Multinomials and the CJS Model

WILD 502 - Jay Rotella

To gain familiarity with the multinomial, it is helpful to work with drawing values from it. We use the function 'rnultinom' to generate several datasets. We start using the example from the textbook for passes, where the probabilities are for a completion, incompletion, and interception. The code generates 10 replications of releases (trials) of 100 individuals on occasion 1.

```
x <- rmultinom(10, size = 100, prob = c(0.6, 0.38, (1 - 0.6 - 0.38)))
row.names(x) <- c("complete", "incomplete", "intercept")
x
```

```
##            [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## complete     59   62   58   59   58   61   60   60   60    52
## incomplete   40   37   39   40   39   38   40   39   37    46
## intercept     1    1    3    1    3    1    0    1    3     2
```

CJS Example Using Multinomial

We'll set constant parameter values for ϕ_i and p_i and use them to (1) calculate the probability of encounter history that's possible in a 3-year study, (2) see 2 versions of probability statements for encounter histories that end in 0's, and (3) check that the probabilities sum to 1.0. Examine the probability statement for the 100 encounter history: an animal can have this history by living through 2, 1, or 0 of the intervals, i.e., this type of history doesn't provide a lot of information. If you have low values of p, you can end up with a lot of these, and your precision will suffer.

```
phi <- 0.8
p <- 0.3
(pr111 <- phi * p * phi * p)
```

```
## [1] 0.0576
```

```
(pr110 <- phi * p * ((1 - phi) + phi * (1 - p)))
```

```
## [1] 0.1824
```

```
(pr110.alt <- phi * p * (1 - (phi * p)))
```

```
## [1] 0.1824
```

```
(pr101 <- phi * (1 - p) * phi * p)
```

```
## [1] 0.1344
```

```
(pr100 <- (phi * (1 - p) * phi * (1 - p) +
           phi * (1 - p) * (1 - phi) +
           (1 - phi)))
```

```
## [1] 0.6256
```

```
(pr100.alt <- 1 - (pr111 + pr110 + pr101))
```

```
## [1] 0.6256
```

```
round(c(pr111, pr110, pr101, pr100, sum(pr111, pr110, pr101, pr100)), 3)
```

```
## [1] 0.058 0.182 0.134 0.626 1.000
```

Generate CJS Capture Histories with Random Multinomials

You can quickly generate summarized capture history information for a CJS study using the `rmultinom` function.

```
set.seed(502) # you can set the seed to make the results repeatable
(rm <- rmultinom(10, size = 250, prob = c(pr111, pr110, pr101, pr100)))
```

```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
## [1,]   17   17   14   10   12   12   15   16   19    21
## [2,]   45   51   43   41   50   47   44   48   43    43
## [3,]   33   30   31   37   30   34   33   28   26    32
## [4,]  155  152  162  162  158  157  158  158  162   154
```

```
# Check proportions
round(rm/250, 3)
```

```
##        [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]  [,9] [,10]
## [1,] 0.068 0.068 0.056 0.040 0.048 0.048 0.060 0.064 0.076 0.084
## [2,] 0.180 0.204 0.172 0.164 0.200 0.188 0.176 0.192 0.172 0.172
## [3,] 0.132 0.120 0.124 0.148 0.120 0.136 0.132 0.112 0.104 0.128
## [4,] 0.620 0.608 0.648 0.648 0.632 0.628 0.632 0.632 0.648 0.616
```

```
# compare average proportions against original data
round(apply(rm, 1, mean)/250,3)
```

```
## [1] 0.061 0.182 0.126 0.631
```

```
round(c(pr111, pr110, pr101, pr100),3)
```

```
## [1] 0.058 0.182 0.134 0.626
```

Analyze Data from Each Simulation in MARK

phi	se.phi	p	se.p
0.771	0.092	0.328	0.061
0.719	0.082	0.372	0.064
0.765	0.100	0.302	0.061
0.934	0.136	0.217	0.051
0.750	0.097	0.316	0.062
0.826	0.109	0.278	0.057
0.788	0.100	0.304	0.060
0.695	0.083	0.367	0.066
0.654	0.076	0.397	0.069
0.734	0.082	0.369	0.063

Summarize Results of Many Simulated Datasets

The above example only used 10 simulated datasets, which is nice for displaying the steps but inadequate for properly assessing the expected outcomes, which would require many simulated datasets, e.g., thousands. Ignoring the fact that we've only done 10 simulations, let's check the average performance and the variability in performance of the CJS model for the data and estimates generated above. Notice that here because we have multiple estimates of *phi* and *p*, we can empirically estimate the SE of each by calculating the SD of all the estimates of *phi* and *p*.

parameter	mean	sd	min	max
phi	0.764	0.077	0.654	0.934
se.phi	0.096	0.017	0.076	0.136
p	0.325	0.054	0.217	0.397
se.p	0.062	0.005	0.051	0.069

Simulations in Program MARK

You can conduct simulations for various types of capture-recapture studies in Program MARK. Appendix A of the on-line Cooch and White book has really helpful material on the topic, which we can review later in the semester if you like. Conducting simulations can be very helpful in understanding how models operate, evaluating the impacts of violating assumptions, and designing studies. Having simulation code becomes especially helpful when you have many occasions where working out all the probability statements can be tedious at best, and/or when covariates are involved.