17.6: Bayesian Analysis of Contingency Tables

Time to change gears. Up to this point I've been talking about what Bayesian inference is and why you might consider using it. I now want to briefly describe how to do Bayesian versions of various statistical tests. The discussions in the next few sections are not as detailed as I'd like, but I hope they're enough to help you get started. So let's begin.

The first kind of statistical inference problem I discussed in this book appeared in Chapter 12, in which we discussed categorical data analysis problems. In that chapter I talked about several different statistical problems that you might be interested in, but the one that appears most often in real life is the analysis of *contingency tables*. In this kind of data analysis situation, we have a cross-tabulation of one variable against another one, and the goal is to find out if there is some *association* between these variables. The data set I used to illustrate this problem is found in the `chapek9.Rdata` file, and it contains a single data frame `chapek9`

```
load("./rbook-master/data/chapek9.Rdata")
head(chapek9)
```

```
##   species choice
## 1   robot flower
## 2   human   data
## 3   human   data
## 4   human   data
## 5   robot   data
## 6   human flower
```

In this data set, we supposedly sampled 180 beings and measured two things. First, we checked whether they were humans or robots, as captured by the `species` variable. Second, we asked them to nominate whether they most preferred flowers, puppies, or data. When we produce the cross-tabulation, we get this as the results:

```
crosstab <- xtabs( ~ species + choice, chapek9 )
crosstab
```

```
##         choice
## species puppy flower data
##   robot    13     30   44
##   human    15     13   65
```

Surprisingly, the humans seemed to show a much stronger preference for data than the robots did. At the time we speculated that this might have been because the questioner was a large robot carrying a gun, and the humans might have been scared.

17.6.1 orthodox text

Just to refresh your memory, here's how we analysed these data back in Chapter@refch:chisquare. Because we want to determine if there is some *association* between `species` and `choice` , we used the `associationTest()` function in the `lsr` package to run a chi-square test of association. The results looked like this:

```
library(lsr)
```

```
## Warning: package 'lsr' was built under R version 3.5.2
```

```
associationTest( ~species + choice, chapek9 )
```

```
##
##      Chi-square test of categorical association
##
## Variables:   species, choice
##
## Hypotheses:
##    null:        variables are independent of one another
##    alternative: some contingency exists between variables
##
## Observed contingency table:
##        choice
## species puppy flower data
##   robot    13     30   44
##   human    15     13   65
##
## Expected contingency table under the null hypothesis:
##        choice
## species puppy flower data
##   robot  13.5   20.8 52.7
##   human  14.5   22.2 56.3
##
## Test results:
##    X-squared statistic:  10.722
##    degrees of freedom:  2
##    p-value:  0.005
##
## Other information:
##    estimated effect size (Cramer's v):  0.244
```

Because we found a small p value (in this case p<.01), we concluded that the data are inconsistent with the null hypothesis of no association, and we rejected it.

17.6.2 Bayesian test

How do we run an equivalent test as a Bayesian? Well, like every other bloody thing in statistics, there's a lot of different ways you *could* do it. However, for the sake of everyone's sanity, throughout this chapter I've decided to rely on one R package to do the work. Specifically, I'm going to use the `BayesFactor` package written by Jeff Rouder and Rich Morey, which as of this writing is in version 0.9.10.

For the analysis of contingency tables, the `BayesFactor` package contains a function called `contingencyTableBF()`. The data that you need to give to this function is the contingency table itself (i.e., the `crosstab` variable above), so you might be expecting to use a command like this:

```
library( BayesFactor )           # ...because we have to load the package
contingencyTableBF( crosstab )   # ...because that makes sense, right?
```

However, if you try this you'll get an error message. This is because the `contingencyTestBF()` function needs one other piece of information from you: it needs to know what *sampling plan* you used to run your experiment. You can specify the sampling plan using the `sampleType` argument. So I should probably tell you what your options are! The `contingencyTableBF()` function distinguishes between four different types of experiment:

- **Fixed sample size**. Suppose that in our `chapek9` example, our experiment was designed like this: we deliberately set out to test 180 people, but we didn't try to control the number of humans or robots, nor did we try to control the choices they made. In this design, the total number of observations N is fixed, but everything else is random. This is referred to as "joint multinomial"

sampling, and if that's what you did you should specify `sampleType = "jointMulti"`. In the case of the `chapek9` data, that's actually what I had in mind when I invented the data set.

