Measurement and visualization, 2

Frank Edwards

10/1/2019

Last time

Visualizing data

- ggplot

```
ggplot(ipv, ## object, usually a data.frame
       aes(x = region)) + ## aesthetic variables, generally x, y, color, etc
  geom_bar() ## a geom to plot the aesthetics

## Note that + in ggplot() works the same way as %>% in tidyverse:
## It strings together commands, evaluated in sequence
```

- geom_bar(), geom_histogram(), geom_boxplot(), geom_point()
- Wickham Chapter 3:
  `https://r4ds.had.co.nz/data-visualisation.html`
- Recommended reading: Kieran Healy, *Data Visualization*. Available free at socviz.co
- Available workshops on tidyverse, ggplot, rmarkdown at New Brunswick

Today: Measurement in the social sciences

- Survey methods with randomization
- Administrative data and agency surveys
- Unit, item non-response
- Desirability bias
- Latent variables, latent groups
- More visualization

Surveys and censuses

- A census records information about a population, with measurement for each individual or unit in the population
- A survey samples from a population to make an inference about population characteristics

The basic motivation for survey sampling

```r
n <- 1e+06
marbles <- data.frame(color = c(rep("blue", n), rep("green", n),
    n)))
table(marbles$color)
```

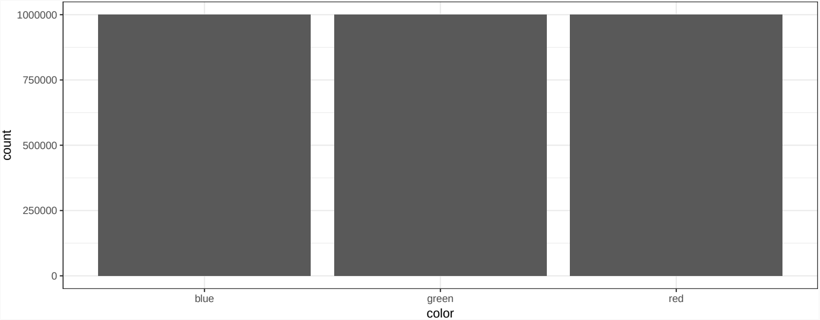
```
##
##    blue   green     red
## 1000000 1000000 1000000
```

How could we know how many of each color are in the enormous bag of marbles?

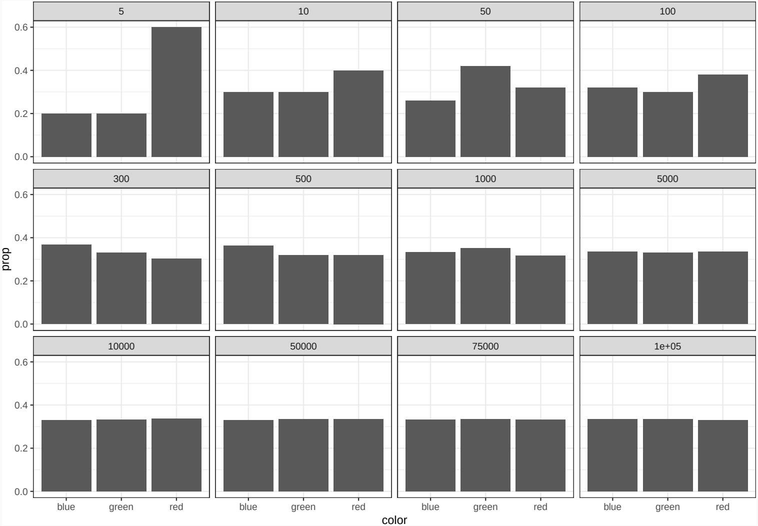
- Count them all (tedious!)
- Sample

The Truth

```r
ggplot(marbles, aes(x = color)) + geom_bar()
```



How many random draws is enough to accurately measure the characteristics of 3 million marbles?



Random sampling

With a sufficiently large sample and equal probability of sampling for all units in the population, a simple random sample allows for unbiased measurement of population characteristics.

Such a sample is representative of the population across both measured and unmeasured characteristics

If you are sampling, it should (in general) be randomized

Stratified random sampling

If we wish to learn about particular sub-populations (i.e. geographies), we can use multi-stage or stratified sampling

1. Randomly sample larger units (geographic) or select larger units of interest purposively
2. Randomly sample individuals within these larger units

EXAMPLE: The American Community Survey (simplified)

1. Take a list of all US Census tracts
2. Randomly sample households within tract based on complete list of addresses (sampling frame)
3. Randomly sample adults within household, conduct survey

When surveys go wrong

1. Unit non-response
2. Item non-response
3. Lying

Unit non-response

Individual (or organization) doesn’t respond to the survey

- How are surveys actually administered?
- Response rates are generally low (and decreasing!)

