Chapter 1

Contingency Tables

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(mosaic)
library(mosaicData)  # where the Mites data lives
```

We are often interested in experiments where the response variable is categorical in nature. For example, perhaps we treat a bunch of plots with two types of insecticides and after 2 weeks observe if the plots are infested or not infested with some insect. Our goal would be to decide if the proportion of plots invested is different amongst the two treatment types.

We will have two questions:

1. What statistic could be calculated from the observed data to measure how far the observed data is from the null hypothesis?
2. Given the statistic in part 1, how should it vary from sample to sample assuming the null hypothesis (no difference in treatments) is true?

1.1 2 x 2 Contingency Tables

Example:

We will examine these questions in the context of a study where researchers suspected that attack of a plant by one organism induced resistance to subsequent attack by a different organism. Individually potted cotton plants were randomly allocated to two groups: infestation by spider mites or no infestation. After two weeks the mites were dutifully removed by a conscientious research assistant, and both groups were inoculated with Verticillium, a fungus that causes Wilt disease.

```
data(Mites)
str(Mites)

## 'data.frame': 47 obs. of  2 variables:
##  $ treatment: Factor w/ 2 levels "mites","no mites": 1 1 1 1 1 1 1 1 1 1 ...
##  $ outcome  : Factor w/ 2 levels "no wilt","wilt": 2 2 2 2 2 2 2 2 2 2 ...
```

We will summarize the data into a *contingency table* that counts the number of plants in each treatment/wilt category[1].

```
# Using mosaic's tally function
mosaic::tally(outcome ~ treatment, data=Mites, # table of outcome by treatment
              format='count')                  # give the raw counts, not percentages

##          treatment
## outcome   mites no mites
##   no wilt    15        4
##   wilt       11       17

# Using dplyr and tidyr
Mites %>% group_by(outcome, treatment) %>%
  summarise(count = n()) %>%
  spread(treatment, count)

## Source: local data frame [2 x 3]
## Groups: outcome [2]
##
##   outcome mites no mites
##    (fctr) (int)    (int)
## 1 no wilt    15        4
## 2    wilt    11       17

# A dataframe summarizing function in base R.
# The select command is to switch the order of the
# columns so that the result matches the two above
table(Mites %>% select(outcome, treatment))

