Various Issues in Fitting Contingency Tables

Statistics 149

Spring 2006

Complete Tables with Zero Entries

In contingency tables, it is possible to have zero entries in a table. There are two ways this can occur

- Structural zeros: These occur when the sampling scheme forces them to be 0. For example in the Wave Damage to Cargo Ship example, they classified the number of damage incidents by the 3 factors.

  - Ship type: A - E
  - Year of construction: 1960-64, 1965-69, 1970-74, 1975-1979
  - Period of operation: 1960-74, 1975-1979

  In this example, any cell involving operation period = 1960-1974 and construction year = 1975-1979 must have a count = 0. Actually there is one more structural zero. The cell for ship type = E, construction year = 1960-1964, and operation period = 1975-1979 had service time = 0, which forces the number of damage incidents to 0.

These situations are easily handled by just removing these observations from the data set. Note that the total number of observations won't be IJK but $IJK - \nu_s$, where ν_s is the number of structural zeros in the data set. In this example $\nu_s = 0$, one for each ship type plus the one cell that had a 0 service times, even though it could have been positive. So the effective number of observations is $34 = 5 \times 4 \times 2 - 6$.

```
> summary(wave.glm)

Call:
glm(formula = Damage ~ Type + Construct + Operation,
    family = poisson(), data = wave2, offset = log(Service))

    Null deviance: 146.328  on 33  degrees of freedom
Residual deviance:  38.695  on 25  degrees of freedom
```

- Sampling zeros: Cell counts $= 0$ can also occur at random. This will tend to happen when μ is small (equivalently when π is close to 0)

  Example: Food poisoning (Bishop, Fienberg, & Holland, pp 90-91)

  The following data are from and epidemiologic study following an outbreak of food poisoning at an outing held for personnel of an insurance company. Questionnaires were completed by 304 of the 320 persons attending. Of the food eaten, interest focused on potato salad and crabmeat.

	Crabmeat		No Crabmeat		
Potato Salad	Yes	No	Yes	No	Total
Ill	120	4	22	0	146
Not Ill	80	31	24	23	158

  There is no reason, apriori, to assume that people who got ill must have eaten at least one of the potato salad or crabmeat.

Model	df	X_p^2	X^2	$\hat{\mu}_{122}$
(IP, IC, PC)	1	1.70	2.74	1.08
(IP, IC)	2	7.21	7.64	0.60
(IP, PC)	2	5.09	6.48	1.59
(IC, PC)	2	44.35	53.66	7.33

The 0 cell for these models doesn't seem to cause a problem. For example, for the model (IP, IC, PC)

```
> summary(poison.ic.ip.cp)
Call: glm(formula = count ~ .^2, family = poisson(),
          data = poison)

Deviance Residuals:
       1         2         3         4         5         6
-0.09857   0.59995   0.23481  -1.47176   0.12164  -0.19230
       7         8
-0.21783   0.22947
```

```
Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)         3.0873     0.2102  14.686  < 2e-16 ***
illYes             -3.0075     0.5676  -5.299 1.17e-07 ***
crabYes             0.3811     0.2697   1.413   0.1577
potatoYes           0.1349     0.2837   0.476   0.6343
illYes:crabYes      0.6097     0.3170   1.923   0.0544 .
illYes:potatoYes    2.8259     0.5362   5.270 1.36e-07 ***
crabYes:potatoYes   0.7651     0.3432   2.229   0.0258 *
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 295.2526  on 7  degrees of freedom
Residual deviance:   2.7427  on 1  degrees of freedom
AIC: 53.074

Number of Fisher Scoring iterations: 5
```

Everything looks fine here.

Now lets fit the saturated model.

```
> summary(poison.icp)

Call:
glm(formula = count ~ .^3, family = poisson(), data = poison,
    epsilon = 1e-16, maxit = 50)

Deviance Residuals:
[1]  0  0  0  0  0  0  0  0
```

To be expected since $\hat{\mu}_{ijk} = n_{ijk}$

```
Coefficients:
                          Estimate Std. Error   z value Pr(>|z|)
(Intercept)              3.135e+00  2.085e-01    15.037   <2e-16 *
illYes                  -5.544e+01  6.711e+07 -8.26e-07    1.000
crabYes                  2.985e-01  2.752e-01     1.085    0.278
potatoYes                4.256e-02  2.918e-01     0.146    0.884
illYes:crabYes           5.339e+01  6.711e+07  7.96e-07    1.000
illYes:potatoYes         5.535e+01  6.711e+07  8.25e-07    1.000
crabYes:potatoYes        9.055e-01  3.604e-01     2.512    0.012 *
illYes:crabYes:potYes -5.290e+01  6.711e+07 -7.88e-07    1.000
---
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 2.9525e+02  on 7  degrees of freedom
Residual deviance: 1.1546e-14  on 0  degrees of freedom
AIC: 52.332

Number of Fisher Scoring iterations: 50
```

This doesn't look so good. The `glm` function didn't converge here, as the number of iterations happens to be the `maxit`. Also notice that some of the standard errors for the λs are huge.