Completely random non-response is not a problem.

Unit non-response

Individual (or organization) doesn't respond to the survey

- How are surveys actually administered?
- Response rates are generally low (and decreasing!)

Completely random non-response is not a problem.

When would non-response be an issue?

Question non-response

Individual takes the survey, but refuses to answer (skips) a particular question

- Why might this occur?

Question non-response

Individual takes the survey, but refuses to answer (skips) a particular question

- Why might this occur?
- When would this be a problem?

Lying (ok... misrepresentation)

- Social desirability bias
  - Did you vote? Remember HW 1?
  - Are you a racist?
  - What kinds of crimes do you like to do?

Examining non-response in a survey of exposure to violence in Afghanistan

Load the data

```r
library(qss)
data(afghan)
data(afghan.village)
```

Explore the variables in afghan

```r
table(afghan$province)
```

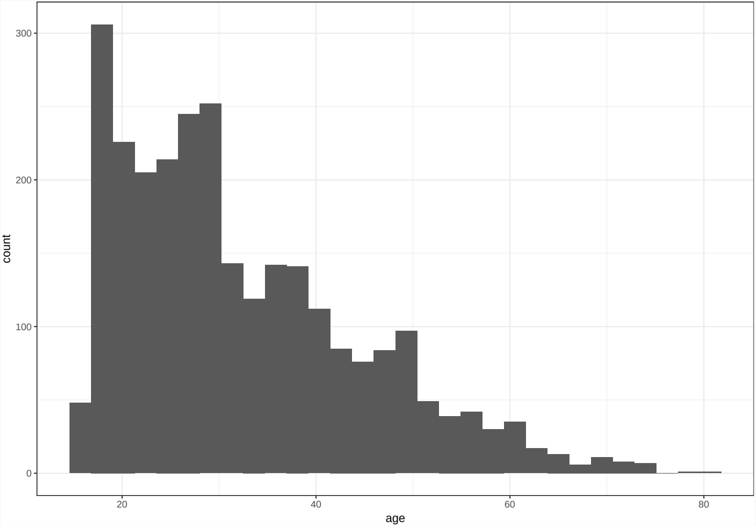
```
##
## Helmand   Khost   Kunar   Logar Uruzgan
##     855     630     396     486     387
```

```r
table(afghan$district)
```

```
##
##        Asadabad             Bak    Baraki Barak      Chapa Dara          Dangam
##              54              54             180             108              63
##        Dihrawud         Garmser       Ghaziabad    Khas Uruzgan          Khoshi
##             117             225              63             117              54
##           Khost    Lashkar Gah       Musa Qala         Naw Zad       Puli Alam
##             243             108             225             216             252
##        Qalandar Shahidi Hassas           Spira            Tani          Washer
##              63             153              99             171              81
##        Wata Pur
##             108
```

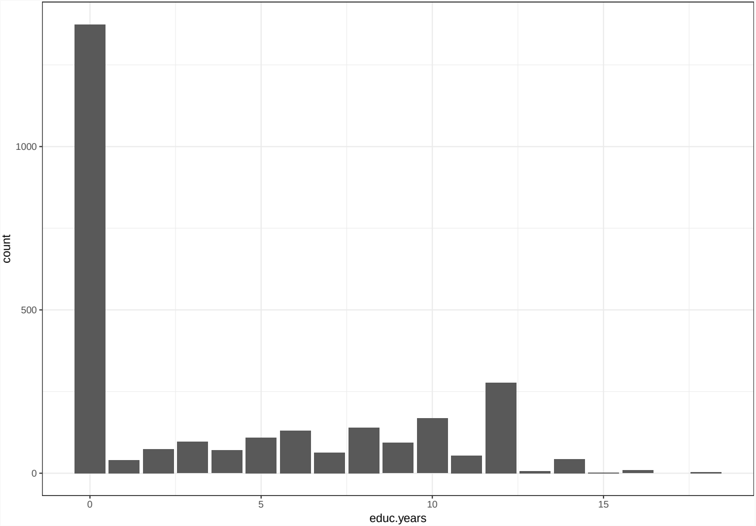
Explore the variables in afghan

```
ggplot(afghan, aes(x = age)) + geom_histogram()
```



Explore the variables in afghan

```
ggplot(afghan, aes(x = educ.years)) + geom_bar()
```



Explore the variables in afghan

```r
table(afghan$employed)
```

```
##
##    0    1
## 1149 1605
```

