Introduction to Statistics and Data Analysis I: Cross-Tabs, Chi-Squared, Measures of Association and Correlations

Dr. Niccole M. Pamphilis

Introduction

This week we started looking a bi-variate relationship, or how two variables move in relation to one another. We looked at the relationship between discrete variables including nominal and ordinal level data.

In order to start with examining the relationship between two discrete variables we looked at the cross-tab to see what the pair-wised observations looked like.

To run a cross-tab in R you use the table command which we have seen before when looking at descriptive statistics of nominal and ordinal level variables. Let's review by starting with a table for the variables: fraud and alter_election, from our updated Scotland data set

```
data9<-readRDS(file = "Update_Scot_data.rds")
```

```
table(data9$fraud)
```

```
##
## Strongly Disagree          Disagree Slightly Disagree           Neither
##                10                16                13                27
##    Slightly Agree             Agree    Strongly Agree
##                20                 8                 6
```

```
table(data9$alter_election)
```

```
##
## Strongly Disagree          Disagree Slightly Disagree           Neither
##                10                13                12                32
##    Slightly Agree             Agree    Strongly Agree
##                17                 7                 9
```

We can see how observations are spread out across the seven categories for each variable here. However, you might notice that all these categories are a bit much to process and a 7x7 cross-tab would produce 49 paired outcomes to visually assess, possible, but not fun.

So, we are going to collapse some of our categories to create more user-friendly variables. We will group all the disagree options into one category and all the agree options in to one category.

```
##Start with Fraud variable

data9$fraud3[data9$fraud=="Strongly Disagree"]<-"Disagree"
data9$fraud3[data9$fraud=="Disagree"]<-"Disagree"
data9$fraud3[data9$fraud=="Slightly Disagree"]<-"Disagree"
data9$fraud3[data9$fraud=="Neither"]<-"Neither"
data9$fraud3[data9$fraud=="Strongly Agree"]<-"Agree"
data9$fraud3[data9$fraud=="Agree"]<-"Agree"
data9$fraud3[data9$fraud=="Slightly Agree"]<-"Agree"
```

```
#The new variable is not ordinal though, so we need to tell R to treat it as ordinal and what#he order

data9$fraud3<-ordered(data9$fraud3,
                      levels=c("Disagree", "Neither", "Agree"),
                      labels=c("Disagree", "Neither", "Agree"))

#Now we do the same for the alter_election variable

#As with before I am creating a new variable instead of coding over my old variable,
#this way if I ever want to use the full 7 categories I have not lost them

data9$alter_election3[data9$alter_election=="Strongly Disagree"]<-"Disagree"
data9$alter_election3[data9$alter_election=="Disagree"]<-"Disagree"
data9$alter_election3[data9$alter_election=="Slightly Disagree"]<-"Disagree"
data9$alter_election3[data9$alter_election=="Neither"]<-"Neither"
data9$alter_election3[data9$alter_election=="Strongly Agree"]<-"Agree"
data9$alter_election3[data9$alter_election=="Agree"]<-"Agree"
data9$alter_election3[data9$alter_election=="Slightly Agree"]<-"Agree"

#And again, we need to tell R to treat it as ordered

data9$alter_election3<-ordered(data9$alter_election3,
                    levels=c("Disagree", "Neither", "Agree"),
                    labels=c("Disagree", "Neither", "Agree"))

#We can check that our coding was succcessful
table(data9$alter_election, data9$alter_election3)
```

```
##
##                     Disagree Neither Agree
##   Strongly Disagree       10       0     0
##   Disagree                13       0     0
##   Slightly Disagree       12       0     0
##   Neither                  0      32     0
##   Slightly Agree           0       0    17
##   Agree                    0       0     7
##   Strongly Agree           0       0     9
```

To create the cross-tab we use the table() function again. This time we include both variables in the function separated by a ,

```
table(data9$fraud3, data9$alter_election3)
```

```
##
##            Disagree Neither Agree
##   Disagree       30       6     3
##   Neither         4      20     3
##   Agree           1       6    27
```

We can compare this table to the one that the original variables would have produced

```
table(data9$fraud, data9$alter_election)
```

```
##
##                     Strongly Disagree Disagree Slightly Disagree Neither
##   Strongly Disagree                 6        1                 1       1
```

```
##   Disagree                          1       10                2       2
##   Slightly Disagree                 2        1                6       3
##   Neither                           1        1                2      20
##   Slightly Agree                    0        0                1       5
##   Agree                             0        0                0       1
##   Strongly Agree                    0        0                0       0
##
##                     Slightly Agree Agree Strongly Agree
##   Strongly Disagree              0     0              1
##   Disagree                       1     0              0
##   Slightly Disagree              1     0              0
##   Neither                        2     1              0
##   Slightly Agree                11     2              1
##   Agree                          2     3              2
##   Strongly Agree                 0     1              5
```

The table above produces a slightly more nuanced table, but one that is not as easier to process visually.

