Summarizing data

Mauricio Romero

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Inference

- Goal: Estimate unknown parameters
- To approximate parameters, we use an estimator, which is a function of the data
- Thus, estimator is a random variable (it is a function of a random variable)
- Use relationship between estimator (its distribution usually) and parameters to infer something about the parameters

Important notation

Based on this tweet: `https://twitter.com/nickchk/status/1272993322395557888`

- Greek letters (e.g., μ) are the truth (i.e., parameters of the true DGP)
- Greek letters with hats (e.g., $\widehat{\mu}$) are estimates (i.e., what we *think* the truth is)
- Non-Greek letters (e.g., X) denote sample/data
- Non-Greek letters with lines on top (e.g., $\overline{X}$) denote calculations from the data (e.g., $\overline{X} = \frac{1}{N}\sum_i X_i$).
- We want to estimate the truth, with some calculation from the data ($\widehat{\mu} = \overline{X}$)
- Data $\longrightarrow$ Calculations $\longrightarrow$ Estimate $\underbrace{\longrightarrow}_{\text{Hopefully}}$ Truth
- Example: X $\longrightarrow \overline{X} \longrightarrow \widehat{\mu} \underbrace{\longrightarrow}_{\text{Hopefully}} \mu$

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Summarizing data

Very good way to describe a variable is by showing it's distribution with a graph

- Density plots (for continuous)
- Histograms (for continuous)
- Bar plots (for categorical)
- Plus:
    - Adding plots together
    - Putting lines on plots
    - Making them look good!

Plot types we won't cover:

- Lots and lots and lots of plot options
- Mosaics, Sankey plots, pie graphs (cause I hate pie graphs)
- Some aren't common in Econ but could be!
- Others are too advanced (like **maps**, but GIS knowledge is a really useful skill!)
- Check out the R Graph Gallery : `https://www.r-graph-gallery.com/`

Density plots and histograms

- Density plots and histograms will show you the full distribution of the variable
- Values along the x-axis, and how often those values show up on y
- The density plots will present a smooth line by averaging nearby values
- A histogram will create "bins" and tell you how many observations fall into each

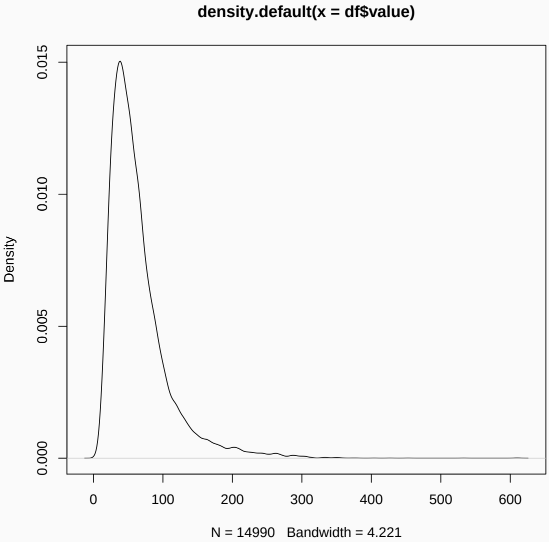
Density plots

- If variable is **continuous**, we can't count how often **each** value comes up

- We smooth it by looking at the number of times it falls within a range

```
df <- read.csv(
'http://www.aire.cdmx.gob.mx/opendata/red_manual/red_manual_particulas_susp.csv',
skip=8,stringsAsFactors = F)
df=filter(df,cve_parameter=="PM10")
plot(density(df$value),ylab="Density",xlab="PM10",lwd=2,
     bty="L",main="Distribution of PM10 in CDMX",
     cex.lab=1.2,cex.axis=1.2)
```

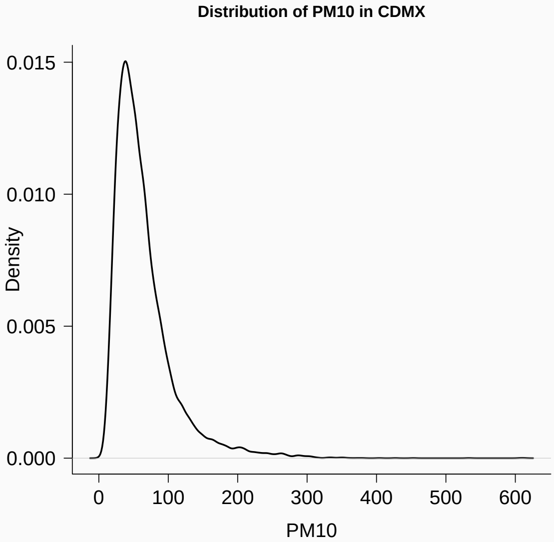
How would you make this figure better?



Titles and labels

- Readability is super important in graphs
- Add labels and titles! Titles with 'main' and axis labels with 'xlab' and 'ylab'
- Usually a good idea to make these bigger than the default (e.g., using ‘cex.lab’)

How would you make this figure better?



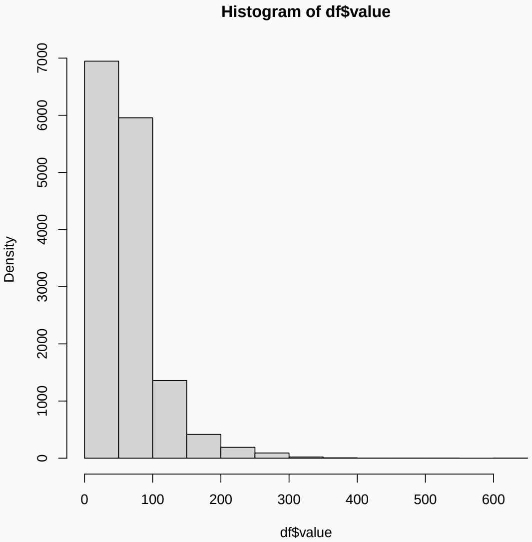
Histograms

- When a variable is **continuous**, we can't count the number of times **each** value comes up

- We create 'bins" and show how many observations fall into each bin

```
hist(df$value,ylab="Density",xlab="PM10",freq=F,bty="L",
     main="Distribution of PM10 in CDMX",breaks=30,col="#FAA43A",
     cex.lab=1.2,cex.axis=1.2)
```

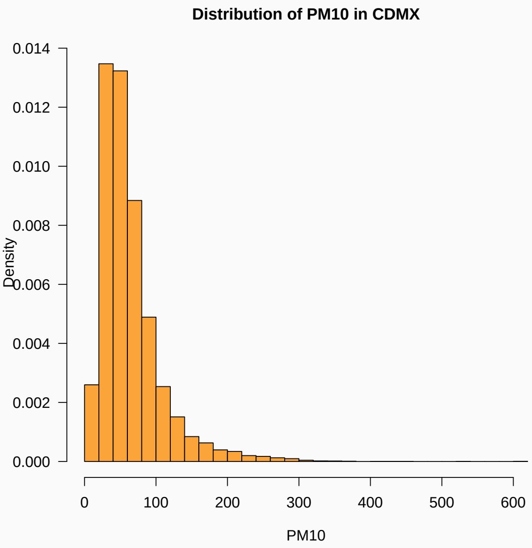
How would you make this figure better?



Histograms

- These need labels too! Other important options:
    - Do proportions with ‘freq=FALSE‘
    - Change how many bins there are, or where they are, with ‘breaks‘

How would you make this figure better?

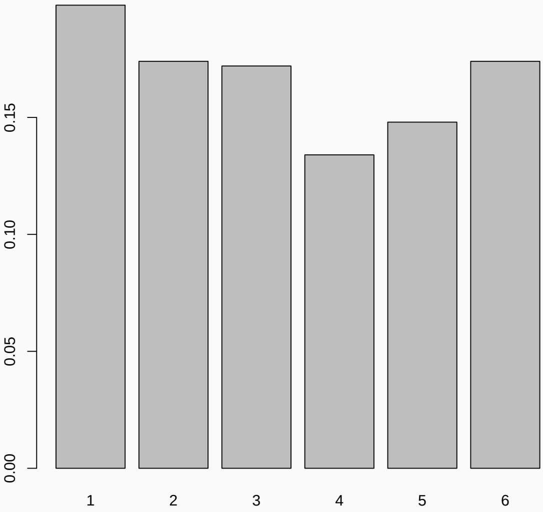


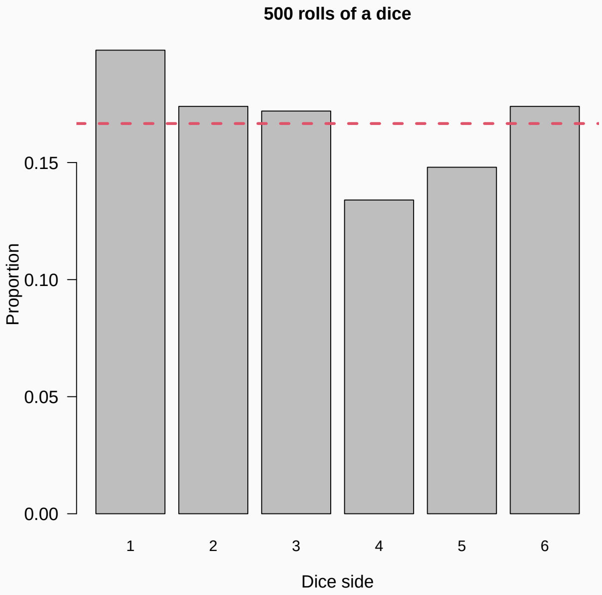
Barplot

- If it's a discreet variable (like a coin toss or a roll of a dice), often best to just count the number (or fraction) of observations in each category
- 'table()' command, shows us the whole distribution of a **categorical**
- Imagine we gather data from 500 rolls of a dice

```
data <- sample(c(1:6),500,replace=TRUE)
data <- data.frame(Result=data)
table(data)
prop.table(table(data))
barplot(prop.table(table(data)))
abline(h=1/6,lwd=2,col=2,lty=2) #This is the true distribution
```

```
> table(data)
data
 1  2  3  4  5  6 
94 81 76 81 97 71 
> prop.table(table(data))
data
    1     2     3     4     5     6 
0.188 0.162 0.152 0.162 0.194 0.142 
> 
```