Explore the variables in afghan

```
table(afghan$income)
```

```
##
##   10,001-20,000    2,001-10,000   20,001-30,000 less than 2,000
##             616            1420              93             457
##     over 30,000
##              14
```

```
## for ordered categorical
afghan <- afghan %>% mutate(income = factor(income, levels = c("less than 2,000",
    "2,001-10,000", "10,001-20,000", "20,001-30,000", "over 30,000")))

table(afghan$income)
```

```
##
## less than 2,000    2,001-10,000   10,001-20,000   20,001-30,000
##             457            1420             616              93
##     over 30,000
##              14
```

Explore the variables in afghan

```r
afghan %>% select(employed, violent.exp.ISAF, violent.exp.taliba
```

```
##     employed       violent.exp.ISAF violent.exp.taliban
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :1.0000   Median :0.0000   Median :0.0000
##  Mean   :0.5828   Mean   :0.3749   Mean   :0.3289
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##                   NA's   :25       NA's   :54
```

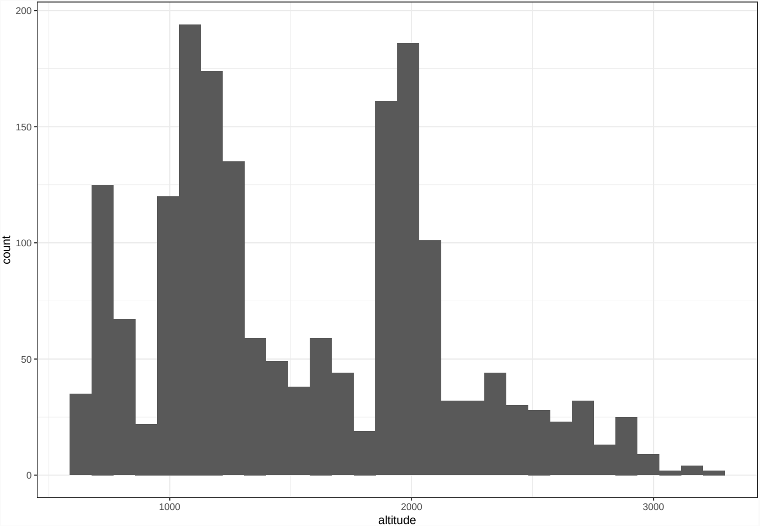
Explore the variables in afghan.village

```r
head(afghan.village)
```

```
##   altitude population village.surveyed
## 1  1959.08        197                1
## 2  2425.88        744                0
## 3  2236.60        179                1
## 4  1691.76        225                0
## 5  1928.04        379                0
## 6  1194.56        617                0
```

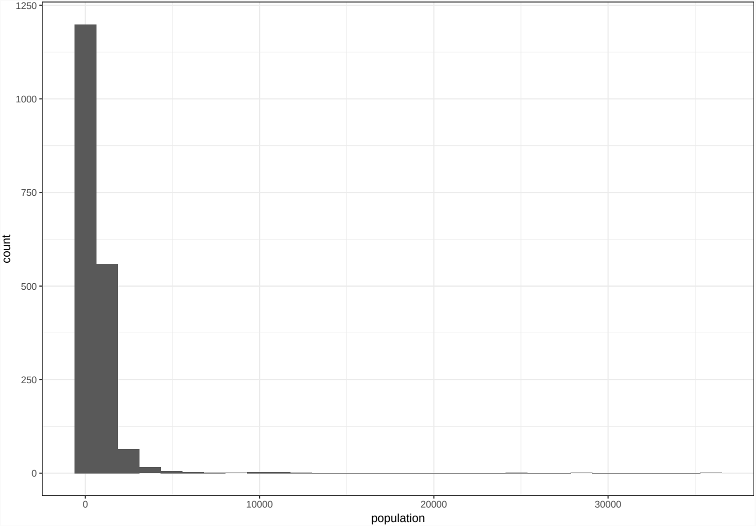
Explore the variables in afghan.village

```
ggplot(afghan.village, aes(x = altitude)) + geom_histogram()
```



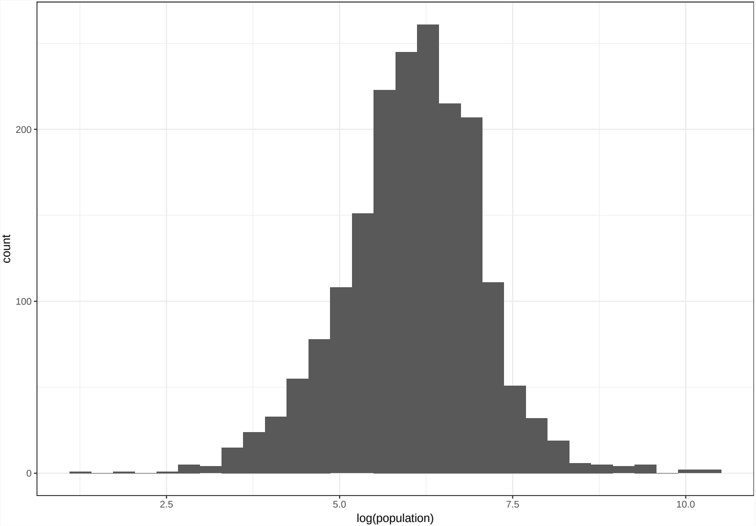
Explore the variables in afghan.village

```
ggplot(afghan.village, aes(x = population)) + geom_histogram()
```



