099192 0.02837903    0 0.166 7.057898
##    EDUC60 LIFE60
## 54 0.0329   47.2
```

We can discover from this method that the outlier is the country of Jordan.

Now we want to see how much the outlier of Jordan affects the normal Pearson correlation between the value of the 2 variables.

```
cor(gdpgrowth$DEF60, gdpgrowth$GR6096)

## [1] 0.2683152

cor(gdpgrowth$DEF60[-54], gdpgrowth$GR6096[-54])

## [1] 0.3608357
```

We see that the correlation went from .268 up to .36 once we removed the outlier. This is a big jump in correlation from removing just one data point.

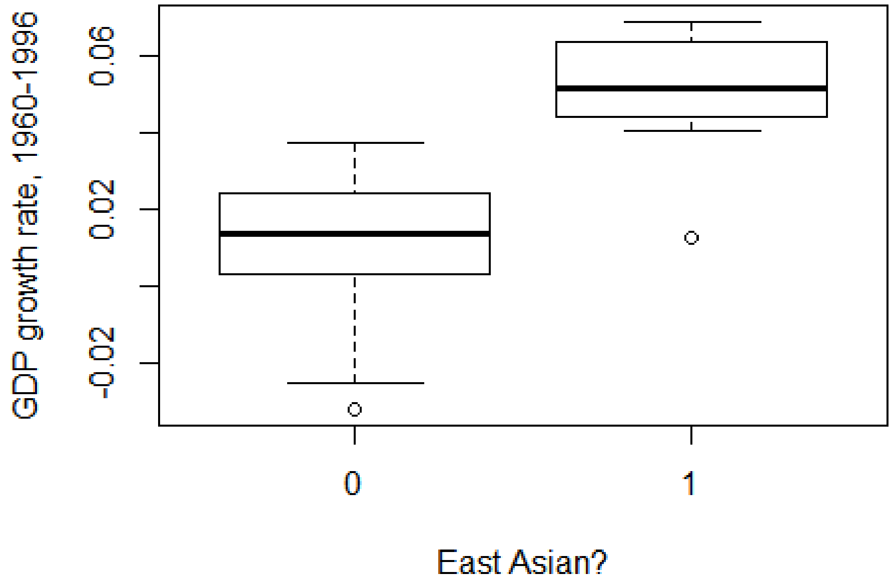
Now we'll look at the same effect of correlation, but this time use a robust measure of correlation, the Spearman correlation. This is a correlation measure between the ranks of the 2 variables, rather than the values.

```
cor(gdpgrowth$DEF60, gdpgrowth$GR6096, method='spearman')

## [1] 0.3381575

cor(gdpgrowth$DEF60[-54], gdpgrowth$GR6096[-54], method='spearman')

## [1] 0.3451648
```

Now we see that the correlation goes up from .338 to .345, a much smaller increase in correlation than we found using the Pearson correlation.

If we wanted a box plot comparing GDP growth rates in East Asian countries vs non East Asian countries we could use the following code which calls the boxplot function with GDP growth as the y and the binary variable EAST as the x.

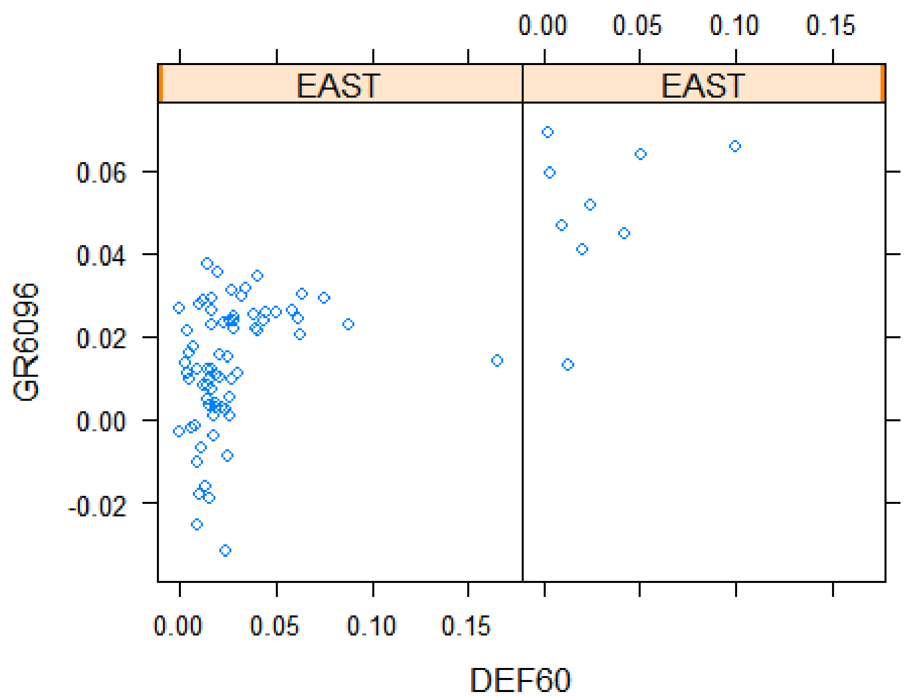
```
boxplot(GR6096 ~ EAST, data=gdpgrowth,
     xlab='East Asian?',
     ylab='GDP growth rate, 1960-1996')
```



YOu can quickly see from this box plot that east asian countries had on average, and in general, higher growth rates than non east asian countries.

If we want to show the relationship between categorical and numerical variables we can use Lattice plots. These plots stratify by a categorical variable. In general, we want the same y-axis in these lattice plots so that we can make comparisons between the 2 sides.

```
xyplot(GR6096 ~ DEF60 | EAST, data=gdpgrowth)
```


EAST
EAST
0.00
0.05
0.10
0.15
0.06
0.04
0.02
0.00
-0.02
GR6096
0.00
0.05
0.10
0.15
DEF60