0.00
0.05
0.10
0.15
1
2
3
4
5
6

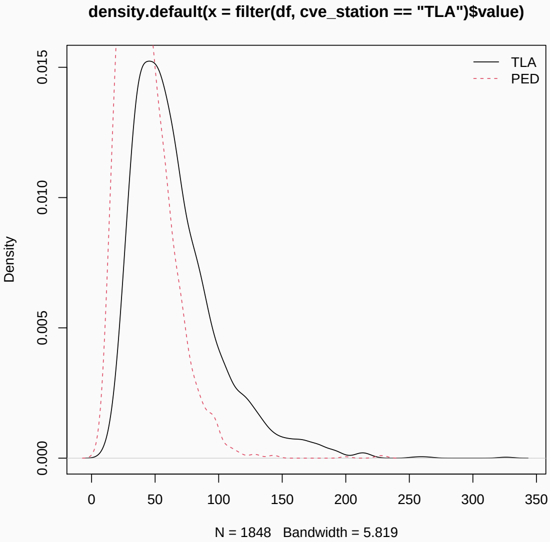

500 rolls of a dice
Proportion
0.00
0.05
0.10
0.15
1
2
3
4
5
6
Dice side

Overlaying Densities

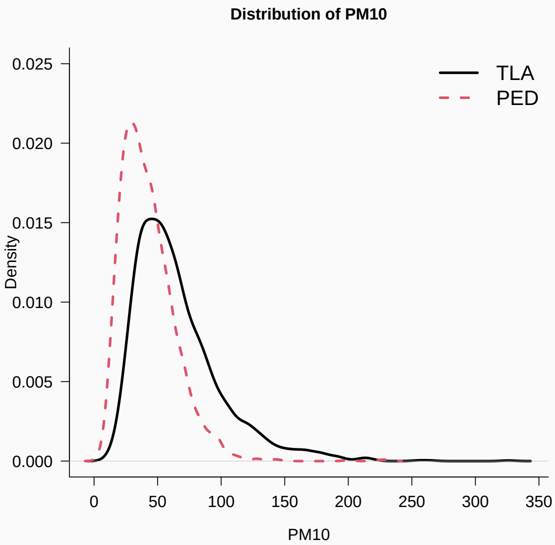
- Sometimes it's nice to be able to compare two distributions
- Because density plots are so simple, we can do this by overlaying them
- The 'lines()' function will add a line to your graph
- Be sure to set colors so you can tell them apart

```
plot(density(filter(df,cve_station=="TLA")$value),
ylab="Density",xlab="PM10",lwd=2,
     bty="L",main="Distribution of PM10",
     cex.lab=1.2,cex.axis=1.2,ylim=c(0,0.015))

lines(density(filter(df,cve_station=="PED")$value),col=2,lwd=2,lty=2)

legend("topright",c("TLA","PED"),
col=c(1,2),lty=c(1,2),cex=1.5,bty="n",lwd=2)
```

How would you make this figure better?



How would you make this figure better?



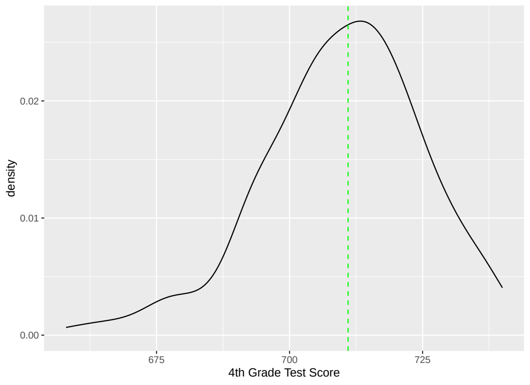
Practice

- Install ‘Ecdata‘, load it, and get the ‘MCAS‘ data
- Make a density plot for ‘totsc4‘, and add a vertical green dashed line at the median
- Create a bar plot showing the proportion of observations that have nonzero values of ‘bilingua‘
- Go back and add appropriate titles and/or axis labels to all graphs

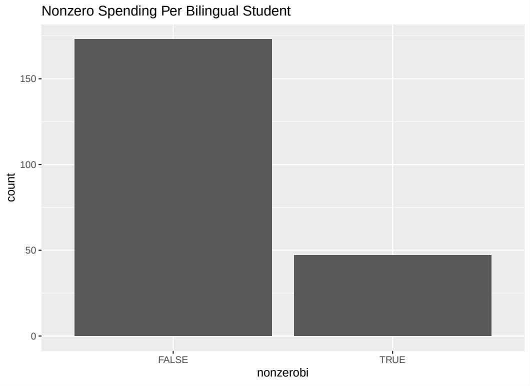
Practice answers

```r
install.packages('Ecdata')
library(Ecdata)
data(MCAS)
plot(density(MCAS$totsc4),xlab="4th Grade Test Score")
abline(v=median(MCAS$totsc4),col='green',lty='dashed')
#THE GGPLOT2 WAY
ggplot(MCAS,aes(x=totsc4))+stat_density(geom='line')+
  geom_vline(aes(xintercept=median(totsc4)),
  color='green',linetype='dashed')+
  xlab("4th Grade Test Score")
MCAS <- MCAS %>%
  mutate(nonzerobi = MCAS$bilingua > 0)
ggplot(MCAS,aes(x=nonzerobi))+geom_bar()+
  ggtitle("Nonzero Spending Per Bilingual Student")
ggplot(MCAS,aes(y=totday))+geom_boxplot()+
  geom_hline(aes(yintercept=mean(totday)))+
  ggtitle("Spending Per Pupil")
```

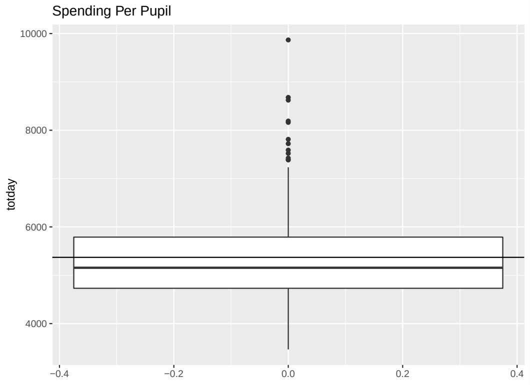
How would you make this figure better?



How would you make this figure better?



How would you make this figure better?



Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Summary statistics

- The mean, median, etc. **describe the distribution** in a condensed way
- Means and medians both describe the **center** of the distribution
- Percentiles describe where other parts not necessarily in the middle are
- Standard deviations and variances describe how **spread out** the distribution is
- Important to differentiate between the truth (e.g., $\mathbb{E}(X)$) and the sample equivalent (e.g., $\overline{X}$)

The Mean — sample equivalent of the expected value

- $\mathbb{E}[X] := \sum_x f(x)x$
- The sample equivalent is the mean (or average)
- Calculated by multiplying each value by the proportion of times it comes up, and adding it all together
- In R, 'mean(x)'
- Essentially we estimate $f(x)$ with the proportion of times x shows up in the data
- Let our random variable be the distribution of the rolls of a die
- $\mathbb{E}(X) = \sum_{x=1}^{6} x\frac{1}{6} = 3$
- From our 500 simulations: $\overline{X} = 3.382$

```r
mean(data)
```

The Mean

- Nice things about the mean:
  - Easy to understand
  - The mean of ‘x-mean(x)‘ is 0 (same for $\mathbb{E}(\mathbb{E}(X) - X) = 0$)
  - Good statistical properties
  - Makes sense with large or small samples, with discrete or continuous variables
- Not so nice:
  - Sensitive to outliers (also true to $\mathbb{E}(X)$)

```
mean(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1))
mean(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,100))
```

- The mean doesn't describe **EVERYTHING** about the distribution!

```
> #mean is sensitive to outliers
> mean(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1))
[1] 1
> mean(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,100))
[1] 5.95
> 
```

