Opinionated Lessons in Statistics

by Bill Press

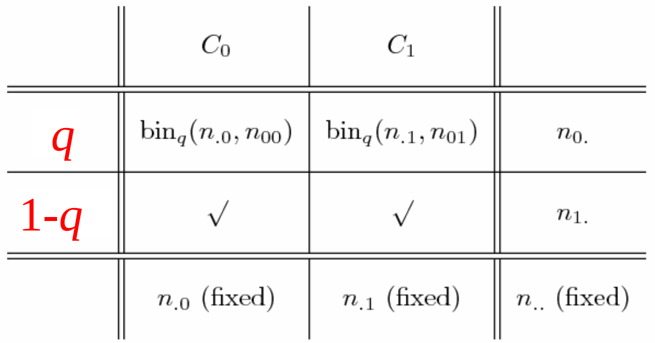
#36 Contingency Tables Have Nuisance Parameters

It’s time to face up to the fact that contingency tables do have nuisance parameters. We need Bayes!

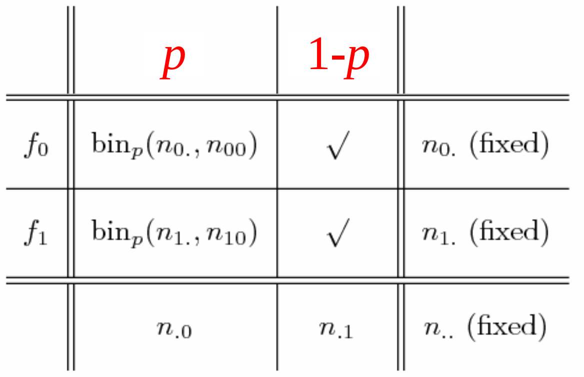
Protocol 1: Retrospective analysis or “case/control study”

Protocol 2: Prospective experiment or “longitudinal study”

Protocol 3: Cross-sectional or snapshot study (no fixed marginals)



	C_0	C_1	
q	$\text{bin}_q(n_{.0}, n_{00})$	$\text{bin}_q(n_{.1}, n_{01})$	$n_{0.}$
1-q	$\surd$	$\surd$	$n_{1.}$
	$n_{.0}$ (fixed)	$n_{.1}$ (fixed)	$n_{..}$ (fixed)



	p	1-p	
f_0	$\text{bin}_p(n_{0.}, n_{00})$	$\surd$	$n_{0.}$ (fixed)
f_1	$\text{bin}_p(n_{1.}, n_{10})$	$\surd$	$n_{1.}$ (fixed)
	$n_{.0}$	$n_{.1}$	$n_{..}$ (fixed)

Segment 33: We saw that the values of p or q or both were necessary to compute the table probability. They are Bayesian “nuisance parameters” to be marginalized over.

Segments 34-35: We took the easy (Fisher) road and pretended that all marginals were fixed. That’s OK as an asymptotic approximation, when all marginals are large. But it’s conceptually wrong!

How shall we estimate the nuisance parameters p and/or q?

Remember "conjugate distributions"? As before, we want to estimate parameters from observed counts. Beta is conjugate to Binomial:

$$P(n|N,q) = \binom{N}{n} q^n (1-q)^{N-n}$$

$$P(q|N,n) \propto q^n (1-q)^{N-n} P(q)$$

Bayes, with prior.

A "conjugate prior" is one that preserves the functional form of the distribution.

$$P(q) \propto q^\alpha (1-q)^\beta$$

($\alpha = \beta = 0$ is a perfectly good choice: flat prior on *q*)

So the conjugate distribution is

$$\begin{aligned} P(q|N,n) &= \frac{q^{n+\alpha}(1-q)^{N-n+\beta}}{\int_0^1 q^{n+\alpha}(1-q)^{N-n+\beta}dq} \\ &= \frac{\Gamma(N+\alpha+\beta+2)}{\Gamma(n+\alpha+1)\Gamma(N-n+\beta+1)} q^{n+\alpha}(1-q)^{N-n+\beta} \\ &\sim \mathrm{Beta}(n+\alpha+1, N-n+\beta+1) \end{aligned}$$