Explore the variables in afghan.village: logs help!

```
ggplot(afghan.village, aes(x = log(population))) + geom_histogra
```



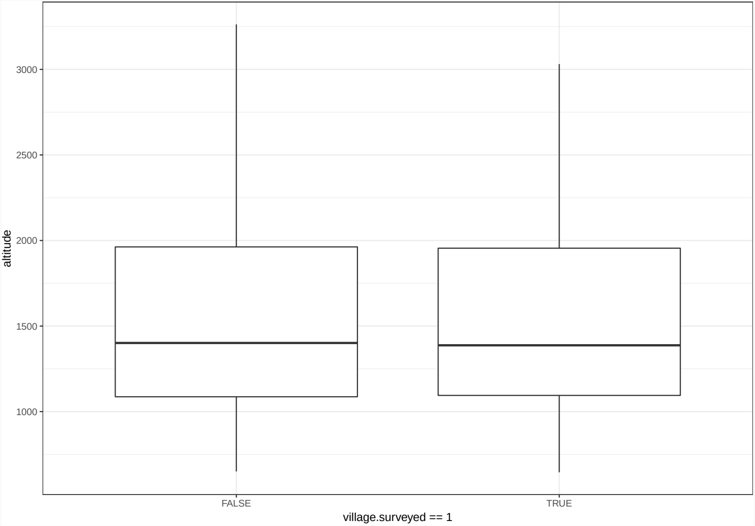
Explore the variables in afghan.village

```
mean(afghan.village$village.surveyed)
```

```
## [1] 0.1094421
```

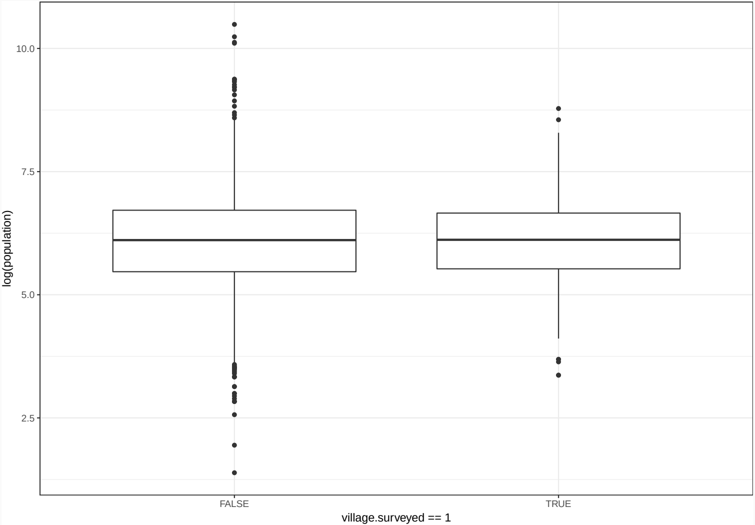
Is the sampling representative of villages?

```
ggplot(afghan.village, aes(x = village.surveyed == 1, y = altitu
```

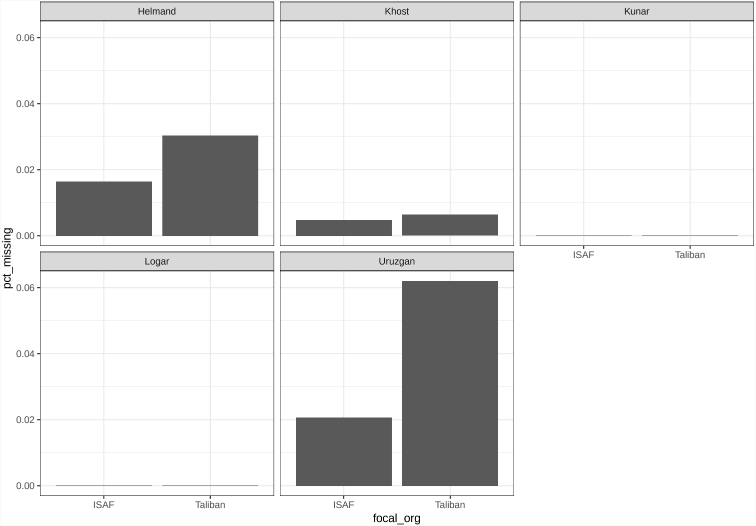


Is the sampling representative of villages?

```
ggplot(afghan.village, aes(x = village.surveyed == 1, y = log(po
    geom_boxplot()
```



Does item non-response bias estimates of violence by region?