The Median

- Order observations from smallest to largest and pick the one in the middle
- If there's an even number of observations, take the mean of the two middle
- Population equivalent is m such that $P(X \leq m) = P(X \geq m) = \frac{1}{2}$

```r
x <- c(3,1,4,2,2)
median(x)
sort(x)[round(length(x)/2)]
```

```
> x <- c(3,1,4,2,2)
> median(x)
[1] 2
> sort(x)[round(length(x)/2)]
[1] 2
> 
```

The Median

- Nice things about the median:
  - Super easy to calculate (you can often do it by hand)
  - Represents the “typical” observation
  - Not sensitive to outliers

```
median(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1))
median(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,100))
```

  - Generally not affected by transforming the data
- Not so nice:
  - Insensitive to outliers means it can ignore real changes in the “tails”
  - Can ignore magnitudes generally
  - May be highly sensitive if there are big gaps between observations

```
> #median is not sensitive to outliers
> median(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1))
[1] 1
> median(c(1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,100))
[1] 1
> 
```

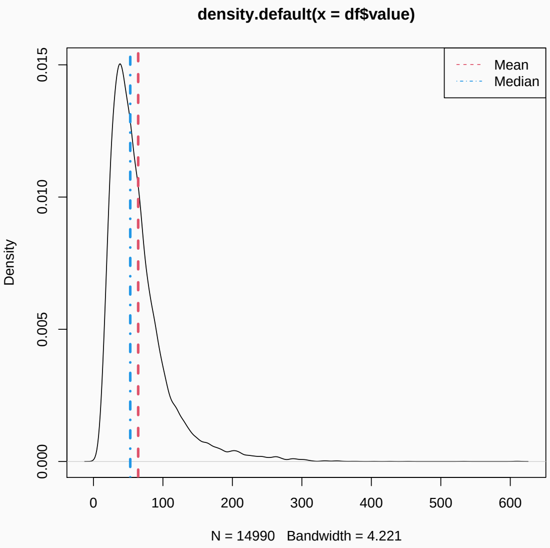
Adding Lines

- It’s pretty common for us to want to add some explanatory lines to our graphs
- For example, adding mean/median/etc. to a density
- Or showing where 0 is
- Do this with ‘abline()‘
- After creating the plot, THEN
  - Add the ‘abline(intercept,slope)‘
  - ‘abline(h=horizontal)‘ for horizontal numbers
  - ‘abline(v=vertical)‘ for vertical numbers
- Don’t forget to add a legend or a figure note explaining what the lines are

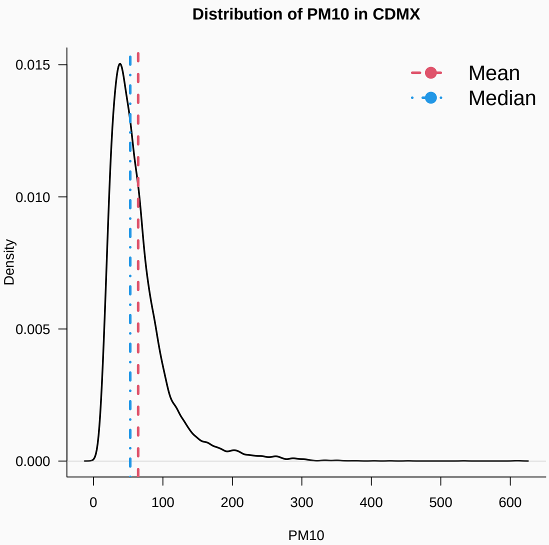
Mean and Median Together

```r
plot(density(df$value),ylab="Density",xlab="PM10",lwd=2,bty="L",
main="Distribution of PM10 in CDMX")
abline(v=mean(df$value),lwd=3,col=2,lty=2)
abline(v=median(df$value),lwd=3,col=4,lty=4)
legend("topright",c("Mean","Median"),col=c(2,4),pch=19,cex=1.5,bty="n")
```

How would you make this figure better?



How would you make this figure better?



Percentiles

- A percentile is just like a median
- Except that you don't necessarily pick the MIDDLE
- Sort observations and pick the (percentile)th observation
- Population equivalent: Percentile *kth* is m such that $P(X < m) < k/100$
- Use the ‘quantile()‘ function, and list the percentiles you want
- Percentiles can fully describe the distribution if you use enough!

```
quantile(c(0,1,2,3,4,5),c(.4,.5,1))
median(c(0,1,2,3,4,5))
```

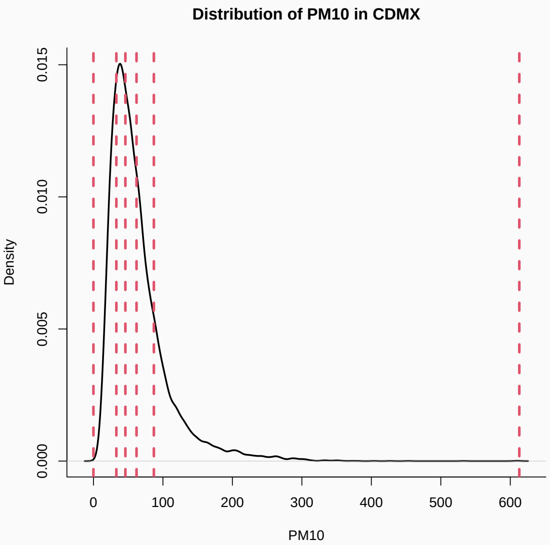
```
> quantile(c(0,1,2,3,4,5),c(.4,.5,1))
 40%  50% 100% 
 2.0  2.5  5.0 
> median(c(0,1,2,3,4,5))
[1] 2.5
> 
```

Percentiles

Exactly 20% of the observations are between each set of lines

```
plot(density(df$value),ylab="Density",xlab="PM10",lwd=2,bty="L",
     main="Distribution of PM10 in CDMX")
abline(v=quantile(df$value,(0:5)/5),lwd=3,col=2,lty=2)
```

How would you make this figure better?



Min and Max

- Also useful are the minimum and maximum (a.k.a. the 0% and 100% percentiles)
- Show you the range of values that the variable takes in the sample
- ‘min()‘ and ‘max()‘

Standard deviation and variance

- Standard ways of understanding how much the data **varies around the mean**
- Variance = Standard deviation squared
- The higher these values, the less good a description the mean is of the variable

Standard deviation and variance

- Start with data and subtract out the mean
- The result is the residuals (left-over part, unexplained part)
- Square the residuals
- Average them (variance) [note: then multiply by N/(N-1)]
- Square root of the variance is the standard deviation
- Why this process rather than some other measure around the mean (i.e. why square it)? Good statistical reasons I promise

Standard deviation and variance

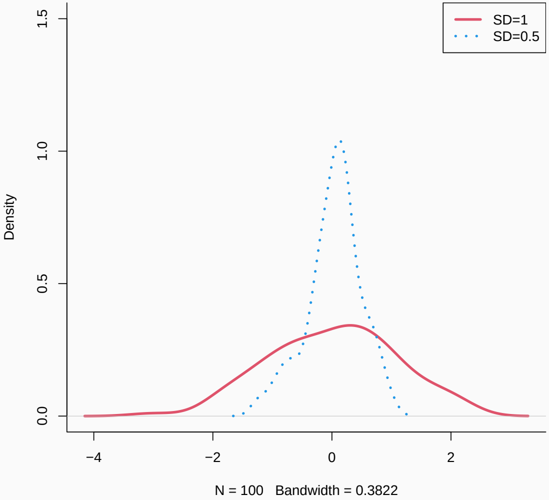
```r
data <- c(1,1,1,1,2)
data <- data - mean(data)
data
#Variance, sd
c((5/4)*mean(data^2),var(c(1,1,1,1,2)),
  sqrt((5/4)*mean(data^2)),sd(c(1,1,1,1,2)))

data2 <- c(100,0,-30,50,80)
data2 <- data2 - mean(data2)
#Variance, sd
c((5/4)*mean(data2^2),var(c(100,0,-30,50,80)),
  sqrt((5/4)*mean(data2^2)),sd(c(100,0,-30,50,80)))
```

```
> #Standard deviations
> data <- c(1,1,1,1,2)
> data <- data - mean(data)
> data
[1] -0.2 -0.2 -0.2 -0.2  0.8
> #Variance, sd
> c((5/4)*mean(data^2),var(c(1,1,1,1,2)),
+   sqrt((5/4)*mean(data^2)),sd(c(1,1,1,1,2)))
[1] 0.2000000 0.2000000 0.4472136 0.4472136
>
> data2 <- c(100,0,-30,50,80)
> data2 <- data2 - mean(data2)
> #Variance, sd
> c((5/4)*mean(data2^2),var(c(100,0,-30,50,80)),
+   sqrt((5/4)*mean(data2^2)),sd(c(100,0,-30,50,80)))
[1] 2950.0000 2950.0000   54.3139   54.3139
> 
```

Graphically, SD and variance tell you how “wide” the distribution is

```
dat <- data.frame(wide = rnorm(100,sd=1),
                  narrow = rnorm(100,sd=1/2))
plot(density(dat$wide),col=2,lwd=2,type="l",bty="L",ylim=c(0,1.5))
lines(density(dat$narrow),col=4,lwd=2,type="l",bty="L",lty=3)
legend("topright",c("SD=1","SD=0.5"),col=c(2,4),lty=c(1,3),lwd=3)
```

How would you make this figure better?