This Beta distribution has

$$\text{mean} = \frac{n+\alpha+1}{N+\alpha+\beta+1} \qquad \text{var} = \frac{(n+\alpha+1)(N-n+\beta+1)}{(N+\alpha+\beta+2)^2(N+\alpha+\beta+3)}$$

Matlab, Mathematica, and NR3 all have methods for generating random Beta deviates

If we generalize to contingency tables other than 2x2, Dirichlet is the relevant conjugate distribution to Multinomial

Multinomial distribution (you can derive it by “repeated binomial” or combinatorics as we did earlier):

$$P(n_1, n_2, \ldots | N, q_1, q_2, \ldots) = \frac{N!}{n_1! n_2! \cdots} q_1^{n_1} q_2^{n_2} \cdots \,, \qquad (\sum n_i = N,\ \sum q_i = 1)$$

Conjugate distribution, using conjugate priors:

$$P(q_1, q_2, \ldots | N, n_1, n_2, \ldots) \propto q_1^{n_1+\alpha_1} q_2^{n_2+\alpha_2} \cdots$$

the Dirichlet distribution

Normalization turns out to be:

$$P(q_1, q_2, \ldots | N, n_1, n_2, \ldots) = \frac{\Gamma(N + \alpha_1 + 1 + \alpha_2 + 1 + \cdots)}{\Gamma(n_1 + \alpha_1 + 1)\Gamma(n_2 + \alpha_2 + 1) \cdots} q_1^{n_1+\alpha_1} q_2^{n_2+\alpha_2} \cdots$$

Rather amazingly, there is a simple way to generate a (non-independent) set of ***q*** deviates from an independent set of Gamma deviates:

$$y_i \sim \text{Gamma}(n_i + \alpha_i + 1), \quad p(y) = \frac{y^{n_i+\alpha_i} e^{-y}}{\Gamma(n_i + \alpha_i + 1)} \qquad q_i = y_i \Big/ \sum_i y_i$$

(In fact, in the case with I=2, this is how Beta deviates [previous slide] are usually generated.)

So let's reanalyze assuming that the condition (column) marginals were fixed by the protocol, and we Bayes-sample (i.e. marginalize) the row probabilities:

(You'll never believe my encapsulated function unless I go through an example!)

	C_0	C_1
f_0	8	3
f_1	16	26

$$\begin{matrix} 8 & 16 \\ 3 & 26 \end{matrix}$$

```
>> table = [8 3; 16 26]
table =
     8     3
    16    26
>> marfix = sum(table,1)'
```

column marginals (transposed)

```
marfix =
    24
    29
>> marvar = sum(table,2)'
```

row marginals (transposed)

```
marvar =
    11    42
>> gammas = gamrnd(marvar+1,1)
```

~Gamma(n_i+1)

```
gammas =
   12.1000   44.5735
>> q = gammas ./ sum(gammas)
```

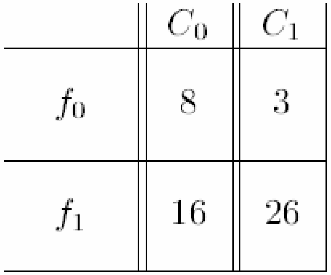
generated (random) row probabilities q_i

```
q =
    0.2135    0.7865
>> qmat = repmat(q,[size(table,2),1])
qmat =
    0.2135    0.7865
    0.2135    0.7865
>> tabout = mnrnd(marfix,qmat)'
tabout =
     4     8
    20    21
```

finally, we generate multinomial deviates for each column, using the generated row probabilities

The reason everything is done in the transpose is because of the way that Matlab's mnrnd function expects its arguments to be shaped. Sorry about that!

