Fitting Cormack-Jolly-Seber Models to Capture-Recapture Data with ADMB and R2admb

Jeff Laake

November 15, 2012

Cormack-Jolly-Seber models are often used to estimate survival from release-recapture data in which animals are marked and released and then possibly recaptured at a set of discrete sampling occasions if they survive. Specification of a CJS model includes defining a model for p, capture probability and ϕ, survival probability. The basic data are a matrix of capture histories with a row for each of the n animals and a column for each of the m sampling occasions. Let ω_i be the capture history for the i^{th} animal. It is a vector of m values $(\omega_{i1}, \omega_{i2}, ..., \omega_{im})$ where ω_{ij} is 1 if animal i was initially released or captured on occasion j and 0 otherwise. Let f_i be the occasion the animal was released and l_i the last occasion the animal was seen. The CJS likelihood conditions on the first occasion when the animal is released. Following Pledger et al. (2003), the probability of a particular capture can be composed hierarchically by conditioning on an unknown time of death and summing over the possible times of death. Let $Pr(\omega_i|\, f_i,\, d)$ be the probability of observing the capture history given death occurred in the time interval after occasion d. The probability death occurs after occasion d is represented as $Pr(d\,|\,f_i)$. The possible times of death are those after the last occasion the animal was seen until the last possible occasion m. The probability of the capture history $Pr(\omega_i) = Pr(\omega_i|\, f_i) = \sum_{d=l_i}^{m} Pr(\omega_i|\, f_i,\, d) Pr(d\,|\,f_i)$ implicitly depends on the first release occasion but it is dropped from the notation. The probability of a capture history expressed as a function of the parameters p_{ij} for capture probability of the i^{th} animal on occasion j and ϕ_{ij} for survival probability of the i^{th} animal between occasion j and $j+1$ is:

$$Pr(\omega_i) = \sum_{d=l_i}^{k_i} \prod_{j=f_i+1}^{d} p_{ij}^{\omega_{ij}} (1-p_{ij})^{(1-\omega_{ij})} \left(\prod_{j=f}^{d-1} \phi_{ij} \right) (1-\phi_{id})$$

where $\phi_{im} = 0$. For animals that are not released on the last occasion they were seen

(loss on capture) typically because they die due to handling, the probability of the capture history is:

$$Pr(\omega_i) = \prod_{j=f_i+1}^{l_i} p_{ij}^{\omega_{ij}}(1-p_{ij})^{(1-\omega_{ij})} \left(\prod_{j=f}^{l_i-1} \phi_{ij}\right)$$

The log-likelihood is:

$$lnL = \sum_{i=1}^{n} ln(Pr(\omega_i))$$

but it can also be expressed as:

$$lnL = \sum_{i=1}^{n'} r_i ln(Pr(\omega_i))$$

where n' is the number of unique values which includes the capture history and covariates used for the model and r_i is the frequency.

Models are specified for p and ϕ as linear functions of covariates for a particular link function that restricts the parameters to the unit interval. Let $X_\phi$and X_p be the design matrices and $\beta_\phi$and β_p be the parameter vectors. The design matrices have $n(K$-1) rows and a column for each parameter. Using the logit link, $\phi = [1 + \exp(-X_\phi\beta_\phi)]^{-1}$ and $p = [1 + \exp(-X_p\beta_p)]^{-1}$. The design matrices are easily created using an appropriate data frame and a formula with model.matrix in R. The code for generating the necessary data frame, model structure and running the model has been implemented into an R package called marked (***https://github.com/jlaake/marked***) that is a work in progress. Here I describe the admbcjs.tpl file shown at the end of this document and provide an example.

The DATA_SECTION contains the number of capture histories (n), the number of capture occasions (m), and the capture history matrix (ch(i=1,n;j=1,m) (ω_{ij}) of 0 or 1 values. From the capture history the first occasion (frst f_i) (and last occasion (lst l_i) that each animal was seen is provided for the summation indices. They are computed in the function process.ch in marked. Next a vector of frequencies (frq r_i) for each capture history and a 0/1 indicator (loc) whether the animal was not released after the last occasion (i.e., loss on capture). Following is tint(i=1,n;j=1,m-1), a matrix of time intervals between occasions used to scale survival to standard unit. The time interval can differ for each animal, although normally they will be constant for animals but may differ for occasions. The remainder of the section is "data"

about the model structure. It includes the design matrices for p (X_p pdm(i=1,n(m-1);j=1,kp) and ϕ($X_\phi$phidm(i=1,n(m-1);j=1,kphi)) which have a column for each modeled animal-occasion for p and animal-interval for ϕ and a column for each parameter. The final optional input allows specific real values (inverse-logit) of p and ϕ to be fixed. The structure is an index into the design matrix and the fixed real value.