Summary

- Unit non-responses can bias survey estimates
- Item non-response can bias survey estimates
- Social desirability can bias survey estimates
- Errors induced by these biases can lead to incorrect conclusions (see polling consensus on 2016 election)

Returning to the IPV example

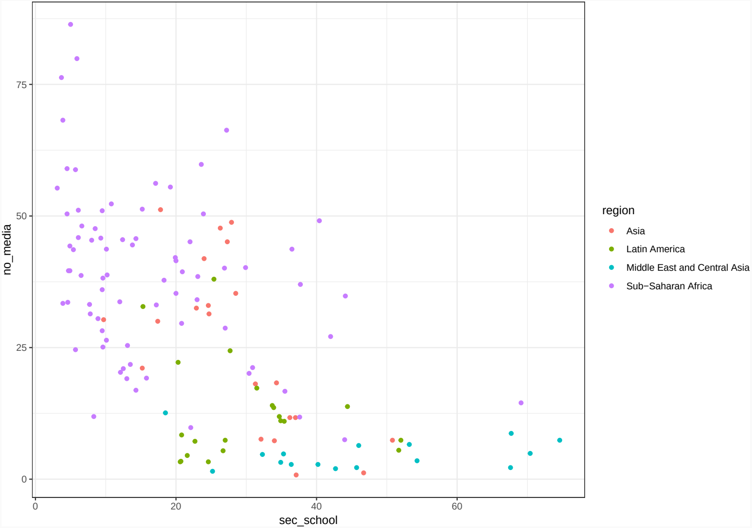
Load the data

```
ipv <- read_csv("./data/dhs_ipv.csv")
## on your machine, path is /slides/data/
head(ipv)
```

```
## # A tibble: 6 x 8
##      X1 beat_burnfood beat_goesout sec_school no_media countr
##   <dbl>         <dbl>        <dbl>      <dbl>    <dbl> <chr>
## 1     1           4.4         18.6       25.2      1.5 Albani
## 2     4           4.9         19.9       67.7      8.7 Armeni
## 3     5           2.1         10.3       67.6      2.2 Armeni
## 4     6           0.3          3.1       46        6.4 Armeni
## 5     7          12.1         42.5       74.6      7.4 Azerba
## 6     8          NA           NA         24       41.9 Bangla
```

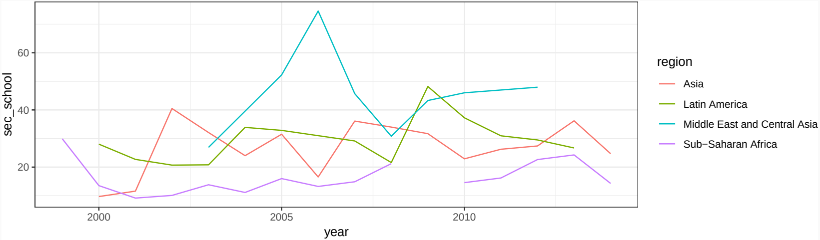
Look at bivariate relationships

```
ggplot(ipv, aes(x = sec_school, y = no_media, color = region)) + geom_point()
```

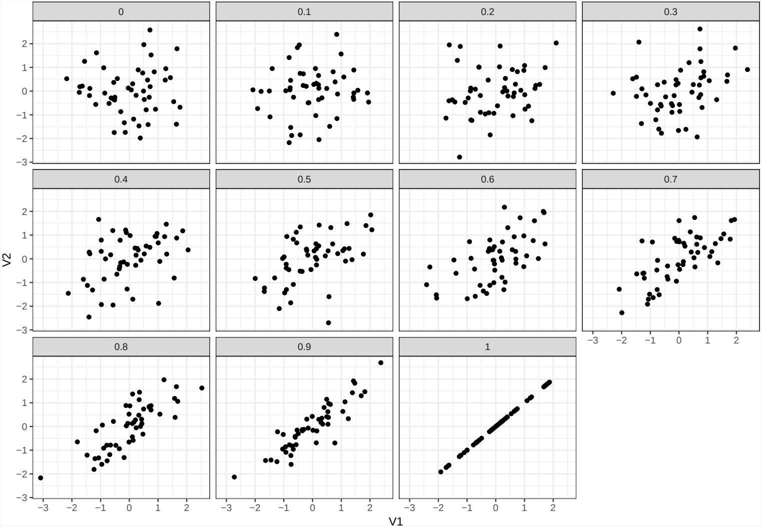


Is there a change in sec_school by region over time across this sample? Does time matter here?