It ends up that sampling zeros can cause problems with estimating the λs in some cases. Whether a problem occurs depends on the model to be fit and the layout of the zeros in the table.

Note that even though there may be problems with estimating the λs, the model may still be well defined in terms of the πs and odds ratio type measures (maybe going to more complicated structures)

Lets look at a couple of 2×2 table examples and fit the hypothetical examples and try to fit the independence model in each case

Observed:

$$\begin{bmatrix} 20 & 0 \\ 0 & 5 \end{bmatrix} \qquad \begin{bmatrix} 20 & 5 \\ 0 & 0 \end{bmatrix}$$

Fitted:

$$\begin{bmatrix} 16 & 4 \\ 4 & 1 \end{bmatrix} \qquad\qquad \begin{bmatrix} 20 & 5 \\ 0 & 0 \end{bmatrix}$$

If we use the upper left cell as the reference cell, the MLEs for the λs satisfy

$$\begin{aligned} \hat{\lambda} &= \log \hat{\mu}_{11} \\ \hat{\lambda}^X &= \log \hat{\mu}_{22} + \log \hat{\mu}_{21} - \log \hat{\mu}_{12} - \log \hat{\mu}_{11} \\ \hat{\lambda}^Y &= \log \hat{\mu}_{22} - \log \hat{\mu}_{21} + \log \hat{\mu}_{12} - \log \hat{\mu}_{11} \end{aligned}$$

where

$$\hat{\mu}_{ij} = \frac{n_{i+} n_{+j}}{n}$$

For the first table we get

$$\begin{aligned}\hat{\lambda} &= \log 16 = 2.7726 \\ \hat{\lambda}^X &= \log 1 + \log 4 - \log 4 - \log 16 = -1.3863 \\ \hat{\lambda}^Y &= \log 1 - \log 4 + \log 4 - \log 16 = -1.3863\end{aligned}$$

However for the second table we get

$$\begin{aligned}\hat{\lambda} &= \log 20 = 2.9957 \\ \hat{\lambda}^X &= \log 0 + \log 0 - \log 5 - \log 20 = -\infty \\ \hat{\lambda}^Y &= \log 0 - \log 0 + \log 5 - \log 20 = -\infty\end{aligned}$$

In this case the fitted cells with 0 are causing trouble.

Lets consider another example

	Z1		Z2	
	Y1	Y2	Y1	Y2
X1	0	b	e	f
X2	c	d	g	0

Lets consider the model (XY, XZ, YZ). Under this model

$$\hat{\mu}_{ij+} = n_{ij+} \qquad \hat{\mu}_{i+k} = n_{i+k} \qquad \hat{\mu}_{+jk} = n_{+jk}$$

Now suppose that $\hat{\mu}_{111} = \Delta > 0$. Then the table of expected counts must look like

	Z1		Z2	
	Y1	Y2	Y1	Y2
X1	Δ	$b-\Delta$	$e-\Delta$	$f+\Delta$
X2	$c-\Delta$	$d+\Delta$	$g+\Delta$	$-\Delta$

Which gives a negative $\hat{\mu}$. Thus $\hat{\mu}_{111} = \hat{\mu}_{222} = 0$ which further implies $\hat{\mu}_{ijk} = n_{ijk}$.

In addition the $\hat{\lambda}$s aren't well defined as the calculations for many of them involve $\log 0$.

If we take an example of this table we get

```
Call:
glm(formula = count ~ (X + Y + Z)^2, family = poisson(),
    data = zero3)
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   -23.81   46570.93  -0.001        1
X2             25.42   46570.93   0.001        1
Y2             26.80   46570.93   0.001        1
Z2             26.11   46570.93   0.001        1
X2:Y2         -25.70   46570.93  -0.001        1
X2:Z2         -25.93   46570.93  -0.001        1
Y2:Z2         -25.70   46570.93  -0.001        1
```

In this case the intercept is actually $\log 0 = \infty$. The calculated result is a result of the convergence criterion used in `glm` and the finite precision of calculations in **R**.