The PARAMETER_SECTION includes the parameter vectors for ϕ(β_ϕ phibeta (j=1,kphi)) and p (β_p pbeta (j=1,kp)). The remainder defines temporary variables that are used in the calculation of the likelihood. Note that the vector phi has m real values with phi(m)=0 instead of m-1 to handle survival to the last occasion in the calculation.

The PROCEDURE_SECTION follows the equations defined above. The complete set of real values for ϕ(phix) and p (px) are calculated at the any fixed values are set. Using those real parameter values the probability of each capture history is computed and the -log-likelihood. If the computed probability of a capture history is 0, it is set to a very small value to avoid log(0) issues.

As an example, I use the well-known dipper data and fit a model with constant survival and time-varying capture probability. I do that with the crm (**c**apture-**r**ecapture **m**odel) function in marked (`https://github.com/jlaake/marked`)which creates the admbcjs.dat file and uses R2admb to run the model and read the results file. The following code and results show the fitted model with marked and with MARK (White and Burnham, 1999)using the RMark interface (Laake and Rexstad, 2008).

```
> library(marked)
> # get dipper data
> data(dipper)
> # process data frame but don't accumulate same capture histories
> dipper.proc=process.data(dipper,model="cjs",begin.time=1)

294 capture histories collapsed into  55

> # create design data
> dipper.ddl=make.design.data(dipper.proc)
> # Fit model
> admb.mod=crm(dipper.proc,dipper.ddl,use.admb=TRUE,hessian=TRUE,
+                       model.parameters=list(Phi=list(formula=~1),
+                                             p=list(formula=~time)))
```

```
Computing initial parameter estimates
Accumulating capture histories based on design. This can take awhile.
55  capture histories collapsed into  32

Elapsed time in minutes:  0.1245

> # Show results
> admb.mod

crm Model Summary

Npar :  7
-2lnL:  664.48
AIC  :  678.48

Beta
                Estimate        se          lcl       ucl
Phi.(Intercept) 0.213164 0.1121135 -0.006578373 0.4329064
p.(Intercept)   1.295540 0.7437289 -0.162168648 2.7532486
p.time3         0.800513 1.1635526 -1.480050107 3.0810761
p.time4         0.651253 1.0018591 -1.312390793 2.6148968
p.time5         0.997713 0.9454529 -0.855374669 2.8508007
p.time6         1.465850 1.0303981 -0.553730259 3.4854303
p.time7         1.990040 3.0641579 -4.015709395 7.9957894

> # Now fit the same model with MARK via RMark
> # detach marked and attach RMark
> detach("package:marked")
> library(RMark)
> # process same dataframe for RMark
> dipper.proc=process.data(dipper,model="CJS",begin.time=1)
> # make design data (different format from marked)
> dipper.ddl=make.design.data(dipper.proc)
> # fit model
> mark.mod=mark(dipper.proc,dipper.ddl,output=FALSE,
+                 model.parameters=list(Phi=list(formula=~1),
+                                       p=list(formula=~time)))
> # Show results
> summary(mark.mod,brief=T)
```

```
Output summary for CJS model
Name : Phi(~1)p(~time)

Npar :  7
-2lnL:  664.4802
AICc :  678.7481

Beta
                  estimate        se        lcl       ucl
Phi:(Intercept) 0.2131640 0.1121137 -0.0065788 0.4329068
p:(Intercept)   1.2955228 0.7437227 -0.1621736 2.7532193
p:time3         0.8005309 1.1635480 -1.4800233 3.0810851
p:time4         0.6512803 1.0018563 -1.3123580 2.6149186
p:time5         0.9977312 0.9454476 -0.8553460 2.8508085
p:time6         1.4658904 1.0303992 -0.5536920 3.4854728
p:time7         1.9900819 3.0642426 -4.0158338 7.9959975

>
```

References

Laake, J. and Rexstad, E. (2008). RMark – an alternative approach to building linear models in MARK. In Cooch, E. and White, G. C., editors, *Program MARK: A Gentle Introduction*.

Pledger, S., Pollock, K. H., and Norris, J. L. (2003). Open capture-recapture models with heterogeneity: I. Cormack-Jolly-Seber model. *Biometrics*, 59(4):786–794.