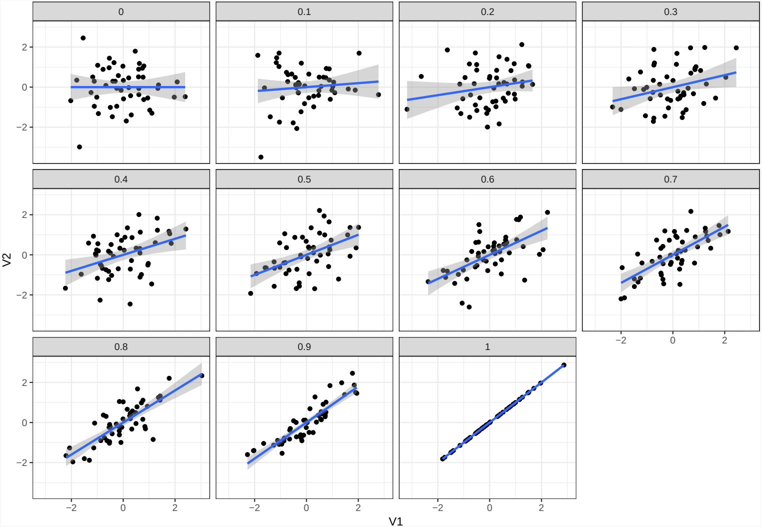
```
ipv_ts <- ipv %>% group_by(region, year) %>% summarise(sec_school = mean(sec_school))

ggplot(ipv_ts, aes(x = year, y = sec_school, color = region)) + geom_line()
```



Correlation



Correlation



Correlation (math time): Z-scores

First, we need the variables to be comparable, so we transform them to be on a standard deviation scale.

A *z*-score scales a variable measures the number of standard deviations an observation is away from it's mean.

$$z \text{ score of } x_i = \frac{x_i - \bar{x}}{S_x}$$

Where $\bar{x}$ is the mean, and S_x is the standard deviation of variable x. Z scores have a mean zero, and a range defined by the range of the data on a standard deviation scale.

For a normally (Gaussian) distributed variable, this will typically range between $[-3, 3]$

In R, we can transform a numeric into a *z*-score using `scale()`

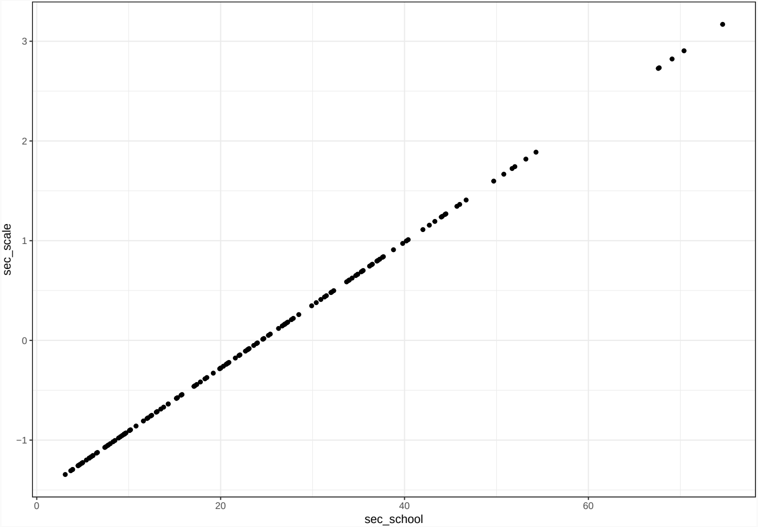
Z-scores in R

```
ipv_scale <- ipv %>% mutate(sec_scale = scale(sec_school)) %>% s
    sec_scale)
summary(ipv_scale)
```

```
##    sec_school        sec_scale.V1
##  Min.   : 3.10   Min.   :-1.345006
##  1st Qu.:10.18   1st Qu.:-0.898292
##  Median :22.40   Median :-0.126408
##  Mean   :24.40   Mean   : 0.000000
##  3rd Qu.:34.90   3rd Qu.: 0.662840
##  Max.   :74.60   Max.   : 3.169492
##  NA's   :3       NA's   :3
```

Z-scores in R

```
ggplot(ipv_scale, aes(x = sec_school, y = sec_scale)) + geom_point()
```