Note that the parameterization selected can't be used to solve the problem. Different parameterizations will exhibit the problem differently, but it will always be there.

Effects on the Degrees of Freedom

In general, the degrees of freedom are determined by

$$df = T_e - T_p$$

where T_e is the number of cells in the table where μ is estimated and T_p is the number of parameters to be fitted. Note this is just the same formula as before.

If there are no cells with fitted zeros, this is correct formula.

However this is not correct if there are fitted zeros which can occur two ways

1. λ terms in the model cannot be estimated due to the arrangements of sampling zeros, even though the specifying configurations may have all non-zero values.
2. Zero cells lead to empty cells for the specifying configuration.

In the case of fitted zeros, the formula for degrees of freedom needs to be modified to

$$df = (T_e - z_e) - (T_p - z_p)$$

where z_e is the number of cells with fitted zeros and z_p is the number of parameters that can't be estimated.

So for the artificial 3-way table with the model (XY, XZ, YZ)

$$T_e = 8 \qquad T_p = 7 \qquad z_e = 2 \qquad z_p = 1$$

giving

$$df = (8 - 2) - (7 - 1) = 0$$

which seems reasonable since $\hat{\mu}_{ijk} = n_{ijk}$ for this model.

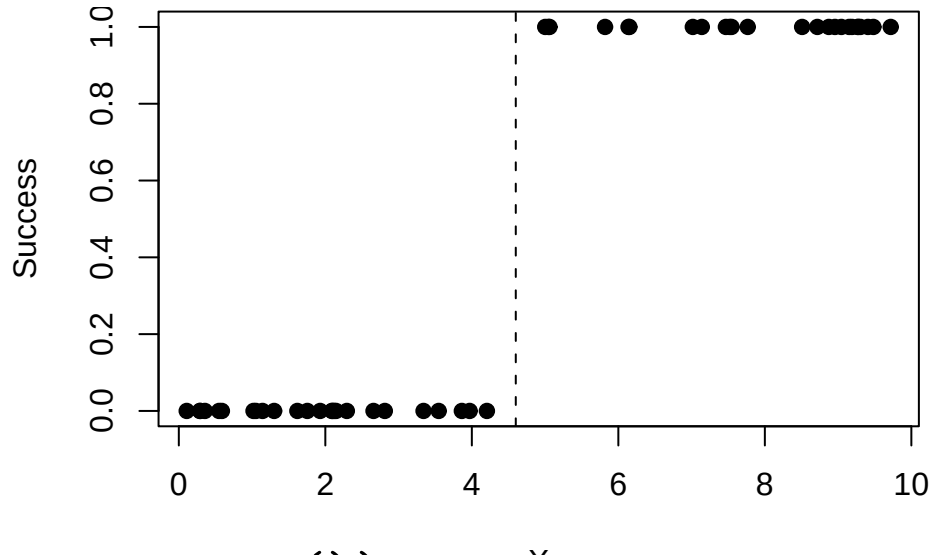
For the

Note that z_p is often difficult to determine. For this problem it happens to be 1. It ends up you can only fit 2 of the 3 two factor interactions at one time.

Separating Hyperplanes in Logistic Regression

There is a related problem in logistic regression where is it not possible to get parameter estimates.

The plot to the right shows an example of when the problem occurs.




```
> sp.glm <- glm(ysp ~ xsp, family=binomial())
Warning messages:
1: algorithm did not converge in:
   glm.fit(x = X, y = Y, weights = weights, start = start,
           etastart = etastart,
2: fitted probabilities numerically 0 or 1 occurred in:
   glm.fit(x = X, y = Y, weights = weights, start = start,
           etastart = etastart,
```