Chi-Squared Test

Looking at the table for the relationship between the two variables it does appear at a lot of the observations fall along the diagonal and that there might be a relationship between the two variables. But, as always, it is better to test instead of guess.

To run a chi-squared test you first need to create a table for your data. R will use the table to then compare the observed pair-wise frequencies to the expected and give you a result.

```
#In situations where you need to create something before anlaysing it, it is better to
#save the information as an object instead of layer all the code together in one string.

t1 <- table(data9$fraud3, data9$alter_election3)
```

```
chisq.test(t1)
```

```
##
##  Pearson's Chi-squared test
##
## data:  t1
## X-squared = 86.884, df = 4, p-value < 2.2e-16
```

However, if you want you can run all the code at once.

```
chisq.test(table(data9$fraud3, data9$alter_election3))
```

```
##
##  Pearson's Chi-squared test
##
## data:  table(data9$fraud3, data9$alter_election3)
## X-squared = 86.884, df = 4, p-value < 2.2e-16
```

Either approach will get you the same answer, it just depends on which is more comfortable and cleaner for you to write.

Looking at the results we can see the chi-squared (X-Squared in the output) and the associated p-value. For the above example, we would reject the null and assume that there was some level of association between the two variables.

Proportions versus Frequencies

When you make a table in R the default is to present you with the observed frequencies in each cell. If you would like to see the proportions this just takes an extra step in your code.

```
#First you make your table, then you use the prop.table() command.

t1p <- prop.table(t1, 2)*100
```

```
t1p
```

```
##
##             Disagree   Neither     Agree
##   Disagree 85.714286 18.750000  9.090909
##   Neither  11.428571 62.500000  9.090909
##   Agree     2.857143 18.750000 81.818182
```

Measures of Association

The chi-squared test allows us to determine IF there is a relationship between two variables, but not how strong that relationship (i.e., are we better able to predict outcomes by a little or a lot).

To determine how strong the relationship is we need to use an appropriate measure of association. These measures can be found in different packages for R. For the ones below you will need to install the psych package and the desctools package before you run the code.

Phi-coefficient

The first measure we introduced in class looked at the relationship between two nominal level variables with only two categories each.

Our dataset did not have two binary variables so I will create one for age to pair with our ideol_dum variable.

```
  library(psych) #Package that has phi function in it

  data9$age_dum<-ifelse(data9$age<65, c("<65"), c("+65"))

  td<-table(data9$ideol_dum,data9$age_dum )

  td
```

```
##
##         +65 <65
##   left   23  31
##   right  13   7
```

```
  phi(td)
```

```
## [1] -0.2
```

To calculate the phi coefficient we first needed to create the table the same way we had with our cross-tab. Then we can execute the command and it produces the value for us. Here we have a value of -0.2, which would suggest a weak relationship.

Typically, we would run the chi-squared test first to determine if there was even a relationship worth measuring. So let's do that now:

```r
chisq.test(td)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  td
## X-squared = 2.1049, df = 1, p-value = 0.1468
```

Turns out that are variables do not have relationship, so we shouldn't be surprised by the phi-coefficient results.

Cramer's V

We talked in class about measures for nominal variables with more than 2 categories and the Cramer's V is such a test. In fact, the Phi-coefficient is just a specialized case of Cramer's V. We can see that by calculating Cramer's V for our binary data from the phi-coefficient above:

```
##
## Attaching package: 'DescTools'
```

```
## The following objects are masked from 'package:psych':
##
##     AUC, ICC, SD
```

```
## [1] 0.199095
```

