Adjustment for varying effort in **secr**

Murray Efford

2014-11-17

Contents

When sampling effort varies between detectors or over time in a capture–recapture study we expect a commensurate change in the number of detections. Allowing for known variation in effort when modelling detections has these benefits: - detection parameters are related to a consistent unit of effort (e.g., one trap set for one day) - the fit of the detection model is improved - trends in the estimates may be modelled without confounding.

Borchers and Efford (2008) allowed the duration of exposure to vary between sampling occasions in their competing-hazard model for multi-catch traps. Efford, Borchers and Mowat (2013) generalised the method to allow joint variation in effort over detectors and over time (occasions), and considered other detector types. This document reiterates some of their material and describes the implementation in **secr**.

Background

We use T_{sk} for the effort on occasion s at detector k. It is expected that for small T_{sk} the number of detections increases linearly with T_{sk} (saturation may occur at higher effort, depending on the detector type) and that there are no detections when $T_{sk} = 0$. Examples of possible effort variables are the number of days that each automatic camera was operated in a tiger study, or the number of rub trees sampled for DNA in each grid cell of a grizzly bear study.

The observations to be modelled are either binary (represented by δ_{sk}, an indicator variable for the presence of an animal on occasion s at binary detector k), or integer (represented by y_{sk}, the number of detections on occasion s at count detector k). We assume the probability of detecting an individual declines with the distance $d_k(X)$ between a detector k and the animal's range centre at coordinates $X = (x, y)$. The binary relationship is described by a spatial detection function $g(d_k(X); \theta)$, where θ is a vector of parameters. We define $g(.)$ so that its intercept when $d_k = 0$ is a non-spatial scale parameter g_0 $(0 < g_0 \leq 1)$. For a concrete example, the half-normal detection function uses $g(d_k(X); g_0, \sigma) = g_0 \exp(-d_k(X)^2/(2\sigma^2))$.

If the data are counts rather than binary observations, we may choose to define the spatial detection function as the decline in expected count with distance $\lambda(d_k(X); \theta')$. We use the symbol λ_0 for the intercept $(\lambda_0 > 0)$. For a particular distribution of the counts we can switch back and forth between the binary and expected-count representations (e.g., $g(X) = 1 - e^{-\lambda(X)}$ when the counts are Poisson-distributed). The transformation is non-linear so, for example, a half-normal form for $g(.)$ does not correspond to half-normal form for $\lambda(.)$.

Previous methods

Two options were provided for effort adjustment in earlier versions of **secr**:

1. If only a subset of detectors is used on any occasion s, and there is no other variation in effort, T_{sk} is a binary indicator taking the values 0 (detector not used) or 1 (detector used). This case is handled simply by setting log-likelihood components for occasion s and detector k to 0 whenever $T_{sk} = 0$ (via the 'usage' attribute of 'traps' objects in **secr**).
2. The parameter g_0 or λ_0 may be modelled on an appropriate link scale (logit or log) as a linear function of T_{sk} or other time-varying detector-level covariates.

The first is effective for binary use vs non-use of detectors, but does not encompass other gradations of effort. The second is suboptimal because varying effort is not expected to have a linear additive effect on either of the default link scales, and the estimation of additional parameters is an unnecessary burden.

Linear hazard models

A more comprehensive approach to effort adjustment follows from the hazard model of Borchers and Efford (2008). We assume detections are independent of each other except as allowed by the competing hazard model for multi-catch traps. The variables to be modelled are δ_{sk}, an indicator variable for the presence of an animal on occasion s at binary detector k, and y_{sk}, the number of detections on occasion s at detector k, if k is a 'count' detector (i.e., one that can record multiple independent occurrences of an animal).

The properties of various detector types and the expressions for detection as a function of effort T_{sk} are given in Table 1. Only in the Poisson case is the expected number of detections linear on effort. For binomial count detectors we use a formulation not based directly on instantaneous hazard, as explained in Efford, Borchers and Mowat (2013).

Table 1. Including effort in SECR models for various detector types. δ_{sk} is an indicator variable for detection of an individual on occasion s at detector k, and y_{sk} is a count of detections of an individual on occasion s at detector k.

Detector type	Model	Detector- and occasion-specific detection
Multi-catch trap	$\delta_{sk} \sim \text{Bernoulli}(p_{sk})$	$p_{sk}(X) = [1 - e^{-h_{\cdot s}(X)}] h_{sk}(X)/h_{\cdot s}(X)$
Binary proximity	$\delta_{sk} \sim \text{Bernoulli}(p_{sk})$	$p_{sk}(X) = 1 - [1 - g(d_k(X))]^{T_{sk}}$
Poisson count	$y_{sk} \sim \text{Poisson}(\lambda_{sk})$	$\lambda_{sk}(X) = \lambda_0 T_{sk} g(d_k(X))$
Binomial count	$y_{sk} \sim \text{Binomial}(N_{sk}, p_{sk})$	$N_{sk} = T_{sk}$, $p_{sk} = g(d_k(X))$

Implementation in secr

The 'usage' attribute of a 'traps' object in **secr** is a K x S matrix recording the effort (T_{sk}) at each detector k ($k = 1...K$) and occasion s ($s = 1...S$). If the attribute is missing (NULL) it will be treated as all ones. Extraction and replacement functions are provided (`usage` and `usage<-`, as demonstrated below). All detector types accept usage data in the same format, with the exceptions in Table 2.

