Expected sample size from spatially explicit capture–recapture designs

Murray Efford

2017-10-02

Contents

Introduction

Spatially explicit capture–recapture (SECR) combines a model for the distribution of animal home range centres in 2 dimensions and a model for observations of animals, usually at fixed points (detector locations). The number of individuals detected n and the total number of detections r are useful summaries of the sample size from an SECR study. This note gives formulae for the expectation of these counts given a particular sampling design and parameter values.

SECR model and parameters

Animal home range centres are assumed to follow an inhomogeneous Poisson distribution with intensity $D(\mathbf{x})$ at point $\mathbf{x}$. It is often good enough to assume D is flat (does not vary with $\mathbf{x}$), but the formulae accommodate modelled variation in D. The Poisson model assumes individuals are located independently of each other.

We assume detections have been made at K detectors on S occasions, and that the hazard of detection in detector k for an animal centred at $\mathbf{x}$ depends on the distance $d_k(\mathbf{x})$. The precise form of the relationship is not critical for our calculations - a halfnormal relationship is commonly assumed i.e., $\lambda(d_k(\mathbf{x});\theta) = \lambda_0 \exp[-d_k(\mathbf{x})^2/(2\sigma^2)]$ where θ is the parameter vector (λ_0, σ). It is convenient to formulate the detection process in terms of hazard $\lambda(d_k(\mathbf{x}))$ rather than probability $g(d_k(\mathbf{x}))$, but the two are interchangeable $(\lambda(d_k(\mathbf{x})) = -\log[1 - g(d_k(\mathbf{x}))])$.

We define the quantity $\Lambda_s(\mathbf{x})$ as $\Lambda_s(\mathbf{x}) = \sum_K \lambda(d_k(\mathbf{x}))$.

If all potential detections are recorded then $\Lambda_s(\mathbf{x})$ is the expected total number of detections on one occasion for an animal centred at $\mathbf{x}$. Only Poisson 'count' detectors are assumed to act like this. Other detector types collect binary data (e.g. 'proximity' detectors record only whether an individual appeared at least once or not at all at a detector on a certain occasion). Nevertheless, $\Lambda_s(\mathbf{x})$ is useful for predicting the outcome for binary detector types as shown later. Single-catch traps are a special case for which there are not closed-form expressions for $\mathrm{E}(n)$ and $\mathrm{E}(r)$.

Aggregating over occasions gives $\Lambda(\mathbf{x}) = \sum_s \Lambda_s(\mathbf{x})$.

Number of individuals n

The expected number of individuals detected at least once is

$$E(n) = \int [1 - \exp\{-\Lambda(\mathbf{x})\}] \times D(\mathbf{x})\, d\mathbf{x}.$$

This is the same for all detector types in which individuals are detected independently of each other ('multi', 'proximity' or 'count'). Integration is over all locations in the plane from which an individual might be detected. The region of integration is represented in practice by a discretized 'habitat mask', and integration is performed by summing over cells.

Number of detections C

The total number of detections C depends on the detector type, as follows.

Detector type 'count'

This is the simplest case –

$$E(C) = \int \Lambda(\mathbf{x}) \times D(\mathbf{x})\, d\mathbf{x}.$$

Detector type 'multi'

Data from 'multi' detectors are binary at the level of each animal × occasion, with Bernoulli probability $p_s = 1 - \exp\{-\Lambda_s(\mathbf{x})\}$. This leads to the overall number of detections –

$$E(C) = \int \sum_s p_s(\mathbf{x}) \times D(\mathbf{x})\, d\mathbf{x}.$$

Detector type 'proximity'

The data from 'proximity' detectors are binary at the level of each animal × detector × occasion, with Bernoulli probability $p_{ks}(\mathbf{x}) = 1 - \exp\{-\lambda(d_k(\mathbf{x}))\}$. This leads to the overall number of detections –

$$E(C) = \int \sum_s \sum_k p_{ks}(\mathbf{x}) \times D(\mathbf{x})\, d\mathbf{x}.$$

Number of recaptures r

For all detector types the expected number of recaptures is simply

$$E(r) = E(C) - E(n).$$

Number of movements m

A movement is a recapture (redetection) at a site other than the previous one. Movements are a subset of recaptures. We calculate the expected number of movements by considering each recapture event in turn and calculating the conditional probability that it is at the same site as before. This is a sum of squared detector-wise conditional probabilities.

Conditional on detection somewhere, the probability of detection in detector k is $q_k(\mathbf{x}) = \lambda(d_k(\mathbf{x}))/\sum_k \lambda(d_k(\mathbf{x}))$. For clarity in the following detector-specific expressions we use $a(\mathbf{x}) = 1 - \exp\{-\Lambda(\mathbf{x})\})$ and $b(\mathbf{x}) = 1 - \sum_k q_k(\mathbf{x})^2$.