Cramer's V can also be used for variables with more than two categories. Here I will run it for region in Scotland against partisanship.

```
##
##                       Conservative Labour Lib Dem SNP Green None Brexit
##   North East Scotland            1      3       2   2     0    4      0
##   Highlands & Islands            1      1       1   3     0    6      0
##   South Scotland                10      1       2   1     1    2      1
##   West Scotland                  1      3       0   4     0    1      0
##   Central Scotland               2      1       1   4     0    2      0
##   Mid-Scotland & Fife            2      2       3   2     1    4      1
##   Lotians                        0      1       1   1     1    3      0
##   Glasgow                        0      3       1   1     1    3      0
##
##                       Don't know
##   North East Scotland          0
##   Highlands & Islands          2
##   South Scotland               1
##   West Scotland                1
##   Central Scotland             1
##   Mid-Scotland & Fife          1
##   Lotians                      1
##   Glasgow                      0
```

```
## [1] 0.2734085
```

```
## Warning in chisq.test(t2): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
```

```
## data:  t2
## X-squared = 51.803, df = 49, p-value = 0.365
```

Lambda Coefficient

To run the lambda test we can use the DescTools package which has a lambda function in it. As with the other measures of association, you will need to produce the table first. Additionally, you will need to tell R if you are using the columns to predict or the rows to predict.

```
  library(DescTools)

  table(data9$fraud3)
```

```
##
## Disagree  Neither    Agree
##       39       27       34
```

```
  t1
```

```
##
##            Disagree Neither Agree
##   Disagree       30       6     3
##   Neither         4      20     3
##   Agree           1       6    27
```

```
  Lambda(t1,
       direction="column")
```

```
## [1] 0.6461538
```

Gamma Coefficient

The final test we spoke about in class was for two ordinal level variables. The gamma test tells us strength and direction of the relationship. Remember negative values correspond to a negative relationship and positive values correspond to a positive relationship. As will all our measures it is bounded, here between -1 and 1.

For the gamma coefficient you just need to tell R what two variables you want to use, you do not need to generate a table first.

```
 library(DescTools)

  GoodmanKruskalGamma(data9$fraud3, data9$alter_election3)
```

```
## [1] 0.8783383
```

Scatter Plots and Correlations

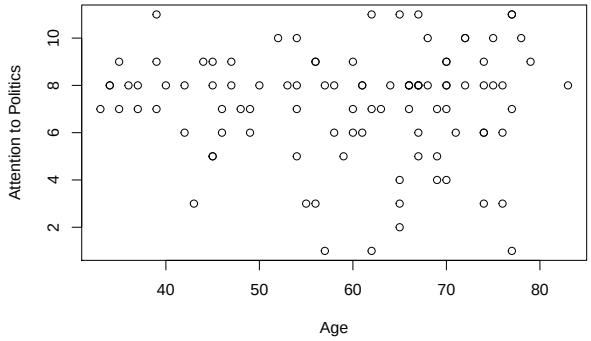
~Scatter Plots

A good way to start to understand what your data look like are to plot them in a graph. To examine the relationship between two continuous level variables we can use a scatter plot.

In week 2 we looked at how to create graphs, including scatter plots, which we will review here now.

First, we looked at how to create graphs using the basic graphing commands in R.

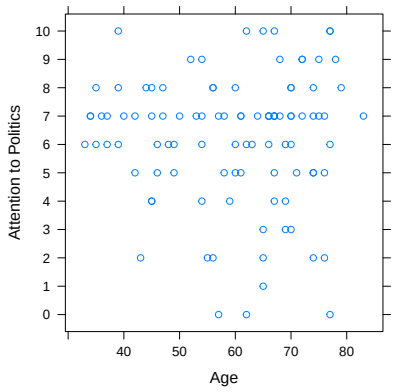
```
plot(data9$age, data9$polattention,
     xlab="Age",
     ylab="Attention to Politics")
```



We can also use the lattice package to create our scatter plot.

```
library(lattice)

xyplot(data9$polattention~data9$age,
       xlab="Age",
       ylab="Attention to Politics",
       asp=1)
```



In the graph above, it does not look like there is a strong relationship between attention to politics and a respondent's age. However, we will want to check with the actual statistics and tests, instead of just trying our visual assessment.

Covariance and Correlation

A quick way to summarize the relationship between to continuous level variables is by using a correlation statistic. The correlation value, recall, is a standardized measure which ranges between -1 and 1, which helps use have a standard way way of interpreting the results.

The correlation value is based on the co-variance, which is then standardized using the standard deviations for the two variables.