White, G. C. and Burnham, K. P. (1999). Program MARK: survival estimation from populations of marked animals. *Bird Study*, 46:120–139.

```
// Cormack-Jolly-Seber model; fixed effects only for survival (Phi) and capture probability (p)
// Capture history can represent more than one animal; frq is the frequency for the capture history
// Jeff Laake; 15 Nov 2012
DATA_SECTION
  init_int n;                          // number of capture histories
  init_int m;                          // number of occasions
  init_imatrix ch(1,n,1,m);            // capture history matrix
  init_ivector frst(1,n);              // occasion first seen for each history
  init_ivector lst(1,n);               // occasion last seen for each history
  init_vector frq(1,n);                // frequency of each history
  init_ivector loc(1,n);               // 0 or 1, 1 if lost on capture at last event
  init_matrix tint(1,n,1,m-1);         // time interval between occasions for each history-interval
  init_int kphi;                       // number of columns in the design matrix for Phi - survival
  int nrows;                           // number of entries in design matrix m-1 values for each of n histories
  !! nrows=n*(m-1);
  init_matrix phidm(1,nrows,1,kphi); // design matrix for Phi
  init_int kp;                         // number of columns in the design matrix for p - capture probability
  init_matrix pdm(1,nrows,1,kp);       // design matrix for p
  init_int K;                          // number of fixed Phi values
  init_matrix PhiF(1,K,1,2);           // Phi fixed matrix with index in first column and value in second column
  init_int L;                          // number of fixed p values
  init_matrix pF(1,L,1,2);             // p fixed matrix with index in first column and value in second column
PARAMETER_SECTION
  init_vector phibeta(1,kphi);         // parameter vector for Phi
  init_vector pbeta(1,kp);             // parameter vector for p
  vector phi(1,m);                     // temp vector for Phis for each occasion for a single history
  vector p(1,m-1);                     // temp vector for Phis for each occasion for a single history
  number pch;                          // probability of capture history
  vector phicumprod(1,m);              // cummulative survival probability across occasions
  vector cump(1,m);                    // cummulative probability of being seen across occasions
  vector phix(1,nrows);                // vector of all real survival values
  vector px(1,nrows);                  // vector of all real capture probability values
  objective_function_value f;          // objective function - negative log-likelihood
PROCEDURE_SECTION
  int i,i1,i2,j;                                          // miscellaneous ints
  f=0.0;                                                  // initialize to 0
  phix=1/(1+exp(-phidm*phibeta));                         // compute all phi values using inverse logit
  for(i=1;i<=K;i++)                                       // assign any fixed real phi values
    phix(PhiF(i,1))=PhiF(i,2);
  px=1/(1+exp(-pdm*pbeta));                               // compute all p values using inverse logit
  for(i=1;i<=L;i++)                                       // assign any fixed real p values
    px(pF(i,1))=pF(i,2);
  phi=0;                                                  // set all phi values to 0
  for(i=1;i<=n;i++)                                       // loop over each history
  {
    phicumprod=1.0;                                       // set cummulative survival to 1
    cump=1.0;                                             // set cummulative capture prob to 1
    i1=(m-1)*(i-1);                                       // compute beginning index in design matrix
    for(j=frst(i)+1;j<=m;j++)                             // loop over occasions from frst to m
    {
      i2=i1+j;                                            // compute index in design matrix for this occasion
      phi(j-1)=pow(phix(i2-1),tint(i,j-1));               // get phi for the interval adjusted for time length
      p(j-1)=px(i2-1);                                    // get p for the occasion
      phicumprod(j)=phicumprod(j-1)*phi(j-1);             // compute cummulative survival
      cump(j)=cump(j-1)*((1-p(j-1))*
            (1-ch(i,j))+p(j-1)*ch(i,j));                  // compute cummulative capture probability
    }
    pch=0.0;                                              // initialize capture history probability
    for(j=lst(i);j<=((1-loc(i))*m+loc(i)*lst(i));j++)     // loop over last occasion to m unless loss on capture
    {                                                     //  to compute prob of the capture history
```

```
        i2=i1+j;                                        // index occasion
        if(loc(i)==1)
           pch=pch+cump(j)*phicumprod(j);               // probability of history for loss on captures
        else
           pch=pch+cump(j)*phicumprod(j)*(1-phi(j));    // probability of history given possible last occasion alive
     }
     if(pch < 1E-15 & frq(i)> 0) pch=1E-307;            // avoid log(0)
     f-=frq(i)*log(pch);                                // sum -log-likelihood frequency*log(pr(ch))
}
```