Table 2. Deviations from standard treatment of sampling effort.

Detector	Usage
polygon	usage matrix has one row for each polygon
transect	usage matrix has one row for each transect
signal strength	usage is not considered when fitting acoustic models
binomial counts	N_{sk} is determined by `secr.fit` from usage, rounded to an integer, when "binomN = 1", or equivalently "binomN = 'usage' "

Data entry Usage data may be input as extra columns in the text file of detector coordinates (see `?read.traps` and secr-datainput.pdf). When only binary (0/1) codes are used, and the `read.traps` argument 'binary.usage = TRUE', separation with white space is optional. This means that '01000' and '0 1 0 0 0' are equivalent. For non-binary values always set 'binary.usage = FALSE' and separate with spaces.

The input file for polygons and transects has multiple rows for each unit (one row for each vertex). Usage data are taken from the first vertex for each polygon or transect.

Usage codes may be added to an existing traps object, even after it has been included in a capthist object. For example, the traps object in the demonstration dataset 'captdata' starts with no usage attribute:

```r
library(secr)
usage(traps(captdata))
```

```
## NULL
```

Suppose that we knew that traps 14 and 15 caught no animals on occasions 1–3 because they were not set. We could construct and assign a binary usage matrix to indicate this:

```r
mat <- matrix(1, nrow = 100, ncol = 5)
mat[14:15,1:3] <- 0
usage(traps(captdata)) <- mat
```

Model fitting Following on from the preceding example, we can confirm our assignment and fit a new model.

```r
summary(traps(captdata))
```

```
## Object class      traps
## Detector type     single
## Detector number   100
## Average spacing   30 m
## x-range           365 635 m
## y-range           365 635 m
## Usage range by occasion
##     1 2 3 4 5
## min 0 0 0 1 1
## max 1 1 1 1 1
```

```
mask <- make.mask(traps(captdata), buffer = 100, type = 'trapbuffer')
```

```
fit <- secr.fit(captdata, mask = mask, trace = FALSE)
```

```
predict(fit)
```

```
##        link estimate SE.estimate     lcl     ucl
## D       log   5.4675     0.64522  4.3420  6.8848
## g0    logit   0.2767     0.02735  0.2264  0.3333
## sigma   log  29.3938     1.30807 26.9398 32.0713
```

The result in this case is only subtly different from the model with uniform usage (compare `predict(secrdemo.0)`). [We pre-define the mask to avoid a warning message from `secr.fit` regarding `bias.D`: this function is usually run by `secr.fit` after any model fit using the 'buffer' argument, but it does not handle varying effort.]

Usage is hardwired into the traps object, and will be applied (in the sense of Table 1) when a model is fitted with `secr.fit`. There are two ways to suppress this. The first is to remove or replace the usage attribute. For example,

```
usage(traps(captdata)) <- NULL
```

returns our demonstration dataset to its original state (this would happen in any case when we started a new R session). The second is to bypass the attribute for a single model fit by calling `secr.fit` with 'details = list(ignoreusage = TRUE)'.

For a more interesting example, we simulate data from an array of proximity detectors (such as automatic cameras) operated over 5 occasions, using the default density (5/ha) and detection parameters (g0 = 0.2, sigma = 25 m) of `sim.capthist`. We choose to expose all detectors twice as long on occasions 2 and 3 as on occasion 1, and three times as long on occasions 4 and 5:

```
simgrid <- make.grid(nx = 10, ny = 10, detector = 'proximity')
usage(simgrid) <- matrix(c(1,2,2,3,3), byrow = TRUE, nrow = 100, ncol = 5)
simCH <- sim.capthist(simgrid, popn = list(D = 5, buffer = 100), detectpar =
                        list(g0 = 0.1, sigma = 25), noccasions = 5)
summary(simCH)
```

```
## Object class      capthist
## Detector type     proximity
## Detector number   100
## Average spacing   20 m
## x-range           0 180 m
## y-range           0 180 m
## Usage range by occasion
##     1 2 3 4 5
## min 1 2 2 3 3
## max 1 2 2 3 3
## Counts by occasion
##                     1   2   3   4   5 Total
## n                  14  22  17  26  23   102
## u                  14  14   2   5   2    37
## f                  10   4  12   7   4    37
## M(t+1)              14  28  30  35  37    37
## losses               0   0   0   0   0     0
## detections          20  39  27  55  46   187
## detectors visited   19  35  23  44  37   158
## detectors used     100 100 100 100 100   500
```