Detector type 'count'

$$E(m) = \int \{\Lambda(\mathbf{x}) - a(\mathbf{x})\} \times b(\mathbf{x}) \times D(\mathbf{x})\, d\mathbf{x}.$$

Detector type 'multi'

$$E(m) = \int \{\sum_s p_s(\mathbf{x}) - a(\mathbf{x})\} \times b(\mathbf{x}) \times D(\mathbf{x})\, d\mathbf{x}.$$

Detector type 'proximity'

$$E(m) = \int \{\sum_s \sum_k p_{ks}(\mathbf{x}) - a(\mathbf{x})\} \times b(\mathbf{x}) \times D(\mathbf{x})\, d\mathbf{x}.$$

Caveat

If an animal may be detected more than once on one occasion (as with 'proximity' and 'count' detector types) and time of detection is not recorded within each occasion (the norm in **secr**) then the temporal sequence of detections is not fully observed. The number of observed (apparent) movements is then less than or equal to the true number. Results from the `moves` function in **secr** are also not to be trusted: they effectively assume any repeat detections at the same site precede other redetections rather than being interspersed in time. Precise formulae are not available for the expected number of observed movements among proximity and

count detectors. There should be little discrepancy between observed and true numbers when detections are sparse. The predicted number of movements is close to the apparent number in simulations (see later section; this deserves further investigation).

Single-catch traps

All the preceding calculations assume independence among animals. If traps can catch only one animal at a time then animals effectively compete for access (the first arrival is most likely to be caught). This depresses the realised hazard of detection $\lambda(d_k(\mathbf{x}); \theta)$; the effect increases with density. No closed-form expressions exist for this case. The computed E(n), E(r) and E(m) for multi-catch traps (detector 'multi') will exceed the true values for the single-catch traps (detector 'single') *given the same detection parameters*. That final caveat is significant because a pilot value of $\hat{\lambda}_0$ from fitting a multi-catch model to single-catch data will be an underestimate[1].

Functions in secrdesign

The **secrdesign** functions `Lambda` and `Enrm` implement the preceding calculations. `Enrm` is in turn used by `scenarioSummary` (see secrdesign-vignette.pdf), `minnrRSE` and `optimalSpacing` (see secrdesign-tools.pdf).

Tests

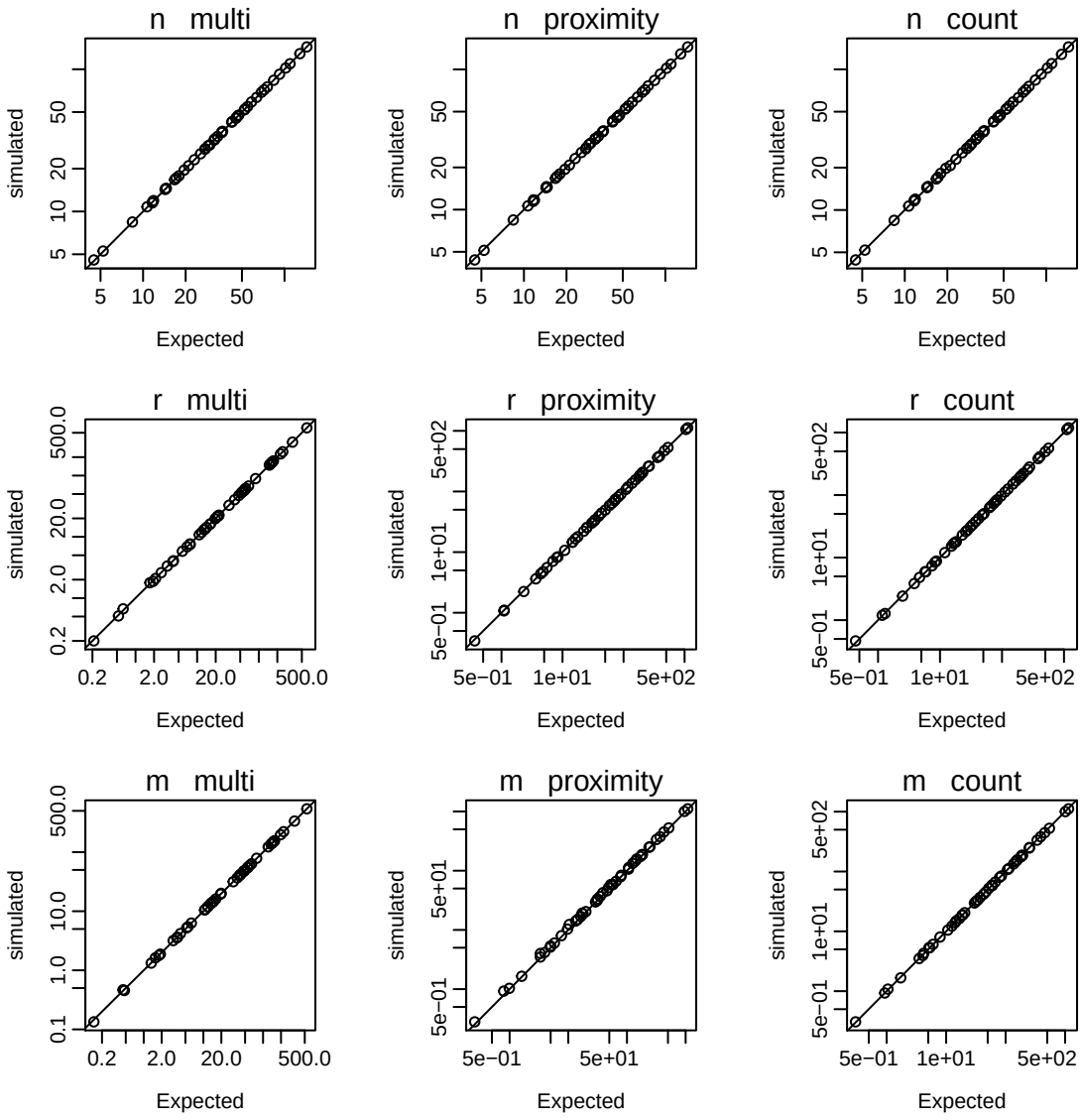
This code can be used to compare expected counts from `Enrm` with averages from stochastic simulations. Agreement is good, which confirms both the formulae and the reliability of the simulations. Using 5 cores for parallel processing the run time for 1000 replicates is typically less than 30 minutes.

```
library(secrdesign)
# generate scenario dataframe
scen <- make.scenarios (trapsindex = 1:6, detectfn = 'HHN', D = c(5,20),
                        lambda0 = c(0.05,0.2), sigma = 1:2, noccasions = c(2, 5, 10))
# set sigma = k/sqrt(D) for k = 0.5, 1
scen$sigma <- c(50,100)[scen$sigma] / sqrt(scen$D)
# detector layouts
traplist <- list(
    make.grid(6,6, detector = 'multi', spacing = 20),
    make.grid(6,6, detector = 'proximity', spacing = 20),
    make.grid(6,6, detector = 'count', spacing = 20),
    make.grid(10,10, detector = 'multi', spacing = 20),
    make.grid(10,10, detector = 'proximity', spacing = 20),
    make.grid(10,10, detector = 'count', spacing = 20)
)
# deterministic summary: expected counts
nrm <- scenarioSummary(scen, traplist)
# stochastic simulations
sumnrm <- function(CH)
    c(n = nrow(CH), r = sum(CH) - nrow(CH),
      m = sum(unlist(secr::moves(CH))>0, na.rm = TRUE))
```