Correlation

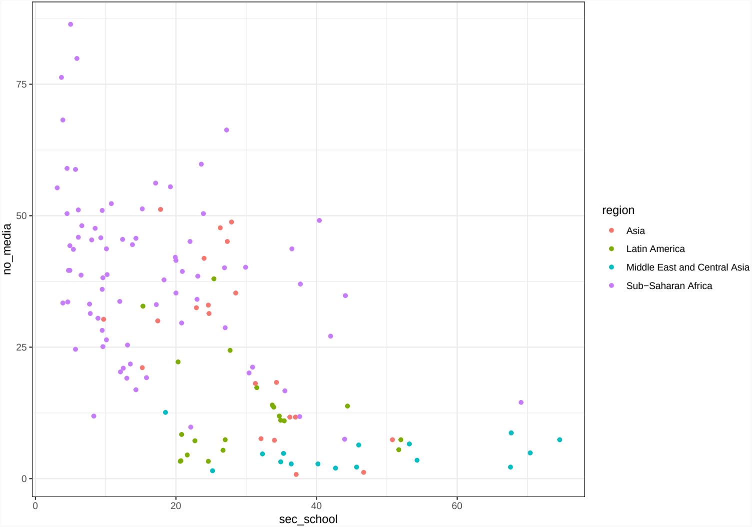
Correlation measures the degree to which two variables are associated with each other. We often use the letter r to denote a correlation.

$$r(x, y) = \frac{1}{n} \sum_{i=1}^{n} \frac{x_i - \overline{x}}{S_x} \times \frac{y_i - \overline{y}}{S_y}$$

Note that this is equal to the average of the product of the $z - scores$ of x and y

In R, you can use `cor()`

Returning to our example: Are sec_school and no_media correlated?



Obtaining the correlation coefficient

```r
cor(ipv$sec_school, ipv$no_media, use = "complete")
```

```
## [1] -0.6077951
```

```r
## z score method
mean(scale(ipv$sec_school) * scale(ipv$no_media), na.rm = TRUE)
```

```
## [1] -0.6084724
```

Latent structure

Data often *cluster* based on unobserved or unobservable characteristics. We can use *classification methods* to try to uncover these latent structures in data.

k-means is a straightforward method we can use to identify k latent groupings in our data, based on proximity of observations for specified variables.

The k-means algorithm

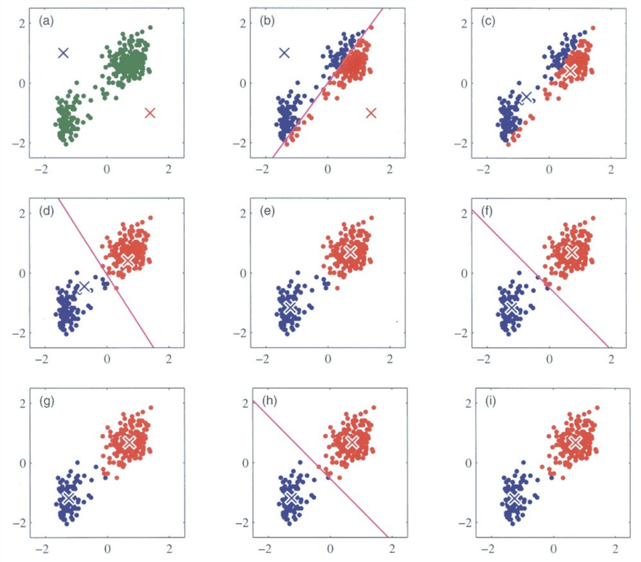
An algorithm is a sequential set of steps used to solve a problem.

A *centroid* is the mean value of a cluster within a group.

1. Choose the initial centroids for each of the k clusters
2. Assign each observation to the cluster with the nearest centroid
3. Assign a new centroid based on the within-cluster mean for assigned observations
4. Repeat steps 2 and 3 until the cluster assignments no longer change

We arbitrarily choose the number of clusters k, and R randomly selects starting centroid values for step 1.

The k-means algorithm



Implementing k-means for the IPV data

```
ipv_kmeans <- ipv %>% select(sec_school, no_media) %>% mutate(sec_school = scale(sec_school),
    no_media = scale(no_media)) %>% filter(!(is.na(sec_school)), !(is.na(no_media))) %>%
    kmeans(centers = 3, nstart = 10)

ipv_kmeans
```

```
## K-means clustering with 3 clusters of sizes 72, 17, 46
##
## Cluster means:
##   sec_school   no_media
## 1 -0.6135490  0.7910351
## 2  1.8803248 -1.1669709
## 3  0.2071354 -0.7678187
##
## Clustering vector:
##   [1] 3 2 2 2 2 1 1 1 1 1 1 3 3 1 1 1 1 3 1 1 1 1 1 3 3 1 1 1 1 3 3 3 3 3
##  [36] 3 3 2 1 1 1 1 2 1 3 3 1 1 2 2 1 3 3 3 3 1 3 3 3 2 2 3 3 2 1 3 1 1 1 1
##  [71] 1 1 1 3 1 1 1 2 3 1 1 3 3 3 1 1 3 1 1 1 1 1 1 3 3 3 2 3 3 3 3 3 3 1 1
## [106] 1 3 3 3 3 3 1 1 2 1 1 1 1 1 1 1 2 1 1 1 1 3 2 2 1 1 1 1 1 1
##
## Within cluster sum of squares by cluster:
## [1] 52.858371  8.657364 24.241269
##  (between_SS / total_SS =  68.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Working with the k-means object