Now we fit three models with a half-normal detection function. The first model (fit.usage) implicitly adjusts for effort. The second (fit.null) has no adjustment because we wipe the usage information. The third (fit.t) again ignores effort, but allows for occasion-to-occasion variation by fitting a separate g0 each time. We use 'trace = FALSE' to suppress output from each likelihood evaluation.

```
mask <- make.mask(simgrid, buffer = 100, type = 'trapbuffer')
```

```
fit.usage <- secr.fit(simCH, mask = mask, trace = FALSE)
usage(traps(simCH)) <- NULL
fit.null <- secr.fit(simCH, mask = mask, trace = FALSE)
fit.t <- secr.fit(simCH, model = g0 ~ t, mask = mask, trace = FALSE)
```

```
AIC(fit.usage, fit.null, fit.t)
```

```
##                          model   detectfn npar logLik  AIC AICc  dAICc AICcwt
## fit.usage D~1 g0~1 sigma~1 halfnormal    3 -547.3 1101 1102  0.000 0.9827
## fit.t     D~1 g0~t sigma~1 halfnormal    7 -544.9 1104 1110  8.074 0.0173
## fit.null  D~1 g0~1 sigma~1 halfnormal    3 -566.1 1138 1139 37.666 0.0000
```

From the likelihoods we can see that failure to allow for effort (model fit.null) dramatically reduces model fit. The fully time-varying model (fit.t) captures the variation in detection probability, but at the cost of fitting $S - 1$ additional parameters. The model with built-in adjustment for effort (fit.usage) has the lowest AIC, but how do the estimates compare? This is a task for the **secr** function `collate`.

```
collate(fit.usage, fit.null, fit.t, newdata = data.frame(t =
    factor(1:5)))[,,'estimate','g0']
```

```
##      secr1  secr2   secr3
## t=1 0.1261 0.2549 0.09413
## t=2 0.1261 0.2549 0.20156
## t=3 0.1261 0.2549 0.20016
## t=4 0.1261 0.2549 0.41318
## t=5 0.1261 0.2549 0.37398
```

The null model fits a single 'average' g0 across all occasions that is approximately twice the true rate on occasion 1 (0.2). The estimates of g0 from fit.t mirror the variation in effort. The effort-adjusted model estimates the fundamental rate for one unit of effort (0.2).

```
collate(fit.usage,fit.null,fit.t)[,,,'D']
```

```
##       estimate SE.estimate   lcl   ucl
## secr1    3.837      0.7552 2.618 5.622
## secr2    3.834      0.7550 2.616 5.619
## secr3    3.829      0.7537 2.613 5.611
```

The density estimates themselves are almost entirely unaffected by the choice of model for g0. This is not unusual. Nevertheless, the example shows how unbalanced data may be analysed with a minimum of fuss.

Adjustment for varying usage will be more critical in analyses where (i) the variation is confounded with temporal (between-session) or spatial variation in density, and (ii) it is important to estimate the temporal or spatial pattern. For example, if detector usage was consistently high in one part of a landscape, while true density was constant, failure to allow for varying usage might produce a spurious density pattern.

Data manipulation and checking The various functions in **secr** for manipulating traps and capthist objects (`subset`, `split.traps`, `rbind.capthist`, `MS.capthist`, `join` etc.) attempt to deal with usage intelligently.

When occasions are collapsed or detectors are lumped with the `reduce` method for capthist objects, usage is summed for each aggregated unit.

The function `usagePlot` displays a bubble plot of spatially varying detector usage on one occasion. The arguments 'markused' and 'markvarying' of `plot.traps` may also be useful.

Polygons and transects Binary or count data from searches of polygons or transects (Efford 2011) do not raise any new issues for including effort, at least when effort is homogeneous across each polygon or transect. Effects of varying polygon or transect size are automatically accommodated in the models of Efford (2011). Models for varying effort within polygons or transects have not been needed for problems encountered to date. Such variation might in any case be accommodated by splitting the searched areas or transects into smaller units that were more nearly homogeneous (see the `snip` function for splitting transects).

Miscellaneous

The units of usage determine the units of g_0 or λ_0 in the fitted model. This must be considered when choosing starting values for likelihood maximisation. Ordinarily one relies on `secr.fit` to determine starting values automatically (via `autoini`), and a simple linear adjustment for usage, averaged across non-zero detectors and occasions, is applied to the value of g0 from `autoini`.

Usage values other than 0 and 1 require significant additional computation because the adjustment is re-computed for each combination of detector x occasion x mask point x detection history x finite mixture. Execution speed may be improved in future versions.

Absolute duration does not always equate with effort. Animal activity may be concentrated in part of the day, or older DNA samples from hair snares may fail to amplify (Efford et al. 2013).

References

Borchers, D. L. and Efford, M. G. (2008) Spatially explicit maximum likelihood methods for capture–recapture studies. *Biometrics* **64**, 377–385.

Efford, M. G. (2011) Estimation of population density by spatially explicit capture–recapture analysis of data from area searches. *Ecology* **92**, 2202–2207.

Efford, M. G., Borchers D. L. and Mowat, G. (2013) Varying effort in capture–recapture studies. *Methods in Ecology and Evolution* **4**, 629–636.