If you wanted, you can produce the co-variance in R.

```
cov(data9$age, as.numeric(data9$polattention))
```

```
## [1] -0.6505051
```

Notice, in the code above I had to add in the option "as.numeric" in front of the variable political attention. I had to do this because I have told R that political attention was an ordinal level variable, however, correlations should only be calculated between continuous level variables (so, why have I done this... once a variable has 7 or more categories is starts to behave like a continuous level variable).

Is the co-variance value produced by R large or small? It is hard to tell based on how the variables were measure, so we should turn to the correlation.

To calculate the correlation you do not need to start with the co-variance, you can just tell R to produce the correlation value.

```
cor(data9$age, as.numeric(data9$polattention))
```

```
## [1] -0.02157662
```

Looking at the correlation value, what is the direction of the relationship between the two variables? Is it weak or is it strong?

Chi-Squared Test for Correlation Values

Just like with our cross-tabs we can test for statistical significance of our correlation value. Sometimes, we may have what appears to be a weak relationship between to variables, but it is still one that is statistically meaningful, while other times we might have what appears to be a strong relationship that fails to achieve statistical significance.

We can run a chi-squared test on our correlation value, with the null set that the correlation in the population is actually equal to 0.

Let's test our correlation from before.

```
cor.test(data9$age, as.numeric(data9$polattention))
```

```
##
##  Pearson's product-moment correlation
##
## data:  data9$age and as.numeric(data9$polattention)
## t = -0.21365, df = 98, p-value = 0.8313
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.2170748  0.1755856
## sample estimates:
##         cor
## -0.02157662
```

You can see the output for the test is set-up in a similar fashion to our tests from previous weeks. We see correlation value at the bottom, the alternative hypothesis listed, and our p-value.

Based on the output from the test, what can we say about this correlation value?

If you want, you can run your correlation test, assuming a directional hypothesis too. And similar to our other tests you use the option alternative and specify either less or greater

```r
cor.test(data9$age, as.numeric(data9$polattention),
         alternative="less")
```

```
##
##  Pearson's product-moment correlation
##
## data:  data9$age and as.numeric(data9$polattention)
## t = -0.21365, df = 98, p-value = 0.4156
## alternative hypothesis: true correlation is less than 0
## 95 percent confidence interval:
##  -1.0000000  0.1444129
## sample estimates:
##         cor
## -0.02157662
```

Correlation Matrix

Sometimes we are interested in the correlations between a lot of the variables in our dataset. Particularly in the early stages of analysis. In these situations we COULD run a lot of individual correlation statistics, or we could just run all our correlations at once a put the values in a grid to examine.

To do this we are going to use the Hmisc package (which will need to be installed, if you have not used it before). Following the rcorr command we combine a list of all the variables we are interested in seeing the correlations between.

```
## Loading required package: survival
```

```
## Loading required package: Formula
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'ggplot2'
```

```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
##
## Attaching package: 'Hmisc'
```

```
## The following objects are masked from 'package:DescTools':
##
##     %nin%, Label, Mean, Quantile
```

```
## The following object is masked from 'package:psych':
##
##     describe
```

```
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
##       [,1]  [,2]  [,3]  [,4]  [,5]
## [1,]  1.00 -0.02  0.13  0.18 -0.16
## [2,] -0.02  1.00 -0.11 -0.07 -0.11
## [3,]  0.13 -0.11  1.00  0.76 -0.51
## [4,]  0.18 -0.07  0.76  1.00 -0.62
```

```
## [5,] -0.16 -0.11 -0.51 -0.62  1.00
##
## n= 100
##
##
## P
##      [,1]   [,2]   [,3]   [,4]   [,5]
## [1,]        0.8313 0.1971 0.0708 0.1055
## [2,] 0.8313        0.2814 0.5014 0.2576
## [3,] 0.1971 0.2814        0.0000 0.0000
## [4,] 0.0708 0.5014 0.0000        0.0000
## [5,] 0.1055 0.2576 0.0000 0.0000
```

The output above shows three sets of information. The first grid represents the correlations between the variables, which are numbered in order of entry in the code. Notice that the diagonal is 1.00, because a variable will be perfectly correlated with itself.

The second piece of information is the number of observations in each pairing (n=100).

The third piece of information are the p-values associated with each of the correlation values (it includes the statistical test for you). From our example earlier, can you find the correlation and p-value in the tables above? Are they the same?