- **Fixed row (or column) totals**. A different kind of design might work like this. We decide ahead of time that we want 180 people, but we try to be a little more systematic about it. Specifically, the *experimenter* constrains it so that we get a predetermined number of humans and robots (e.g., 90 of each). In this design, *either* the row totals or the column totals are fixed, but not both. This is referred to as "independent multinomial" sampling, and if that's what you did you should specify `sampleType = "indepMulti"`.
- **Both row and column totals fixed**. Another logical possibility is that you designed the experiment so that *both* the row totals and the column totals are fixed. This doesn't make any sense at all in the `chapek9` example, but there are other deisgns that can work this way. Suppose that I show you a collection of 20 toys, and then given them 10 stickers that say `boy` and another 10 that say `girl`. I then give them 10 `blue` stickers and 10 `pink` stickers. I then ask you to put the stickers on the 20 toys such that every toy has a colour and every toy has a gender. No matter how you assign the stickers, the total number of pink and blue toys will be 10, as will the number of boys and girls. In this design *both* the rows and columns of the contingency table are fixed. This is referred to as "hypergeometric" sampling, and if that's what you've done you should specify `sampleType = "hypergeom"`.
- **Nothing is fixed**. Finally, it might be the case that *nothing* is fixed. Not the row columns, not the column totals, and not the total sample size either. For instance, in the `chapek9` scenario, suppose what I'd done is run the study for a fixed length of *time*. By chance, it turned out that I got 180 people to turn up to study, but it could easily have been something else. This is referred to as "Poisson" sampling, and if that's what you've done you should specify `sampleType="poisson"`.

Okay, so now we have enough knowledge to actually run a test. For the `chapek9` data, I implied that we designed the study such that the total sample size N was fixed, so we should set `sampleType = "jointMulti"`. The command that we need is,

```
library( BayesFactor )
```

```
## Warning: package 'BayesFactor' was built under R version 3.5.2
```

```
## Loading required package: coda
```

```
## Warning: package 'coda' was built under R version 3.5.2
```

```
## Loading required package: Matrix
```

```
## ************
## Welcome to BayesFactor 0.9.12-4.2. If you have questions, please contact Richard M
##
## Type BFManual() to open the manual.
## ************
```

```
contingencyTableBF( crosstab, sampleType = "jointMulti" )
```

```
## Bayes factor analysis
## --------------
## [1] Non-indep. (a=1) : 15.92684 ±0%
##
## Against denominator:
##   Null, independence, a = 1
## ---
## Bayes factor type: BFcontingencyTable, joint multinomial
```

As with most R commands, the output initially looks suspiciously similar to utter gibberish. Fortunately, it's actually pretty simple once you get past the initial impression. Firstly, note that the stuff at the top and bottom are irrelevant fluff. You already know that you're doing a Bayes factor analysis. You already know that you're analysing a contingency table, and you already know that you specified a joint multinomial sampling plan. So let's strip that out and take a look at what's left over:

```
[1] Non-indep. (a=1) : 15.92684 @plusorminus0%

Against denominator:
  Null, independence, a = 1
```

Let's also ignore those two `a=1` bits, since they're technical details that you don't need to know about at this stage.[268] The rest of the output is actually pretty straightforward. At the bottom, the output defines the null hypothesis for you: in this case, the null hypothesis is that there is no relationship between `species` and `choice`. Or, to put it another way, the null hypothesis is that these two variables are *independent*. Now if you look at the line above it, you might (correctly) guess that the `Non-indep.` part refers to the *alternative* hypothesis. In this case, the alternative is that there *is* a relationship between `species` and `choice`: that is, they are not independent. So the only thing left in the output is the bit that reads

```
15.92684 @plusorminus0%
```

The 15.9 part is the Bayes factor, and it's telling you that the odds for the alternative hypothesis against the null are about 16:1. The ±0% part is not very interesting: essentially, all it's telling you is that R has calculated an exact Bayes factor, so the uncertainty about the Bayes factor is 0%.[269] In any case, the data are telling us that we have moderate evidence for the alternative hypothesis.

17.6.3 Writing up the results

When writing up the results, my experience has been that there aren't quite so many "rules" for how you "should" report Bayesian hypothesis tests. That might change in the future if Bayesian methods become standard and some task force starts writing up style guides, but in the meantime I would suggest using some common sense. For example, I would avoid writing this:

A Bayesian test of association found a significant result (BF=15.92)

To my mind, this write up is unclear. Even assuming that you've already reported the relevant descriptive statistics, there are a number of things I am unhappy with. First, the concept of "statistical significance" is pretty closely tied with p-values, so it reads slightly strangely. Second, the "BF=15.92" part will only make sense to people who already understand Bayesian methods, and not everyone does. Third, it is somewhat unclear exactly which test was run and what software was used to do so.

On the other hand, unless precision is *extremely* important, I think that this is taking things a step too far:

We ran a Bayesian test of association using version 0.9.10-1 of the BayesFactor package using default priors and a joint multinomial sampling plan. The resulting Bayes factor of 15.92 to 1 in favour of the alternative hypothesis indicates that there is moderately strong evidence for the non-independence of species and choice.