Summary statistics table

- Something we will often want to do is display a bunch of summary statistics at once for the variables we have
- This makes it easy to understand a variable's distribution at a glance
- We'll be using the 'stargazer' command for this

```
library(stargazer)
data(LifeCycleSavings)
stargazer(LifeCycleSavings,type='text')
```

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
sr	50	9.671	4.480	0.600	6.970	12.617	21.100
pop15	50	35.090	9.152	21.440	26.215	44.065	47.640
pop75	50	2.293	1.291	0.560	1.125	3.325	4.700
dpi	50	1,106.758	990.869	88.940	288.207	1,795.622	4,001.890
ddpi	50	3.758	2.870	0.220	2.002	4.477	16.710

Summary statistics table

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
Savings	50	9.7	4.5	0.6	7.0	12.6	21.1
% of population under 15	50	35.1	9.2	21.4	26.2	44.1	47.6
% of population over 75	50	2.3	1.3	0.6	1.1	3.3	4.7
Disposable income (DPI)	50	1,106.8	990.9	88.9	288.2	1,795.6	4,001.9
% growth rate of DPI	50	3.8	2.9	0.2	2.0	4.5	16.7

Stargazer

- See help(stargazer) to see what other summary stats you can include
- ‘type=’text’‘ tells it to give us a basic text table.
- Another one is ‘type=’html’‘, especially if we want to output our table to a file
  - You can open up the HTML table and copy/paste it into Excel or Word
- Another one is ‘type=’latex’‘, for when exporting to LaTeX
- ‘out=’filename’‘ will save our results to a file

Putting it all together

- Data about public employees salaries is widely available in Mexico
  - `https://nominatransparente.rhnet.gob.mx/`
  - `ttps://tudinero.cdmx.gob.mx/buscador_personas`
  - `https://datos.cdmx.gob.mx/explore/dataset/remuneraciones-al-personal-de-la-ciudad-de-mexico`
  - `https://www.transparencia.cdmx.gob.mx/`
  - `https://sep.gob.mx/es/sep1/Articulo_73_de_la_Ley_General_de_Contabilidad_Gubernamental_`
- To speed things up, I cleaned a little the data of the 4th trimester of 2019 for the Secretaria de Gobierno (CDMX)
- Will produce some basic statistics

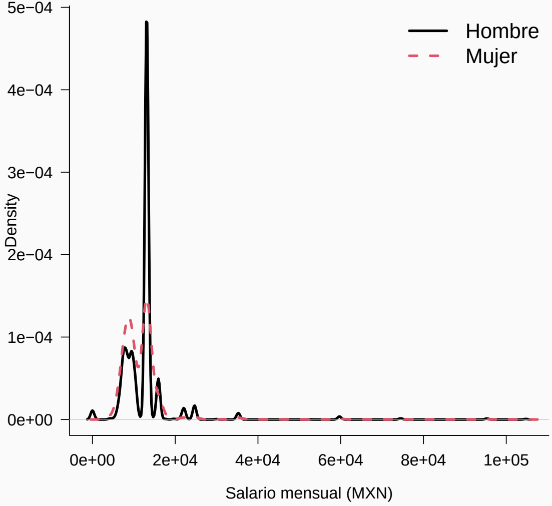
Practice answers

```
library(stargazer)[basicstyle=\tiny]
SalariosCDMX=read.csv("http://mauricio-romero.com/data/class/SalariosCDMX20
SalariosCDMX$Mujer=(SalariosCDMX$Sexo=="Femenino")
SalariosCDMX$Confianza=(SalariosCDMX$Tipo=="Personal de confianza")
```

Summary statistics table

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
Clave puesto	9,773	231.4	231.6	20	159	190	1,184
Monto bruto	9,773	12,170.0	6,390.9	0	8,960	13,119	109,981
Monto neto	9,773	9,700.9	4,837.2	−22,323.2	7,004.3	10,415.8	77,909.5
Mujer (=1)	9,773	0.5	0.5	0	0	1	1
Confianza (=1)	9,773	0.7	0.5	0	0	1	1

How would you make this figure better?



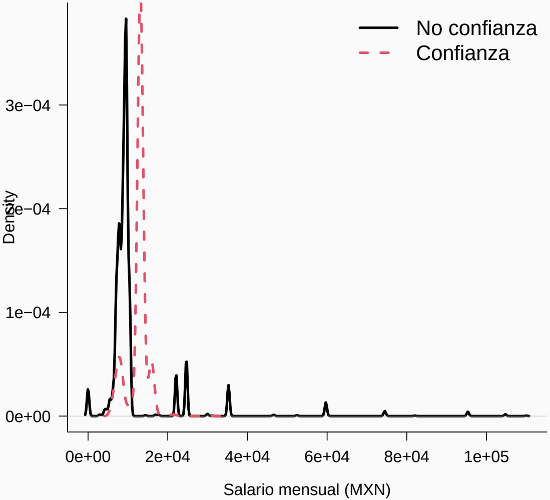
Summary statistics table - Males

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
Clave puesto	5,238	221.5	215.9	24	169	190	1,184
Monto bruto	5,238	12,678.5	6,966.0	0	9,640	13,119	104,740
Monto neto	5,238	10,077.8	5,239.1	0.0	7,305.0	10,523.5	74,971.0
Mujer (=1)	5,238	0.0	0.0	0	0	0	0
Confianza (=1)	5,238	0.7	0.5	0	0	1	1

Summary statistics table - Females

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
Clave puesto	4,535	242.7	248.1	20	150	190	1,096
Monto bruto	4,535	11,582.8	5,597.8	2,816	8,790	13,119	109,981
Monto neto	4,535	9,265.6	4,286.5	−22,323.2	6,745.0	10,292.7	77,909.5
Mujer (=1)	4,535	1.0	0.0	1	1	1	1
Confianza (=1)	4,535	0.6	0.5	0	0	1	1

How would you make this figure better?



Practice

- Use ‘data(LifeCycleSavings)‘ to get the Life Cycle Savings data, and use ‘help()‘ and ‘str()‘ to look at it
- Use ‘stargazer()‘ to get a text table of summary statistics for all the variables EXCEPT ddpi
- Now make an HTML table for all the variables. Open the file and look at it in a browser.
- For each of the statistics that the ‘stargazer()‘ table gives you, plus the median, calculate that statistic on your own for the ‘pop15‘ variable using the appropriate R function
- Calculate the max, min, and median in two ways - using their own respective functions, and as percentiles.

Practice answers

```
library(stargazer)
data(LifeCycleSavings)
help(LifeCycleSavings)
str(LifeCycleSavings)
stargazer(select(LifeCycleSavings,-ddpi),type='text')
stargazer(select(LifeCycleSavings,-ddpi),type='html',out='table.html')
LS <- LifeCycleSavings
c(length(LS$pop15),mean(LS$pop15),sd(LS$pop15),min(LS$pop15),
  quantile(LS$pop15,c(0,.25,.5,.75,1)),max(LS$pop15),median(LS$pop15))
```

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Relationships Between Variables

- We aren't just interested in looking at variables by themselves!
- We want to know how variables can be **related** to each other
- When 'X' is high, would we expect 'Y' to also be high, or be low?
- How are variables **correlated**?
- How does one variable **explain** another?
- How does one variable **cause** another? (what most of this course is about)

What Does it Mean to be Related?

- Two variables are **related** if knowing something about one tells you something about the other
- For example, consider the answer to two questions:
  - Do you have an uterus?
  - Are you pregnant?
- What is the probability that a random person is pregnant?
- What is the probability that a random person **who doesn't have an uterus** is pregnant?

What Does it Mean to be Related?