##          treatment
## outcome   mites no mites
##   no wilt    15        4
##   wilt       11       17
```

From this table we can see that of the $n = 47$ plants, 28 of them wilted. Furthermore we see that the mites were applied to $n_m = 26$ of the plants. Is this data indicative of mites inferring a disease resistance? More formally we are interested in testing

$$\begin{aligned} H_0: &\quad \pi_w = \pi_{w|m} \\ H_0: &\quad \pi_w \neq \pi_{w|m} \end{aligned}$$

where the relevant parameters are π_w, the probability that a plant will wilt, and $\pi_{w|m}$, the probability that a plant will wilt given that it has been treated with mites.

To make our formulas that we are about to derive, we'll make our subscripts be denoted by row and column index values i, j and recall our convention that the first index will refer to the row and the second to the column. So we have observed $O_{1,1} = 15$ plants were in the Mites treatment group and did not wilt, $O_{1,2} = 4$ plants were in the No Mites treatment group and did not wilt. Similarly we define:

- $n_{1,\cdot} = 19$ as the number of plants that did not wilt

[1] If the code below doesn't give the counts, it might be because the packages `mosaic` and `dplyr` are fighting over which gets to define the function `tally()`. You can force your code to use the mosaic version of the function by using `mosaic::tally( outcome ~ treatment, data=mites )` where the key part is to give the package name and then the function. Within `mosaic`'s `tally()` function there is an option `format=` option that allows you to specify if you want the raw `counts`, the `proportion` in each column, or as `percent` in each column.

- $n_{2,\cdot} = 28$ as the number of plants that did wilt
- $n_{\cdot,1} = 26$ as the number of plants that received the Mite treatment
- $n_{\cdot,2} = 21$ as the number of plants that did not receive a Mite treatment.
- $p_{1,\cdot} = \frac{19}{47}$ be the proportion of plants that did not wilt
- $p_{2,\cdot} = \frac{28}{47}$ be the proportion all the plants that wilted,
- $p_{\cdot,1} = \frac{26}{47}$ be the proportion of plants receiving the Mite treatment
- $p_{\cdot,2} = \frac{21}{47}$ be the number of plants receiving the Mite treatment.

		Treatment: Mites	Treatment: No Mites	
Outcome	No Wilt	$O_{1,1} = 15$	$O_{1,2} = 4$	$n_{1,\cdot} = 19$ $p_{1,\cdot} = \frac{19}{47}$
	Wilt	$O_{2,1} = 11$	$O_{2,2} = 17$	$n_{2,\cdot} = 28$ $p_{2,\cdot} = \frac{28}{47}$
		$n_{\cdot,1} = 26$ $p_{\cdot,1} = \frac{26}{47}$	$n_{\cdot,2} = 21$ $p_{\cdot,2} = \frac{21}{47}$	$n = 47$

What would you expect to see if there was absolutely no effect of the mite treatment? The wilting plants should be equally dispersed between the mite and non-mite treatments, but we also have to account for the fact that we have more mite treatments. If Mite treatment and wilting are independent then $P(Wilt \mid Mite) = P(Wilt)$ by our definition of independence and for each cell in the Expected number of plants in the Mite/Wilting cell should

$$
\begin{aligned}
E_{2,1} &= P(Mite \cap Wilt) * n \\
&= P(Mite)\, P(Wilt) * n \qquad \text{This is justified if independent! aka if } H_0 \text{ is true} \\
&= (p_{2,\cdot})\,(p_{\cdot,1})\, n \\
&= \frac{28}{47} \cdot \frac{26}{47} \cdot 47 \\
&= \frac{(n_{2,\cdot})\,(n_{\cdot,1})}{n} \\
&= 15.49
\end{aligned}
$$

A similar calculation for the remain cells gives us the following:

		Treatment: Mites	Treatment: No Mites	
Outcome	No Wilt	$O_{1,1} = 15$ $E_{1,1} = 10.51$	$O_{1,2} = 4$ $E_{1,2} = 8.49$	$n_{1,\cdot} = 19$ $p_{1,\cdot} = \frac{19}{47}$
	Wilt	$O_{2,1} = 11$ $E_{2,1} = 15.49$	$O_{2,2} = 17$ $E_{2,2} = 12.51$	$n_{2,\cdot} = 28$ $p_{2,\cdot} = \frac{28}{47}$
		$n_{\cdot,1} = 26$ $p_{\cdot,1} = \frac{26}{47}$	$n_{\cdot,2} = 21$ $p_{\cdot,2} = \frac{21}{47}$	$n = 47$

This is the first case where our test statistic will not be just plugging in the sample statistic into the null hypothesis. Instead we will consider a test statistic that is more flexible and will handle more general cases (say 3 or more response or treatment groups) Our statistic for assessing how far our observed data is from what we expect under the null hypothesis involves the difference between the observed and the expected for each of the cells, but again we don't want to just sum the differences, instead will make the differences positive by squaring the differences. Second, a difference of 10 between the observed and expected cell count is very different if the number expected is 1000 than if it is 10, so we will scale the observed difference by dividing by the expected cell count.

We define

$$\begin{aligned} X^2 &= \sum_{\text{all ij cells}} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \\ &= \frac{(15 - 10.51)^2}{10.51} + \frac{(4 - 8.49)^2}{8.49} + \frac{(11 - 15.49)^2}{15.49} + \frac{(17 - 12.51)^2}{12.51} \\ &= 1.92 + 2.37 + 1.30 + 1.61 \\ &= 7.20 \end{aligned}$$

If the null hypothesis is true, then this statistic should be small, and a large value of the statistic is indicative of the null hypothesis being incorrect. But how large must the statistic be before we reject the null hypothesis? Again, we'll randomly shuffle the treatment assignments and recalculate the statistic many times and examine the sampling distribution of X^2.

To do this efficiently, we'll need a way of easily calculating this test statistic. In a traditional course I would introduce this test by the name of "Pearson's Chi-squared test" and we can obtain the test statistic using the following code:

```
# function is chisq.test() and we need to tell it not to do the Yates continuity
# correction and just calculate the test statistic as we've described
chisq.test( table(Mites), correct=FALSE )   # do a Chi-sq test

##
##  Pearson's Chi-squared test
##
## data:  table(Mites)
## X-squared = 7.2037, df = 1, p-value = 0.007275
```

R is performing the traditional Pearson's Chi-Squared test which assumes our sample sizes are large enough for several approximations to be good. Fortunately, we don't care about this approximation to the p-value and will use simulation methods which will be more accurate. In order to

use the `chisq.test()` function to do our calculations, we need to extract the test-statistic from the output of the `function`.

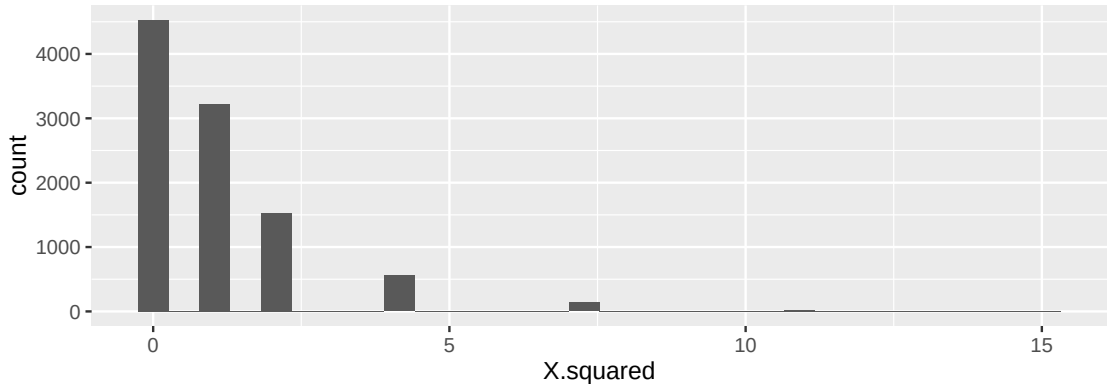
```
# extract the X^2 test statistic from the output
X.sq <- chisq.test( table(Mites), correct=FALSE )$statistic # grab only the test statistic
X.sq

## X-squared
##  7.203748
```

Next we wish to repeat our shuffling trick of the treatment labels to calculate the sampling distribution of X^{2*}, which is the distribution of X^2 when the null hypothesis of no difference between treatments is true.

```
Mites.star <- Mites %>% mutate(treatment = shuffle(treatment))
table(Mites.star)

##           outcome
## treatment  no wilt wilt
##   mites         12   14
##   no mites       7   14

chisq.test( table(Mites.star), correct=FALSE )$statistic # grab only the test statistic

## X-squared
## 0.7928475
```

We see that this code is creating a data frame with a single column called `X.squared` and next we simulate a large number of times and display the sampling distribution of X^{2*}.

```
SamplingDist <- do(10000)*{
  Mites.star <- Mites %>% mutate(treatment = shuffle(treatment))
  chisq.test( table(Mites.star), correct=FALSE )$statistic
}
ggplot( SamplingDist, aes(x=X.squared)) + geom_histogram()

## `stat_bin()` using `bins = 30`.  Pick better value with `binwidth`.
```



At first glance this seems wrong because it is not a nice looking distribution. However there are only a small number of ways to allocate the treatments labels to the two possible outcomes. Second, for the test statistic we have chosen only the right hand side of the distribution (large values of X^*) would be evidence against the null hypothesis, so we only look at $X^{2*} > 7.20$.