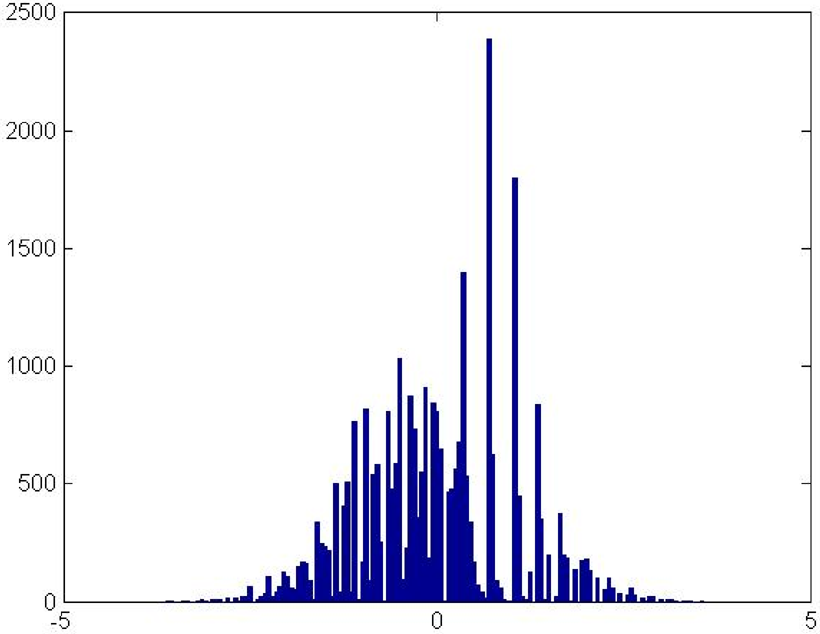
Encapsulate the sampling process into a function.
Then, generate a bunch of samples and look at their Wald statistics.

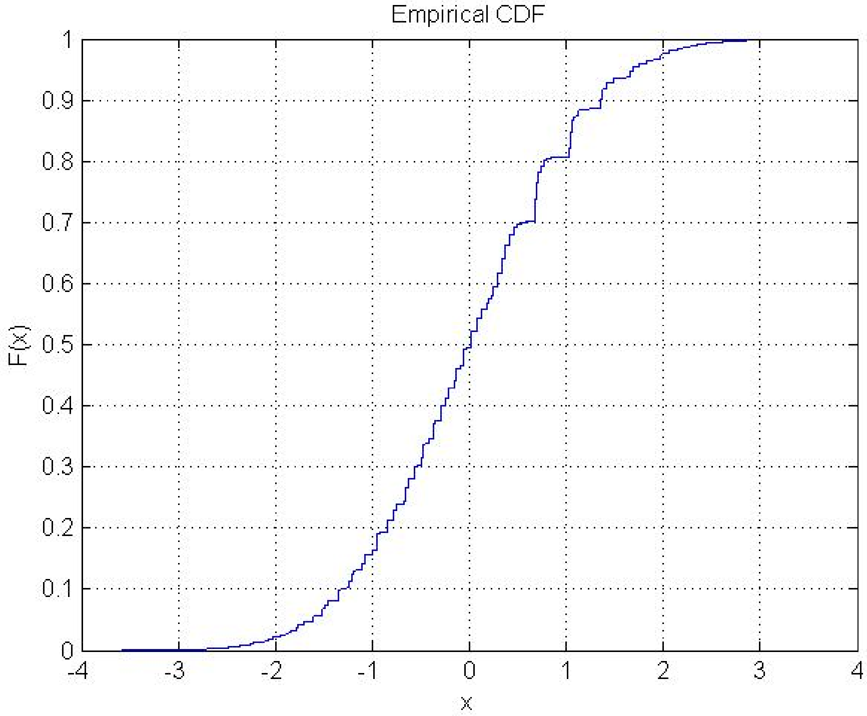


	C_0	C_1
f_0	8	3
f_1	16	26

```
function tabout = tabnullsamp(tabin)
marfix = sum(tabin,1)';
marvar = sum(tabin,2)';
q = gamrnd(marvar+1,1);
qmat = repmat(q./sum(q),[size(tabin,2),1]);
tabout = mnrnd(marfix,qmat)';

wald(table)
ans =
       2.0542

tabnullsamp(table), tabnullsamp(table)
ans =
     6     7
    18    22
ans =
     7     9
    17    20

wald(tabnullsamp(table))
ans =
       1.0392

samps = arrayfun(@(x) wald(tabnullsamp(table)), 1:30000);
hist(samps,-4:.05:4)
cdfplot(samps)
```

There are still discreteness effects (after all, these are integer tables), but they are less troubling:





```
pval = numel(samps(samps>=wald(table)))/numel(samps)
pvaltt = (numel(samps(samps>=wald(table)))+numel(samps(samps<= -wald(table))) )/numel(samps)
pval =
     0.022967
pvaltt =
     0.040067
```

one-tail vs. two-tail now much more reasonble

This is probably the most honest answer that we can get for the significance of this particular contingency table.