- Variables are **dependent** if telling you the value of one gives you information about the value of the other
- Variables are **correlated** if knowing whether one of them is **unusually high** gives you information about whether the other is **unusually high**(positive correlation) or **unusually low** (negative correlation)
- **Explaining** one variable 'Y' with another 'X' means predicting **'Y'** by looking at the distribution of 'Y' for **a given** value of 'X'

An Example: Dependence

```
wage1 <- read_stata(
"http://fmwww.bc.edu/ec-p/data/wooldridge/wage1.dta"
)
table(wage1$numdep,wage1$smsa,
dnn=c('Num. Dependents','Lives in Metropolitan Area'))
```

```
               Lives in Metropolitan Area
Num. Dependents   0   1
              0  60 192
              1  27  78
              2  38  61
              3  13  32
              4   3  13
              5   3   4
              6   2   0
> |
```

An Example: Dependence

- What are we looking for here?
- For **dependence**, simply see if the distribution of one variable changes for the different values of the other.
- Does the distribution of Number of Dependents differ based on your SMSA status?

```
prop.table(table(wage1$numdep,wage1$smsa,
                 dnn=c('Num. Dependents',
                 'Lives in Metropolitan Area')),
           margin=2)
```

```
                  Lives in Metropolitan Area
Num. Dependents            0          1
              0   0.41095890 0.50526316
              1   0.18493151 0.20526316
              2   0.26027397 0.16052632
              3   0.08904110 0.08421053
              4   0.02054795 0.03421053
              5   0.02054795 0.01052632
              6   0.01369863 0.00000000
> 
```

An Example: Dependence

- Does the distribution of SMSA differ based on your Number of Dependents Status?

```
prop.table(table(wage1$numdep,wage1$smsa,
                 dnn=c('Number of Dependents',
                 'Lives in Metropolitan Area')),
           margin=1)
```

- Looks like it!
- What do these two results mean?

```
                   Lives in Metropolitan Area
Number of Dependents         0         1
                   0 0.2380952 0.7619048
                   1 0.2571429 0.7428571
                   2 0.3838384 0.6161616
                   3 0.2888889 0.7111111
                   4 0.1875000 0.8125000
                   5 0.4285714 0.5714286
                   6 1.0000000 0.0000000
> 
```

An Example: Correlation

- We are interested in whether two variables tend to **move together** (positive correlation) or **move apart** (negative correlation)
- One basic way to do this is to see whether values tend to be **high** together
- One way to check in dplyr is to use ‘group_by()‘ to organize the data into groups
- Then ‘summarize()‘ the data within those groups

```
wage1 %>%
  group_by(smsa) %>%
  summarize(numdep=mean(numdep))
```

- When ‘smsa‘ is high, ‘numdep‘ tends to be low - negative correlation!

```
> wage1 %>%
+   group_by(smsa) %>%
+   summarize(numdep=mean(numdep))
`summarise()` ungrouping output (override with `.groups` argument)
# A tibble: 2 x 2
   smsa numdep
  <dbl>  <dbl>
1     0  1.24
2     1  0.968
>
```

An Example: Correlation

- There's also a summary statistic we can calculate **called** correlation, this is typically what we mean by "correlation"
- Ranges from -1 (perfect negative correlation) to 1 (perfect positive correlation)
- Basically "a one-standard deviation increase in ‘X‘ is associated with a "correlation" standard-deviation increase in ‘Y‘"

```
cor(wage1$numdep,wage1$smsa)
cor(wage1$smsa,wage1$numdep)
```

```
> cor(wage1$numdep,wage1$smsa)
[1] -0.09636769
> cor(wage1$smsa,wage1$numdep)
[1] -0.09636769
>
```

An Example: Explanation

- Let's go back to those different means:

```
wage1 %>%
  group_by(smsa) %>%
  summarize(numdep=mean(numdep))
```

- Explanation would be saying that:
    - If you're in an SMSA, I predict that you have these many dependents

      ```
      mean(filter(wage1,smsa==1)$numdep)
      ```

    - If you're not in an SMSA, I predict that you have these many dependents

      ```
      mean(filter(wage1,smsa==0)$numdep)
      ```

- If you are in an SMSA and have 2 dependents, then only some of those dependents are **explained by SMSA** and some of them are **unexplained by SMSA**
- We'll talk a lot more about this later

Coding Recap

- ‘table(df$var1,df$var2)‘ to look at two variables together
- ‘prop.table(table(df$var1,df$var2))‘ for the proportion in each cell
- ‘prop.table(table(df$var1,df$var2),margin=2)‘ to get proportions within each column
- ‘prop.table(table(df$var1,df$var2),margin=1)‘ to get proportions within each row
- ‘df %>% group_by(var1) %>% summarize(mean(var2))‘ to get mean of var2 for each value of var1
- ‘cor(df$var1,df$var2)‘ to calculate correlation

Graphing Relationships

- Relationships between variables can be easier to see graphically

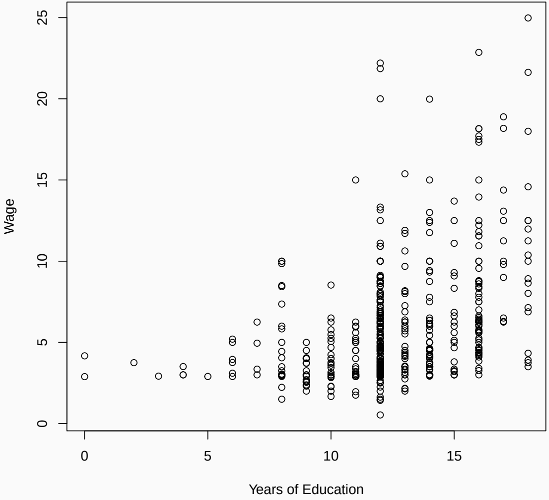
- And graphs are extremely important to understanding relationships and the "shape" of those relationships

Wage and Education

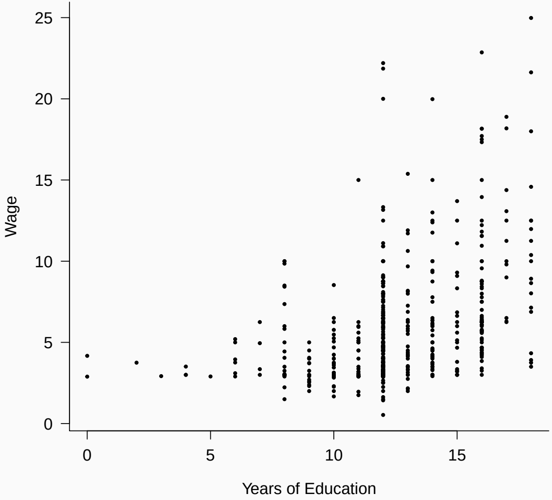
- Let's use ‘plot(xvar,yvar)‘

```
plot(wage1$educ,wage1$wage,xlab="Years of Education",ylab="Wage")
#THE GGPLOT2 WAY
ggplot(wage1,aes(x=educ,y=wage))+geom_point()+
  xlab('Years of Education')+
  ylab('Wage')
```

- As we look at different values of ‘educ‘, what changes about the values of ‘wage‘ we see?



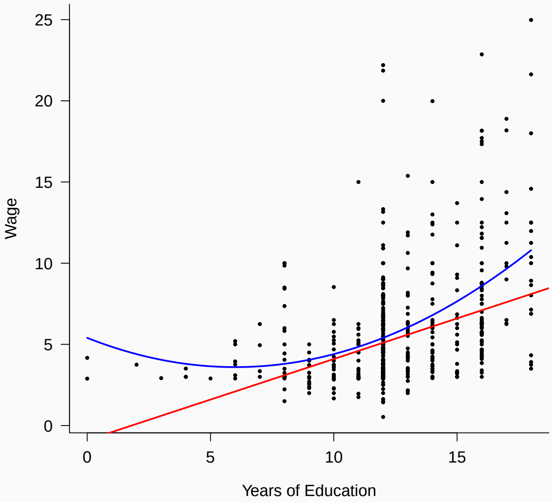
Wage
0
5
10
15
20
25
0
5
10
15
Years of Education


Wage
25
20
15
10
5
0
0
5
10
15
Years of Education

Graphing Relationships

- Try to picture the **shape** of the data
- Should this be a straight line? A curved line? Positively sloped? Negatively?

```
plot(wage1$educ,wage1$wage,xlab="Years of Education",ylab="Wage")
abline(-.9,.5,col='red')
plot(function(x) 5.4-.6*x+.05*(x^2),0,18,add=TRUE,col='blue')


#THE GGPLOT2 WAY
ggplot(wage1,aes(x=educ,y=wage))+geom_point()+
  xlab('Years of Education')+
  ylab('Wage')+
  geom_abline(aes(intercept=-.9,slope=.5),col='red')+
  stat_function(fun=function(x) 5.4-.6*x+.05*(x^2),col='blue')
```



Wage
25
20
15
10
5
0
0
5
10
15
Years of Education

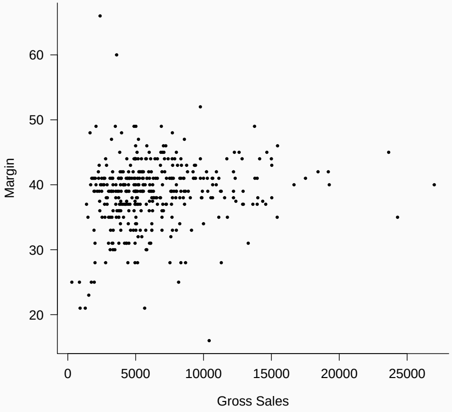
Graphing Relationships

- ‘plot(xvar,yvar)‘ is extremely powerful, and will show you relationships at a glance
- The previous graph showed a clear positive relationship, and indeed ‘cor(wage1$wage,wage1$educ)‘ is 0.406
- Further, we don’t only see a positive relationship, but we have some sense of **how** positive it is, what it looks like roughly