```
str(ipv_kmeans)
```

```
## List of 9
##  $ cluster     : int [1:135] 3 2 2 2 2 1 1 1 1 1 ...
##  $ centers     : num [1:3, 1:2] -0.614 1.88 0.207 0.791 -1.167 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:3] "1" "2" "3"
##   .. ..$ : chr [1:2] "sec_school" "no_media"
##  $ totss       : num 270
##  $ withinss    : num [1:3] 52.86 8.66 24.24
##  $ tot.withinss: num 85.8
##  $ betweenss   : num 184
##  $ size        : int [1:3] 72 17 46
##  $ iter        : int 3
##  $ ifault      : int 0
##  - attr(*, "class")= chr "kmeans"
```

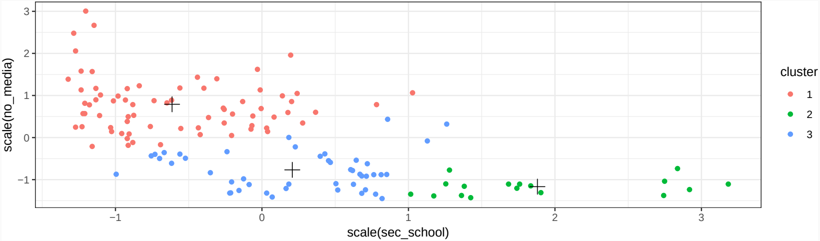
Pull out what we need from the list

```
ipv_clusters <- ipv %>% filter(!(is.na(sec_school)), !(is.na(no_
    mutate(cluster = factor(ipv_kmeans$cluster))

library(broom)
centers <- tidy(ipv_kmeans)
centers
```

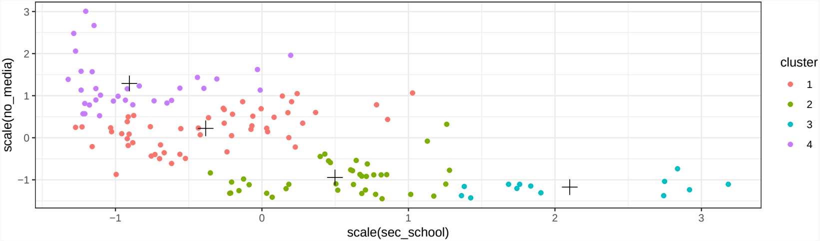
```
## # A tibble: 3 x 5
##        x1     x2  size withinss cluster
##     <dbl>  <dbl> <int>    <dbl> <fct>
## 1 -0.614  0.791    72    52.9  1
## 2  1.88  -1.17     17     8.66 2
## 3  0.207 -0.768    46    24.2  3
```

Plot it!

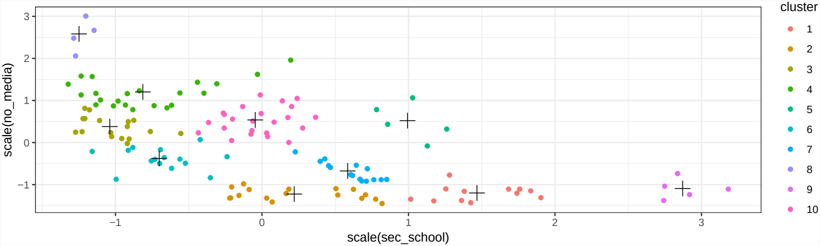
```r
ggplot(ipv_clusters, aes(x = scale(sec_school), y = scale(no_media), color = cluster)) +
    geom_point() + geom_point(data = centers, aes(x = x1, y = x2), color = "black",
    size = 4, shape = 3)
```



What if we thought there were 4 clusters?



What if we thought there were 10 clusters?



Summary

- Measurement and design matter!
- Always check your data, and think about how unit and item non-response may inform your conclusions
- Think about desirability and other forms of response bias as you interpret your results
- Design visuals and exploratory analyses to check hypotheses about what's going on in the data
- Think about the structure of your data, use descriptive statistics like correlations to describe relationships
- Think about latent structures in your data to capture clustering

Homework

- Complete exercise 3.9.2, use data(vignettes)
- You can use geom_vline() to add vertical lines
- See `https://jrnold.github.io/qss-tidy/measurement.html#quantile-quantile-plot` for an example of making a quantile-quantile plot with ggplot()
- You can do this! Use Slack to ask questions when you are stuck!