```
p.value <- SamplingDist %>% summarize( p.value = mean( X.squared >= X.sq ) )
p.value

##   p.value
## 1  0.0162
```

We see that the p-value is 0.0162 and conclude that there is strong evidence to reject the null hypothesis that the mite treatment does not affect the probability of wilting. That is to say, the probability of observing data as extreme as ours is unlikely to occur by random chance when the null hypothesis is true.

As usual, it is pretty annoying to have to program the permutation test ourselves. Fortunately the `chisq.test()` function allows us to option to tell it to do a permutation based test. There is an option `simulate.p.value` which reproduces the simulation test we just performed.

```
chisq.test( table(Mites), simulate.p.value=TRUE, B=10000 )

##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  table(Mites)
## X-squared = 7.2037, df = NA, p-value = 0.0161
```

Before we had our excellent computers, we would have to compare the observed X^2 test statistic to some distribution to determine if it is large enough to be evidence against the null. It can be shown[2] that if the null hypothesis is correct then $X^2 \dot{\sim} \chi^2_1$ where this is the *Chi-squared* distribution with 1 degree of freedom. This is the distribution that the `chisq.test()` compares against if we don't tell it to do a permutation based test. Furthermore, even if the null hypothesis is true the test statistic is only *approximately* normal and that approximation gets better and better as the total sample size increases. The asymptotic approximation is usually acceptable if the observed count in each cell is greater than 5. Even then, a slightly better approximation can be obtained by using the Yates' continuity correction. Typically I will perform the analysis both ways and confirm we get the same inference. If the two methods disagree, I'd trust the permutation method.

Example:

In a study to investigate possible treatments for human infertility, researchers[3] performed a double-blind study and randomly divided 58 patients into two groups. The treatment group ($n_t = 30$) received 100 mg per day of Doxycycline and the placebo group ($n_p = 28$) received a placebo but were unaware that it was a placebo. Within 5 months, the treatment group had 5 pregnancies, while the placebo group had 4.

[2]The table can be thought of as coming from a multinomial distribution and the central limit theorem can be applied to the sum of the cells.

[3]Harrison, R. F., Blades, M., De Louvois, J., & Hurley, R. (1975). Doxycycline treatment and human infertility. *The Lancet*, 305(7907), 605-607.

		Treatment: Doxycycline	Treatment: Placebo	
Outcome	Conceived	$O_{1,1} = 5$ $E_{1,1} = \left(\frac{9 \cdot 30}{58}\right) = 4.66$	$O_{1,2} = 4$ $E_{1,2} = \left(\frac{9 \cdot 28}{58}\right) = 4.34$	$n_{1,\cdot} = 9$
	Not Conceived	$O_{2,1} = 25$ $E_{2,1} = \left(\frac{49 \cdot 30}{58}\right) = 25.34$	$O_{2,2} = 24$ $E_{2,2} = \left(\frac{49 \cdot 28}{58}\right) = 23.66$	$n_{2,\cdot} = 49$
		$n_{\cdot,1} = 30$	$n_{\cdot,2} = 28$	$n = 58$

Just looking at the observed vs expected there doesn't seem to be much difference between the treatments. In fact, due to the discrete nature of the data (i.e. integer values) we can't imagine data that *any* closer to the expected value that what we observed. The p-value here ought to be 1! To confirm this we do a similar test as before.

```
Conceived <-  data.frame(
  Treatment=c(rep('Doxycyline',30), rep('Placebo',28)),
  Outcome=c(rep('Conceived',5), rep('Not Conceived',25),
            rep('Conceived',4), rep('Not Conceived',24)))

chisq.test( table(Conceived), simulate.p.value=TRUE, B=10000 )

##
##  Pearson's Chi-squared test with simulated p-value (based on 10000
##  replicates)
##
## data:  table(Conceived)
## X-squared = 0.062628, df = NA, p-value = 1
```