Graphing Relationships

- Let's compare clothing sales volume vs. profit margin for men's clothing firms

```
library(Ecdat)
data(Clothing)
plot(Clothing$sales,Clothing$margin,xlab="Gross Sales",ylab="Margin")

#THE GGPLOT2 WAY
library(Ecdat)
data(Clothing)
ggplot(Clothing,aes(x=sales,y=margin))+geom_point()+
  xlab('Gross Sales')+
  ylab('Margin')
```

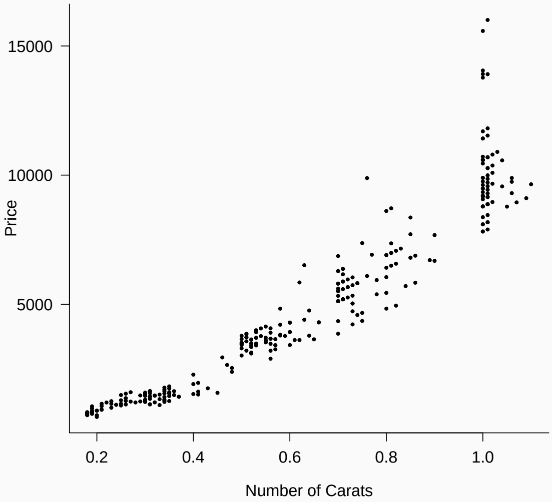


No clear relationship (although correlation is 0.137) but variance is higher for low sales

Graphing Relationships

- Comparing Singapore diamond prices vs. carats

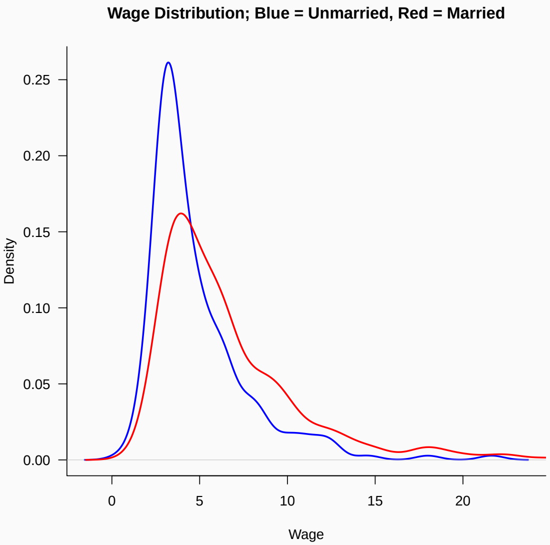
```
library(Ecdat)
data(Diamond)
plot(Diamond$carat,Diamond$price,xlab="Number of Carats",ylab="Price")
#THE GGPLOT2 WAY
library(Ecdat)
data(Diamond)
ggplot(Diamond,aes(x=carat,y=price))+geom_point()+
  xlab('Number of Carats')+
  ylab('Price')
```


15000
10000
5000
Price
0.2
0.4
0.6
0.8
1.0
Number of Carats

Graphing Relationships

- Another way to graph a relationship, especially when one of the variables only takes a few values, is to plot the ‘density()‘ function for different values

```
plot(density(filter(wage1,married==0)$wage),col='blue',
     xlab="Wage",
     main="Wage Distribution; Blue = Unmarried, Red = Married",
     bty="L",las=1,lwd=2)
lines(density(filter(wage1,married==1)$wage),col='red',lwd=2)
#THE GGPLOT2 WAY
ggplot(filter(wage1,married==0),aes(x=wage))+stat_density(geom='line',co
  xlab('Wage')+
  ylab('Density')+
  ggtitle("Wage Distribution; Blue = Unmarried, Red = Married")+
  stat_density(data=filter(wage1,married==1),geom='line',col='red')
```


Wage Distribution; Blue = Unmarried, Red = Married
Density
0.00
0.05
0.10
0.15
0.20
0.25
0
5
10
15
20
Wage

Graphing Relationships

- Different distributions: married people earn more!

- We can back that up other ways

```
wage1 %>% group_by(married) %>% summarize(wage = mean(wage))
cor(wage1$wage,wage1$married)
```

```
> wage1 %>% group_by(married) %>% summarize(wage = mean(wage))
`summarise()` ungrouping output (override with `.groups` argument)
# A tibble: 2 x 2
  married  wage
    <dbl> <dbl>
1       0  4.84
2       1  6.57
> cor(wage1$wage,wage1$married)
[1] 0.2288172
>
```

Keep in mind!

- Just because two variables are **related** doesn't mean we know **why**
- If ‘cor(x,y)‘ is positive, it could be that ‘x‘ causes ‘y‘... or that ‘y‘ causes ‘x‘, or that something else causes both!
- Or many other configurations...
- Plus, even if we know the direction we may not know **why** that cause exists.

Practice

- Install the ‘SMCRM‘ package, load it, get the ‘customerAcquisition‘ data. Rename it ca
- Among ‘acquisition==1‘ observations, see if the size of first purchase is related to duration as a customer, with ‘cor‘ and (labeled) ‘plot‘
- See if ‘industry‘ and ‘acquisition‘ are dependent on each other using ‘prop.table‘ with the ‘margin‘ option
- See if average revenues differ between industries using ‘aggregate‘, then check the ‘cor‘
- Plot the density of revenues for ‘industry==0‘ in blue and, on the same graph, revenues for ‘industry==1‘ in red
- In each case, think about relationship is suggested

Practice Answers

```
install.packages('SMCRM')
library(SMCRM)
data(customerAcquisition)
ca <- customerAcquisition
cor(filter(ca,acquisition==1)$first_purchase,filter(ca,acquisition==1)$duration)
plot(filter(ca,acquisition==1)$first_purchase,filter(ca,acquisition==1)$duration,
     xlab="Value of First Purchase",ylab="Customer Duration")
prop.table(table(ca$industry,ca$acquisition),margin=1)
prop.table(table(ca$industry,ca$acquisition),margin=2)
aggregate(revenue~industry,data=ca,FUN=mean)
cor(ca$revenue,ca$industry)
plot(density(filter(ca,industry==0)$revenue),col='blue',xlab="Revenues",main="Revenue Distribution")
lines(density(filter(ca,industry==1)$revenue),col='red')
```

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Anscome's Quartet

- Developed by F.J. Anscombe in 1973
- Anscombe's Quartet is a set of four datasets, all with the same summary statistics (mean, standard deviation, and correlation)

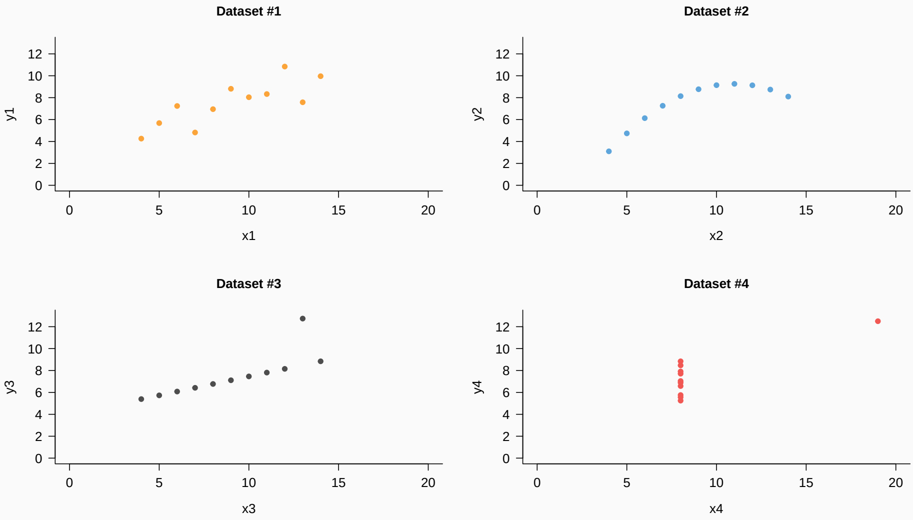
Anscome's Quartet

```r
library(datasets)
datasets::anscombe
stargazer(anscombe,type='text',summary.stat = c("n","mean","sd"))
cor(anscombe$x1,anscombe$y1)
cor(anscombe$x2,anscombe$y2)
cor(anscombe$x3,anscombe$y3)
cor(anscombe$x4,anscombe$y4)
```

```
> library(datasets)
> datasets::anscombe
   x1 x2 x3 x4    y1   y2    y3    y4
1  10 10 10  8  8.04 9.14  7.46  6.58
2   8  8  8  8  6.95 8.14  6.77  5.76
3  13 13 13  8  7.58 8.74 12.74  7.71
4   9  9  9  8  8.81 8.77  7.11  8.84
5  11 11 11  8  8.33 9.26  7.81  8.47
6  14 14 14  8  9.96 8.10  8.84  7.04
7   6  6  6  8  7.24 6.13  6.08  5.25
8   4  4  4 19  4.26 3.10  5.39 12.50
9  12 12 12  8 10.84 9.13  8.15  5.56
10  7  7  7  8  4.82 7.26  6.42  7.91
11  5  5  5  8  5.68 4.74  5.73  6.89
> stargazer(anscombe,type='text',summary.stat = c("n","mean","sd"))

===========================
Statistic N  Mean  St. Dev.
---------------------------
x1        11 9.000  3.317
x2        11 9.000  3.317
x3        11 9.000  3.317
x4        11 9.000  3.317
y1        11 7.501  2.032
y2        11 7.501  2.032
y3        11 7.500  2.030
y4        11 7.501  2.031
---------------------------
> cor(anscombe$x1,anscombe$y1)
[1] 0.8164205
> cor(anscombe$x2,anscombe$y2)
[1] 0.8162365
> cor(anscombe$x3,anscombe$y3)
[1] 0.8162867
> cor(anscombe$x4,anscombe$y4)
[1] 0.8165214
> 
```