[1] Efford, M. G., Borchers D. L. and Byrom, A. E. (2009) Density estimation by spatially explicit capture-recapture: likelihood-based methods. In: D. L. Thomson, E. G. Cooch and M. J. Conroy (eds) *Modeling Demographic Processes in Marked Populations*. Springer, New York. Pp. 255–269.

```
sim <- run.scenarios(scen, trapset = traplist, nrepl = 1000,
                     extractfn = sumnrm, fit = FALSE, ncores = 5)
meanse <- function(xmat)
    c(nrow(xmat), apply(xmat,2,mean), apply(xmat,2,sd) / sqrt(nrow(xmat)))
simout <- round(t(sapply(sim$output, meanse)),4)
dimnames(simout)[[2]] <- c('nrepl', 'n','r','m', 'sen','ser','sem')
# collate deterministic and stochastic results
out <- data.frame(nrm, simout)
```

```
plotc <- function (v = 'n') {
    dtype <- c('multi','proximity','count')
    detector <- rep(dtype,2)[out$trapsindex]
    out2 <- matrix(nrow=3, ncol = 2, dimnames = list(dtype, c('mean','sd')))
    for (d in 1:3) {
        OK <- (detector == dtype[d])
        RB <- (out[OK,v] - out[OK, paste0('E',v)]) / out[OK, paste0('E', v)]
        # use log scales to spread values
        plot(out[OK, paste0('E',v)], out[OK,v], log='xy',
             xlab = 'Expected', ylab = 'simulated')
        abline(0,1) # y = x line
        # return mean and sd of estimated relative bias
        out2[d,] <- round(c(mean = mean(RB), sd = sd(RB)),5)
        mtext (side=3, line=0.2, paste(v, " ", dtype[d]), cex = 0.9)
    }
    out2
}
par(mfrow=c(3,3), mgp=c(2.3,0.6,0), mar=c(4,4,2,1), pty='s')
cat("Relative discrepancy between expected and simulated counts\n")
cat("Number of individuals\n")
plotc('n')
cat("Number of recaptures\n")
plotc('r')
cat("Number of movements\n")
plotc('m')
```




```
## Relative discrepancy between expected and simulated counts
## Number of individuals
##                mean      sd
## multi       0.00087 0.00513
## proximity -0.00112 0.00726
## count       0.00033 0.00594
## Number of recaptures
##                mean      sd
## multi      -0.00161 0.01646
## proximity -0.00230 0.01381
## count       0.00250 0.01518
## Number of movements
##                mean      sd
## multi      -0.00150 0.01724
## proximity  0.05273 0.04276
## count      -0.00659 0.01660
```