	C_0	C_1
f_0	8	3
f_1	16	26

Let’s reanalyze the maternal drinking data using the same methodology, but (as we did before) with the Pearson statistic:

```
function chis = pearson(table)
nhtable = sum(table,2)*sum(table,1)/sum(sum(table));
chis = sum(sum((table-nhtable).^2./nhtable));


table = [17066 14464 788 126 37; 48 38 5 1 1]'
table =
       17066              48
       14464              38
         788               5
         126               1
          37               1

pearson(table)
ans =
       12.082

samps = arrayfun(@(x) pearson(tabnullsamp(table)), 1:10000);
hist(samps,0:0.25:90)
```

transpose to make the unfixed marginals be the rows, as before

the (unrealistic, but don’t worry now) scenario is something like: case-control study where malformation-present came from hospitals, malformation-absent came from a door-to-door survey

columns are the “conditions”

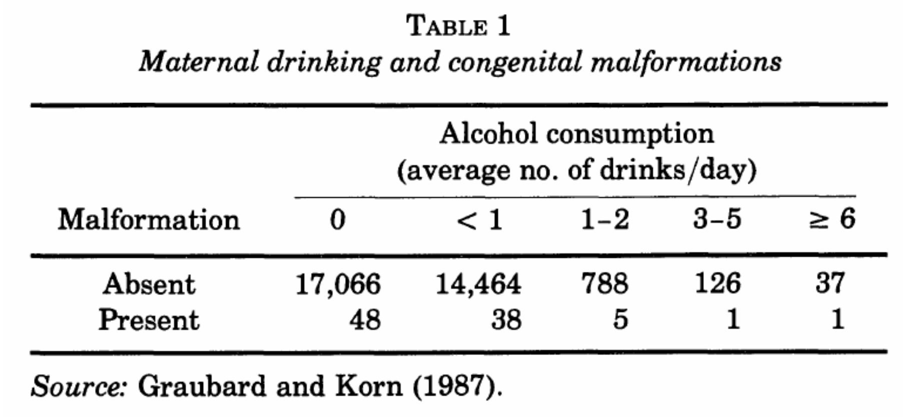
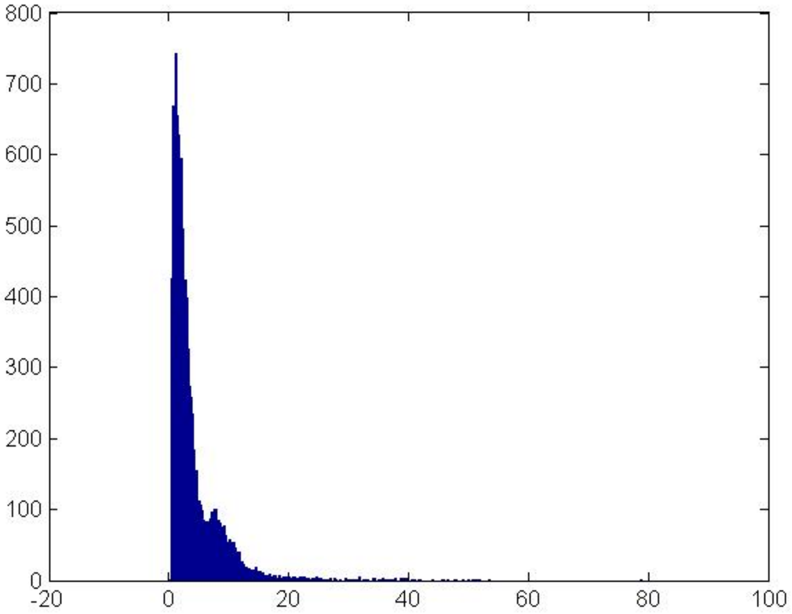


TABLE 1
Maternal drinking and congenital malformations

	Alcohol consumption (average no. of drinks/day)				
Malformation	0	< 1	1–2	3–5	≥ 6
Absent	17,066	14,464	788	126	37
Present	48	38	5	1	1

Source: Graubard and Korn (1987).