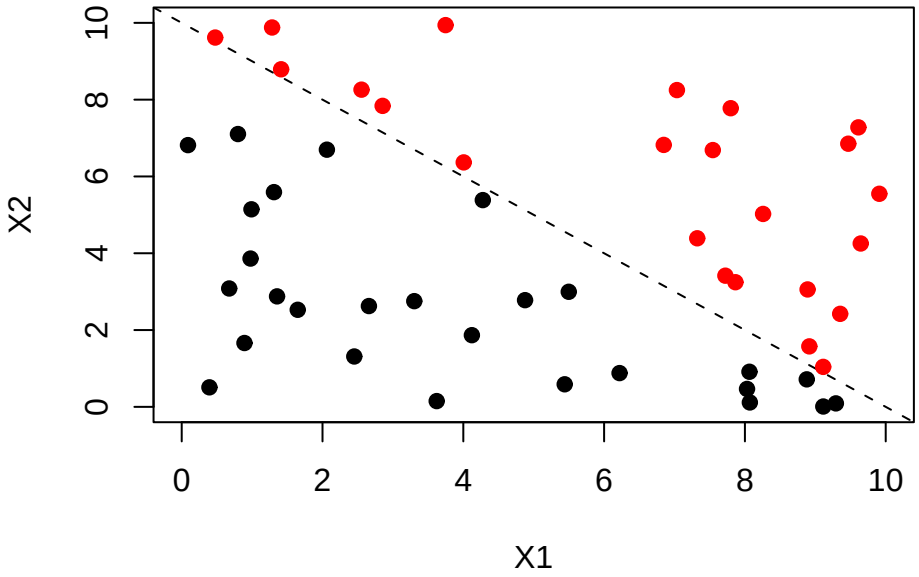
```
> summary(sp.glm)

Call: glm(formula = ysp ~ xsp, family = binomial())

Deviance Residuals:
       Min          1Q      Median          3Q         Max
-6.008e-05  -2.107e-08   0.000e+00   2.107e-08   5.200e-05

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   -234.32  119856.80  -0.002    0.998
xsp             50.93   26078.75   0.002    0.998

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 6.9315e+01  on 49  degrees of freedom
Residual deviance: 6.7379e-09  on 48  degrees of freedom
AIC: 4
Number of Fisher Scoring iterations: 25
```

In this example the data was generated by $y = I(x > 5)$.

When you try to fit this example, the iterates of $\hat{\beta}_1$ in the IRSL algorithm will continue to increase without bound. The only reason that **R** stopped and gave an answer is that the `glm` function has a `maxit` option which forces the function to terminate.

When there is a single predictor x, this problem will exhibit itself whenever you have a dataset with the property

$$y = I(x > x_c) \qquad \text{or} \qquad y = I(x < x_c)$$



When there is a higher dimensional predictor (more xs), this problem can also occur. In two dimensions, the problem occurs if you can draw a line through the scatterplot of the predictors and have all the successes on one side and all the failures on the other.

In 3 dimensions, you look for a separating plane and in 4 and higher dimensions you look for separating hyperplanes.

Usually this problem occurs when you have sparce data, particularly in the range of xs where $\pi(x)$ changes alot. When it comes to study design, you need to choose level of the predictors and number of observations so that the sample proportions should be bounded away from 0 and 1.

Direct Estimates

While for most purposes today being able to get direct estimates of $\hat{\mu}$s is less important today, due to modern computing hardware and software, it does have its uses.

For example, Monte Carlo techniques may require generation of these fits many times, and it may be quicker to generate them directly that to use iterative procedures such as iteratively reweighted least squares.

As mentioned before, not every log linear model has direct estimates. There is a simple algorithm that will determine whether a model has direct estimates based on the sufficient configuration (the notation discussed last time for describing models). The algorithm involves the following steps:

1. Relabel any group of variables that always appear together as a single variable.

2. Delete any variable that occurs in every configuration.

3. Delete any variable that occurs in every configuration.

4. Remove any redundant configuration.

5. Repeat steps 1 - 4 until

   (a) No more than 2 configurations remain $\Longrightarrow$ **Direct Estimates**
   (b) No further steps possible $\Longrightarrow$ **No Direct Estimates**

Examples:

1. (AB, AC, BD)

   - $\xrightarrow{3}$ (AB, AC, B) – D occurs only once
   - $\xrightarrow{4}$ (AB, AC) – B redundent [in AB]
   - $\xrightarrow{3}$ (AB, A) – C occurs only once
   - $\xrightarrow{4}$ (AB) – A redundent [in AB]
   - ≤ 2 configurations $\Longrightarrow$ Direct Estimates

2. (AB, AC, BC)

   - Nothing can be relabeled by step 1
   - Nothing occurs in every configuration – drop nothing
   - Nothing occurs in just on configuration – drop nothing
   - No redundent configurations – drop nothing
   - No further steps possible and > 2 configurations $\Longrightarrow$ No Direct Estimates

3. (ABC, BCD, AD)

   - $\xrightarrow{1}$ (A[BC]', [BC]'D, AD) = (AE, ED, AD) – relabel AB as E
   - No further steps possible – this is equivalent to example 2.
   - $>$ 2 configurations $\Longrightarrow$ No Direct Estimates

4. (BCE, ACF, EG, ABD, ABC)