Anscome's Quartet

```
par(mfrow=c(2,2))
plot(anscombe$x1,anscombe$y1,pch=19,col="#FAA43A",
     bty="L",xlim=c(0,20),ylim=c(0,13),
     xlab="x1",ylab="y1",main="Dataset #1",
     cex.lab=1.2,cex.axis=1.2, las=1)
plot(anscombe$x2,anscombe$y2,pch=19,col="#5DA5DA",
     bty="L",xlim=c(0,20),ylim=c(0,13),
     xlab="x2",ylab="y2",main="Dataset #2",
     cex.lab=1.2,cex.axis=1.2, las=1)
plot(anscombe$x3,anscombe$y3,pch=19,col="#4D4D4D",
     bty="L",xlim=c(0,20),ylim=c(0,13),
     xlab="x3",ylab="y3",main="Dataset #3",
     cex.lab=1.2,cex.axis=1.2, las=1)
plot(anscombe$x4,anscombe$y4,pch=19,col="#F15854",
     bty="L",xlim=c(0,20),ylim=c(0,13),
     xlab="x4",ylab="y4",main="Dataset #4",
     cex.lab=1.2,cex.axis=1.2, las=1)
```


Dataset #1
y1
x1
Dataset #2
y2
x2
Dataset #3
y3
x3
Dataset #4
y4
x4

Datasaurus Dozen datasets

- It's like Anscome's Quartet on steroids
- The original Datasaurus was created by Alberto Cairo (see `http://www.thefunctionalart.com/2016/08/download-datasaurus-never-trust-summary.html`)
- The other twelve were created by Justin Matejka and George Fitzmaurice (see `https://www.autodeskresearch.com/publications/samestats`)
- The code/data in in R via `https://github.com/lockedata/datasauRus`

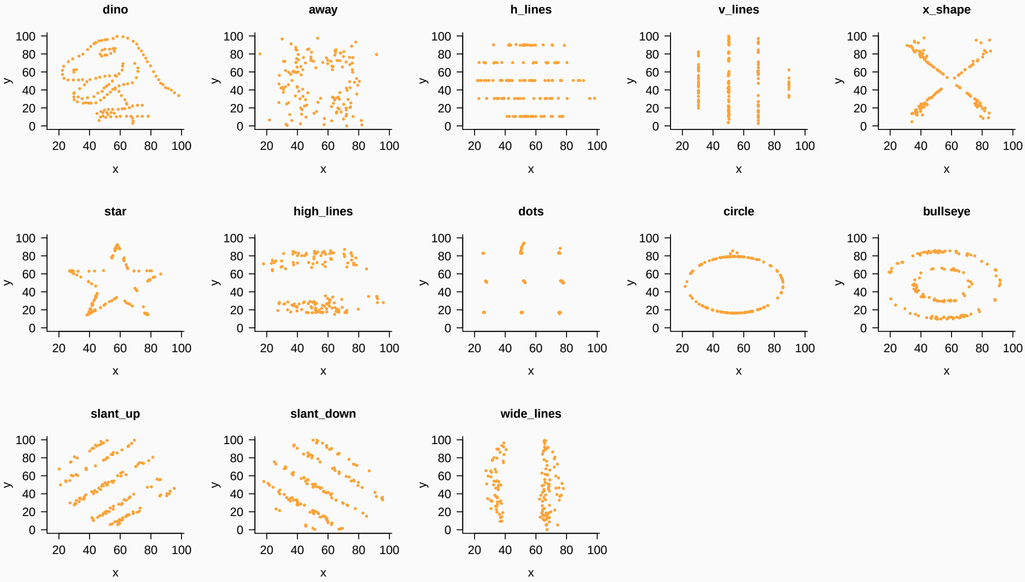
Datasaurus Dozen datasets

```
datasaurus_dozen %>%
  group_by(dataset) %>%
  summarize(
    mean_x    = mean(x),
    mean_y    = mean(y),
    std_dev_x = sd(x),
    std_dev_y = sd(y),
    corr_x_y  = cor(x, y)
  )
```

```
`summarise()` ungrouping output (override with `.groups` argument)
# A tibble: 13 x 6
   dataset    mean_x mean_y std_dev_x std_dev_y corr_x_y
   <chr>       <dbl>  <dbl>     <dbl>     <dbl>    <dbl>
 1 away         54.3   47.8      16.8      26.9  -0.0641
 2 bullseye     54.3   47.8      16.8      26.9  -0.0686
 3 circle       54.3   47.8      16.8      26.9  -0.0683
 4 dino         54.3   47.8      16.8      26.9  -0.0645
 5 dots         54.3   47.8      16.8      26.9  -0.0603
 6 h_lines      54.3   47.8      16.8      26.9  -0.0617
 7 high_lines   54.3   47.8      16.8      26.9  -0.0685
 8 slant_down   54.3   47.8      16.8      26.9  -0.0690
 9 slant_up     54.3   47.8      16.8      26.9  -0.0686
10 star         54.3   47.8      16.8      26.9  -0.0630
11 v_lines      54.3   47.8      16.8      26.9  -0.0694
12 wide_lines   54.3   47.8      16.8      26.9  -0.0666
13 x_shape      54.3   47.8      16.8      26.9  -0.0656
> 
```

Datasaurus Dozen datasets

```
install.packages("datasauRus")
library(datasauRus)
par(mfrow=c(3,4))
for(data in unique(datasaurus_dozen$dataset)){
  print(data)
  df=datasaurus_dozen[which(datasaurus_dozen$dataset==data),]

  plot(df$x,df$y,pch=19,col="#FAA43A",
       bty="L",xlim=range(datasaurus_dozen$x),
       ylim=range(datasaurus_dozen$y),
       xlab="x",ylab="y",main=data,cex=0.5,
       cex.lab=1.2,cex.axis=1.2, las=1)
}
```


dino
away
h_lines
v_lines
x_shape
star
high_lines
dots
circle
bullseye
slant_up
slant_down
wide_lines
y
x
0
20
40
60
80
100

Summarizing data

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Remember...

Plotting the distribution

Inferring things about the distribution

Relationships Between Variables

Don't trust statistics alone, visualize your data!

Simulations and relationship between variables

Let’s Simulate!

- Let’s expand our use of simulation to simulate the relationship between **two** variables
- We can do this by using one variable to build another
- I draw 400 random genders, and add to them 400 random normals)

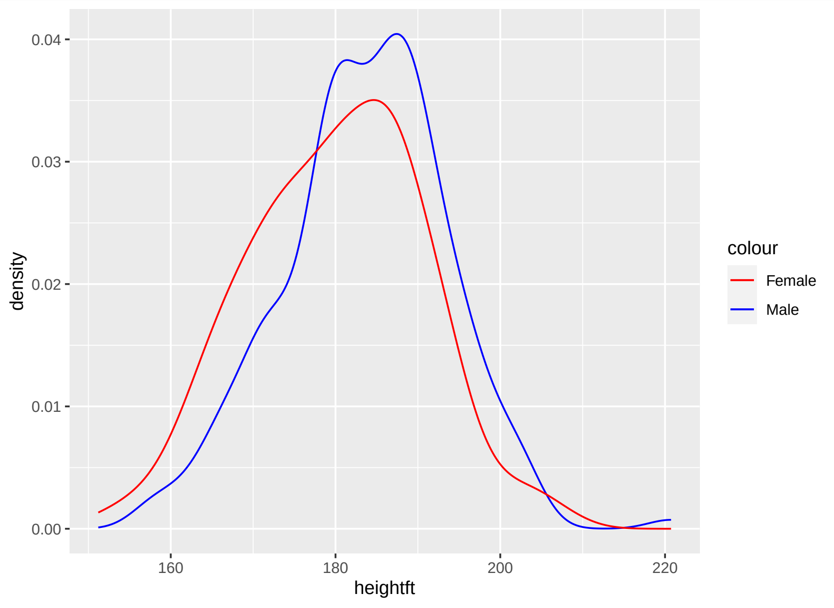
```
# 400 people equally likely to be M or F
simdata <- data.frame(gender = sample(c("Male","Female"),400,replace=T))

 # Height is normally distributed with mean 180cm and sd 10cm
 #and men are 5cm of a foot taller
simdata <- simdata %>%mutate(heightft = rnorm(400,180,10)+5*(gender == "Male"))

simdata %>% group_by(gender) %>% summarize(height = mean(heightft))
```

Two Variable Simulation

- We get in our simulation that men are on average ‘mean(filter(simdata,gender=="Male")$heightft)-mean(filter(simdata,gender=="Female")$heightft)‘ taller than women.