Giving the results



```
pval = numel(samps(samps>=pearson(table)))/numel(samps)
pval =
       0.0408
```

This is different from the Fisher Exact test (permutation test), which, we saw, gave $p=0.0380$. Fisher Exact is not exact. In fact, it’s wrong!

Actually, it seems likely that this data was a cross-sectional study with no fixed marginals. If so, a better sampling of the null hypothesis would be:

```
function tabout = tabnullsamp2(tabin)
marcol = sum(tabin,1);
marrow = sum(tabin,2);
ntot = sum(marcol);
q = gamrnd(marcol+1,1);
q = q./sum(q);
p = gamrnd(marrow+1,1);
p = p./sum(p);
pq = p * q;
tabout = reshape(mnrnd(ntot,pq(:)'),size(tabin));
```

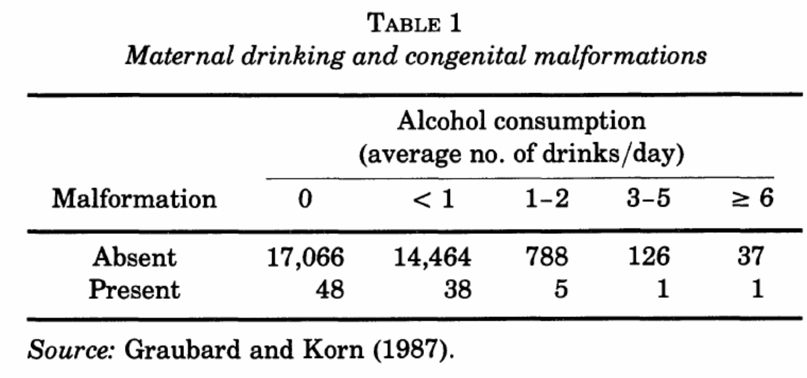


TABLE 1
Maternal drinking and congenital malformations

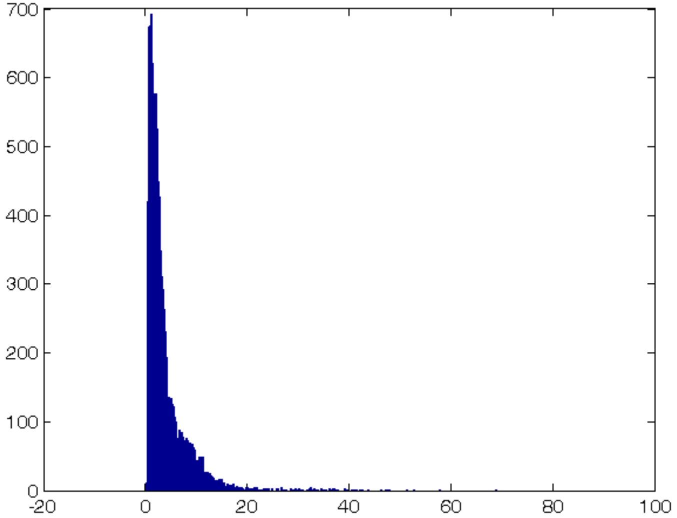
	Alcohol consumption (average no. of drinks/day)				
Malformation	0	< 1	1–2	3–5	≥ 6
Absent	17,066	14,464	788	126	37
Present	48	38	5	1	1

Source: Graubard and Korn (1987).

```
samps = arrayfun(@(x) pearson(tabnullsamp2(table)), 1:10000);
hist(samps,0:0.25:90)
pval = numel(samps(samps>pearson(table)))/numel(samps)
pval =
    0.0445
```

← different from previous: protocol matters!

This would likely be the best answer if the table were nominal, not ordinal. But, since it is ordinal, the Segment 35’s analysis using difference of means is more powerful.



You now know all you need to know about contingency tables – and much more than almost everyone who uses them!