Everything about that passage is correct, of course. Morey and Rouder (2015) built their Bayesian tests of association using the paper by Gunel and Dickey (1974), the specific test we used assumes that the experiment relied on a joint multinomial sampling plan, and indeed the Bayes factor of 15.92 is moderately strong evidence. It's just far too wordy.

In most situations you just don't need that much information. My preference is usually to go for something a little briefer. First, if you're reporting multiple Bayes factor analyses in your write up, then somewhere you only need to cite the software once, at the beginning of the results section. So you might have one sentence like this:

All analyses were conducted using the BayesFactor package in R , and unless otherwise stated default parameter values were used

Notice that I don't bother including the version number? That's because the citation itself includes that information (go check my reference list if you don't believe me). There's no need to clutter up your results with redundant information that almost no-one will actually need. When you get to the actual test you can get away with this:

A test of association produced a Bayes factor of 16:1 in favour of a relationship between species and choice.

Short and sweet. I've rounded 15.92 to 16, because there's not really any important difference between 15.92:1 and 16:1. I spelled out "Bayes factor" rather than truncating it to "BF" because not everyone knows the abbreviation. I indicated exactly what the effect is (i.e., "a relationship between species and choice") and how strong the evidence was. I *didn't* bother indicating whether this was "moderate" evidence or "strong" evidence, because the odds themselves tell you! There's nothing stopping you from including that information, and I've done so myself on occasions, but you don't strictly need it. Similarly, I didn't bother to indicate that I ran the "joint multinomial" sampling plan, because I'm assuming that the method section of my write up would make clear how the experiment was designed. (I might change my mind about that if the method section was ambiguous.) Neither did I bother indicating that this was a *Bayesian* test of association: if your reader can't work that out from the fact that you're reporting a Bayes factor and the fact that you're citing the `BayesFactor` package for all your analyses, then there's no chance they'll understand anything you've written. Besides, if you keep writing the word "Bayes" over and over again it starts to look stupid. Bayes Bayes Bayes Bayes Bayes. See?

17.6.4 Other sampling plans

Up to this point all I've shown you is how to use the `contingencyTableBF()` function for the joint multinomial sampling plan (i.e., when the total sample size N is fixed, but nothing else is). For the Poisson sampling plan (i.e., nothing fixed), the command you need is identical except for the `sampleType` argument:

```
contingencyTableBF(crosstab, sampleType = "poisson" )
```

```
## Bayes factor analysis
## --------------
## [1] Non-indep. (a=1) : 28.20757 ±0%
##
## Against denominator:
##   Null, independence, a = 1
## ---
## Bayes factor type: BFcontingencyTable, poisson
```

Notice that the Bayes factor of 28:1 here is *not* the identical to the Bayes factor of 16:1 that we obtained from the last test. The sampling plan actually does matter.

What about the design in which the row columns (or column totals) are fixed? As I mentioned earlier, this corresponds to the "independent multinomial" sampling plan. Again, you need to specify the `sampleType` argument, but this time you need to specify whether you fixed the rows or the columns. For example, suppose I deliberately sampled 87 humans and 93 robots, then I would need to indicate that the `fixedMargin` of the contingency table is the `"rows"` . So the command I would use is:

```
contingencyTableBF(crosstab, sampleType = "indepMulti", fixedMargin="rows")
```

```
## Bayes factor analysis
## --------------
## [1] Non-indep. (a=1) : 8.605897 ±0%
##
## Against denominator:
##   Null, independence, a = 1
## ---
## Bayes factor type: BFcontingencyTable, independent multinomial
```

Again, the Bayes factor is different, with the evidence for the alternative dropping to a mere 9:1. As you might expect, the answers would be diffrent again if it were the columns of the contingency table that the experimental design fixed.

Finally, if we turn to hypergeometric sampling in which everything is fixed, we get…

```
contingencyTableBF(crosstab, sampleType = "hypergeom")
#Error in contingencyHypergeometric(as.matrix(data2), a) :
#  hypergeometric contingency tables restricted to 2 x 2 tables; see help for contin
```

… an error message. Okay, some quick reading through the help files hints that support for larger contingency tables is coming, but it's not been implemented yet. In the meantime, let's imagine we have data from the "toy labelling" experiment I described earlier in this section. Specifically, let's say our data look like this:

```
toys <- data.frame(stringsAsFactors=FALSE,
        gender = c("girl", "boy"),
        pink = c(8, 2),
        blue = c(2, 8)
        )
```

The Bayesian test with hypergeometric sampling gives us this:

```
contingencyTableBF(toys, sampleType = "hypergeom")

#Bayes factor analysis
#--------------
#[1] Non-indep. (a=1) : 8.294321 @plusorminus0%
#
#Against denominator:
#  Null, independence, a = 1
#---
#Bayes factor type: BFcontingencyTable, hypergeometric
```

The Bayes factor of 8:1 provides modest evidence that the labels were being assigned in a way that correlates gender with colour, but it's not conclusive.