- The true data-generating process is that heightft is a normal variable with mean 180, plus 5cm if you’re male

```
ggplot(filter(simdata,gender=="Male"),aes(x=heightft,col='Male'))+
stat_density(geom='line')+
stat_density(data=filter(simdata,gender=="Female"),aes(x=heightft,col='Female'),geom='line')+
scale_color_manual(values=c('red','blue'))
```


0.04
0.03
0.02
0.01
0.00
density
160
180
200
220
heightft
colour
Female
Male

Two Variable Simulation

So does checking for the difference of means give us back the difference in height from the data-generating process? Let's loop!

```
heightdiff <- c()
for (i in 1:2500) {
  simdata <- data.frame(gender = sample(c("Male","Female"),400,replace=T))

  # Height is normally distributed with mean 180cm and sd 10cm
  #and men are 5cm of a foot taller
  simdata <- simdata %>%mutate(heightft = rnorm(400,180,10)+5*(gender == "Male"))

  simdata %>% group_by(gender) %>% summarize(height = mean(heightft))

  heightdiff[i] <- mean(filter(simdata,gender=="Male")$heightft)-
    mean(filter(simdata,gender=="Female")$heightft)
}

stargazer(as.data.frame(heightdiff),type='text')
```

```
> stargazer(as.data.frame(heightdiff),type='text')

==========================================================================
Statistic     N   Mean  St. Dev.  Min  Pctl(25) Pctl(75)  Max
--------------------------------------------------------------------------
heightdiff 2,500 4.977  0.979   0.957  4.341    5.622   8.019
--------------------------------------------------------------------------
```

Another Example

- So far, no problem, right? Everything works out (I mean, of course it does)
- Of **course** the average number of heads in a sample will on average be 50%
- So what can we actually learn here?
- It may help to see an example where we get the **wrong** answer

Another Example

```
# Is your company in tech? Let's say 30% of firms are
df <- data.frame(tech = sample(c(0,1),500,replace=T,prob=c(.7,.3)))
#Tech firms on average spend $3mil more defending IP lawsuits
df <- df%>% mutate(IP.spend = 3*tech+runif(500,min=0,max=4))
# Now let's check for how profit and IP.spend are correlated!
df <- df%>%mutate(log.profit = 2*tech - .3*IP.spend + rnorm(500,mean=2))
cor(df$log.profit,df$IP.spend)
```

- Uh-oh! Truth is negative relationship, but data says positive (0.109)!!

Another Example

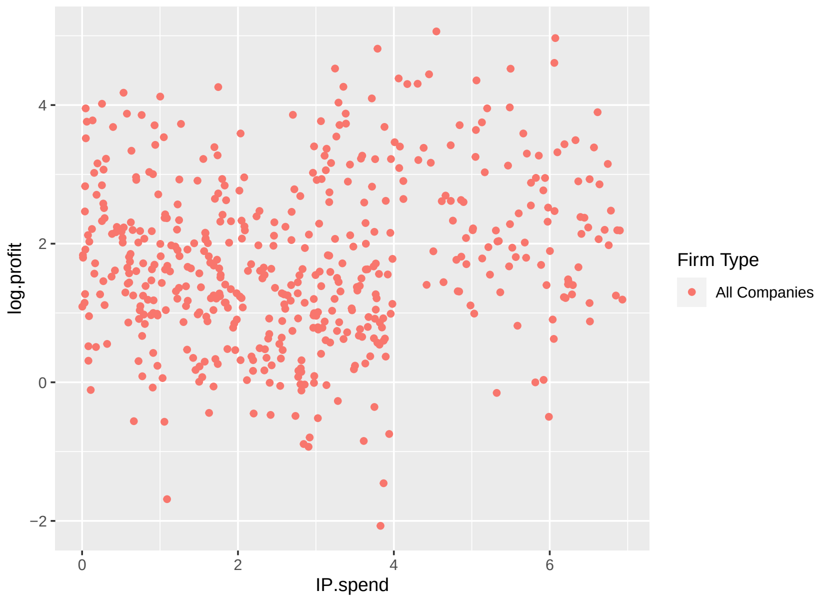
Maybe just a fluke? Let's loop.

```
IPcorr <- c()
for (i in 1:1000) {
  # Is your company in tech? Let's say 30% of firms are
  df <- data.frame(tech = sample(c(0,1),500,replace=T,prob=c(.7,.3)))
  #Tech firms on average spend $3mil more defending IP lawsuits
  df <- df%>% mutate(IP.spend = 3*tech+runif(500,min=0,max=4))
  # Now let's check for how profit and IP.spend are correlated!
  df <- df%>%mutate(log.profit = 2*tech - .3*IP.spend + rnorm(500,mean=2))
  IPcorr[i] <- cor(df$log.profit,df$IP.spend)
}

stargazer(as.data.frame(IPcorr),type='text')
```

```
> stargazer(as.data.frame(IPcorr),type='text')

=================================================================
Statistic   N    Mean  St. Dev.  Min  Pctl(25) Pctl(75)  Max
-----------------------------------------------------------------
IPcorr    1,000 0.140  0.042   0.013  0.113    0.169   0.251
-----------------------------------------------------------------
>
```

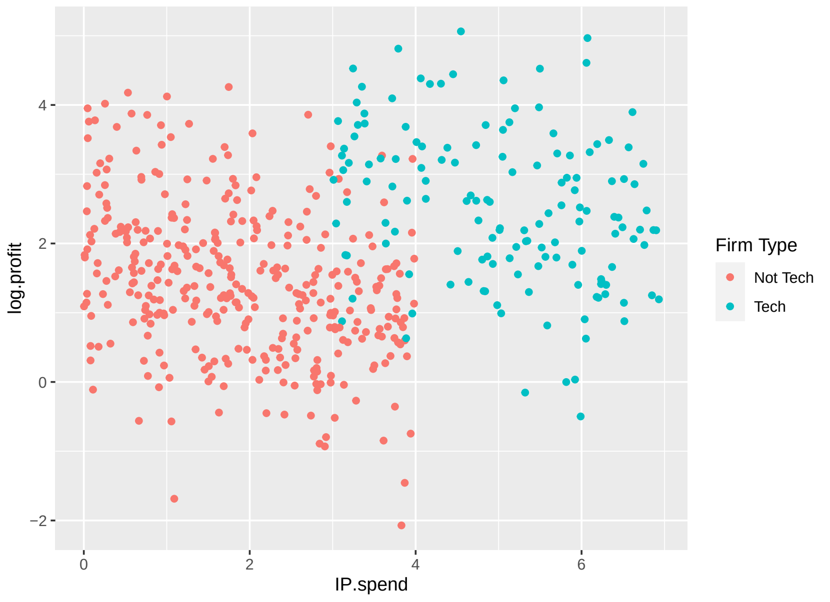
What's Happening? A graph might help

```
ggplot(df,aes(x=IP.spend,y=log.profit,color=as.factor("All Companies")))+
geom_point()+
  guides(color=guide_legend(title="Firm Type"))
```


log.profit
IP.spend
4
2
0
−2
0
2
4
6
Firm Type
All Companies

What’s Happening? A graph might help

```
ggplot(mutate(df,tech=factor(tech,labels=c("Not Tech","Tech"))),
              aes(x=IP.spend,y=log.profit,color=tech))+geom_point()+
  guides(color=guide_legend(title="Firm Type"))
```



log.profit
IP.spend
Firm Type
Not Tech
Tech

Simpson's Paradox

- Here we have a true negative relationship - we know it's in the true model!
- But when we plot it out, it's positive
- WITHIN tech companies and non-tech companies, IP spend is negatively correlated with profit
- But because tech companies have higher IP spend and higher profit, they're positively correlated!
- This is known as "Simpson's Paradox" and shows up in many places

Simpson's Paradox

- So our method (looking at the correlation between them) doesn't work!
- The simulation has shown us that we'd get it wrong if we do this
  - Our analysis method doesn't correct for what tech is doing here
- We need to have some way of incorporating what we know about tech
- Taking what we might know about the true model - that firm type has something to do with this, and adjusting so we get the right answer
- This sort of thinking is what we'll be getting into

Practice

- Use the ‘prob‘ option in ‘sample‘ to generate 300 coin flips weighted to be 55% heads. Calculate heads prop.
- Loop it 2000 times. How often will you correctly claim that the coin is more than 50% heads?
- Create ‘dat‘: 1500 obs of ‘married‘ (0 or 1), ‘educ‘ (unif 0 to 16, plus ‘2*married‘), ‘log.wage‘ (normal mean 5 plus ‘.1*educ‘ plus ‘2*married‘)
- Loop it 1000 times and calculate ‘cor‘ between ‘educ‘ and ‘log.wage‘ each time.
- It’s positive - does that mean it’s right? If not, how do you know?
- Use ‘plot‘ and then ‘points‘ to plot the ‘married==0‘ and ‘married==1‘ data in different colors