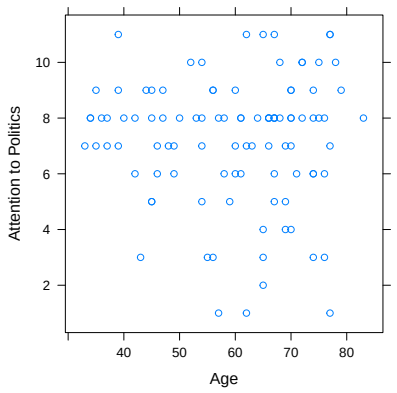
Advanced Scatter Plots

When we started the R lab today we looked at the basic scatter plot for the relationship between age and attention to politics. Looking at just the plotted points, it was difficult to tell if there as anything really going on to capture and measure.

To help interpret scatter plots it can be useful to add a few additional features. The first additional point you might consider adding a best fit linear line (this will be the regression line you learn about in Statistics and Data Analysis II or Regression I).

We can add this to our graph with an additional bit of code.

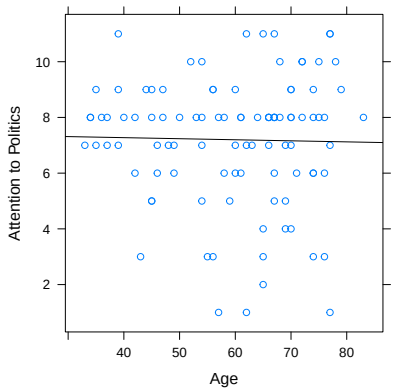
```
#Standard Scatter Plot
xyplot(as.numeric(data9$polattention)~data9$age,
     xlab="Age",
     ylab="Attention to Politics" ,
     asp=1
)
```



To add a straight line to the plot, to see how well it can capture the pattern of the relationship we use the panel option. We set type to include p (which represents the plotted points), and r (which represents the best fit linear line).

The col.line command is used to set the line colour to black so it stands out against the blue points.

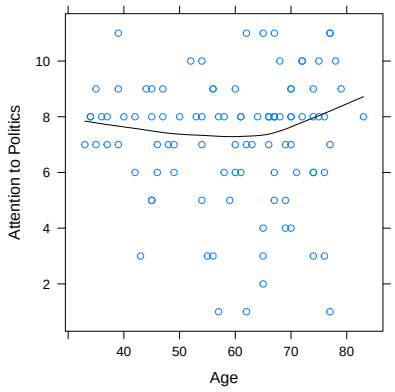
```
#linear fit line
xyplot(as.numeric(data9$polattention)~data9$age,
     xlab="Age",
     ylab="Attention to Politics" ,
     asp=1,
     panel = function(x, y) {
       panel.xyplot(x, y,
                    type = c("p", "r"),
                    col.line = "black")
     })
```



While the linear line can help with understanding it points follow a straight line or not, it can also be useful to include a Loess curve. You can think of this as breaking the data into smaller sections and drawing a line to capture what the pattern looks like and connecting all those lines together. If the pattern is truly linear, the Loess curve will look very similar to the best fit line.

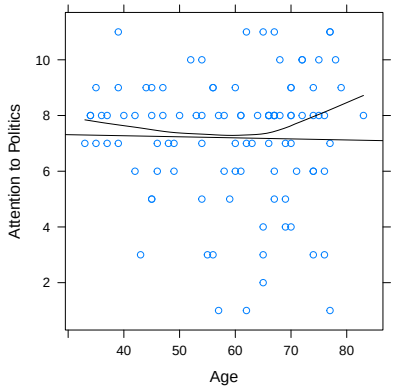
To add the Loess curve, we use the same set-up as above, but now we use the smooth option instead of r.

```
xyplot(as.numeric(data9$polattention)~data9$age,
       xlab="Age",
       ylab="Attention to Politics" ,
       asp=1,
       panel = function(x, y) {
         panel.xyplot(x, y,
                      type = c("p", "smooth"),
                      col.line = "black")
       })
```



You can also combine the scatter plot, the best fit line, and the Loess curve in to one graph so you have the plotted points, you have what the best fit linear line looks like, and you have what pattern the points actually follow.

```
xyplot(as.numeric(data9$polattention)~data9$age,
       xlab="Age",
       ylab="Attention to Politics" ,
       asp=1,
       panel = function(x, y) {
         panel.xyplot(x, y,
                      type = c("p", "smooth", "r"),
                      col.line = "black")
       })
```




```
saveRDS(data9, file="Ver_9_Scotland_data.rds")
```