   - $\xrightarrow{3}$ (BCE, AC, E, ABD, ABC) – F & G removed as they only occur once
   - $\xrightarrow{4}$ (BCE, ABD, ABC) – AC [part of ABC] & E [part of BCE] are now redundant
   - $\xrightarrow{3}$ (BC, AB, ABC) – D & E occur only once
   - $\xrightarrow{4}$ (ABC) – AB & BC both occur in ABC
   - $\leq$ 2 configurations $\Longrightarrow$ Direct Estimates

Now that we can figure out when direct estimates occur, what are they? There is a simple scheme to construct them

1. Numerator has entries from the sufficient configuration

2. Denominator has entries from redundant configurations due to overlapping

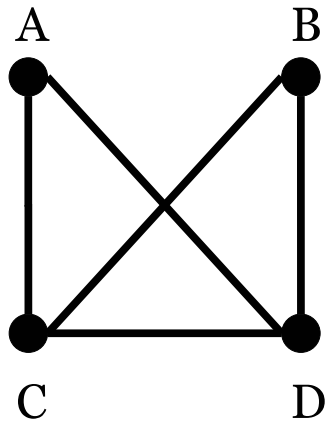
3. Powers of ns to ensure right order of magnitude

Note: When considering the sufficient configuration, there should be no components that should be eliminated by step 4 of the algorithm. So if the model (ACD, BCD, A) is given, the sufficient configuration is (ACD, BCD).

Examples:

1. (ACD, BCD) – Conditional on C & D, A & B are independent

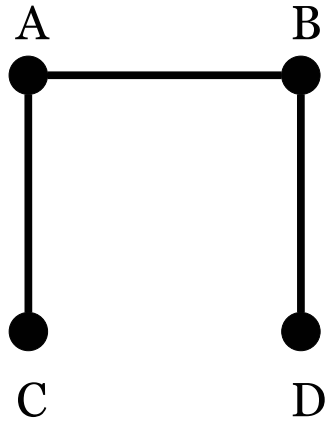
$$\hat{\mu}_{ijkl} = \frac{n_{i+kl} n_{+jkl}}{n_{++kl}}$$



(ACD, BCD)

2. (AB, AC, BD) – Many conditional independence relationships

$$\hat{\mu}_{ijkl} = \frac{n_{ij++} n_{i+k+} n_{+j+l}}{n_{i+++} n_{+j++}}$$



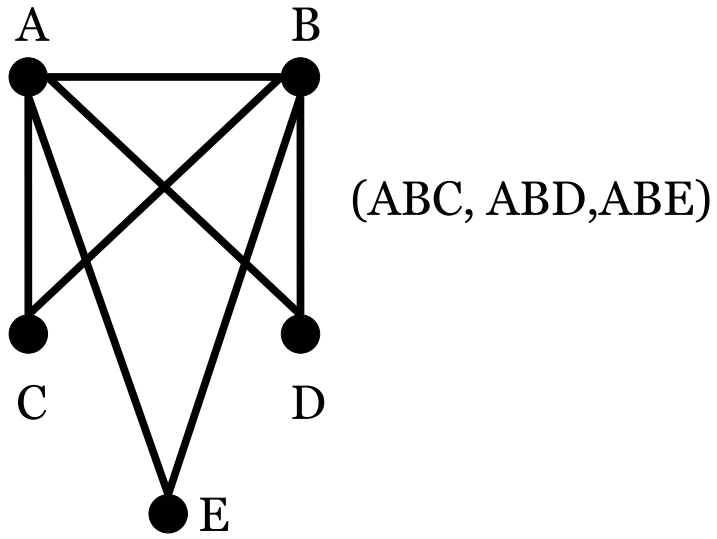
(AB, AC, BD)

Slight notation change: Let n^X correspond to fixing the levels of variables listed in X and adding over the other variables. For example in a model with variables A, B, C, and D,

$$n^{AB} = n_{ij++} \qquad n^C = n_{++k+} \qquad n = n_{++++}$$

3. (ABC, ABD, ABE) – C, D, and E are mutually independent conditional on A and B.

$$\hat{\mu} = \frac{n^{ABC} n^{ABD} n^{ABE}}{(n^{AB})^2}$$



4. (BCE, ACF, EG, ABD, ABC)

$$\hat{\mu} = \frac{n^{ABC} n^{ABD} n^{BCE} n^{ACF} n^{EG}}{n^{AB} n^{AC} n^{BC} n^{E}}$$

This algorithm is related to the fact that X is an element of the sufficient configuration describing a model, then

$$\hat{\mu}^X = n^X$$

Another way of thinking of this, the sufficient configuration describes which marginal tables are fixed when considering the corresponding estimated table for a model.