secr - spatially explicit capture–recapture in R

Murray Efford

February 28, 2010

This document provides an overview of **secr**, an R package for spatially explicit capture–recapture analysis (SECR). It includes some necessary background on SECR, an outline of the package, and a more detailed description of how models are implemented. See Appendix 1 for a glimpse of **secr** in action. For details of how to use **secr** see the help pages.

Contents

Introduction to SECR

SECR is a set of methods for estimating the density of an animal population from capture–recapture data collected with an array of 'detectors'. SECR methods overcome edge effects that are problematic in conventional capture–recapture estimation of animal populations (Otis et al. 1978). Detectors may be live-capture traps, with animals uniquely marked. Detectors also may be sticky traps or snags that passively sample hair, from which individuals are distinguished by their DNA microsatellites, or cameras that take photographs from which individuals are recognized by their natural marks.

The primary data for SECR are (i) the locations of the detectors, and (ii) detections of known individuals on one or more sampling occasions (i.e. their detection histories). The terms 'detectors' and 'detections' cover the full spectrum of possibilities (see 'Detector types' below), but we use them interchangeably with the more familiar 'traps' and 'captures', respectively. Table 1 gives a concrete example.

Table 1: Example of spatially explicit detection data. Each entry (e.g. A9) records the detector at which a known animal (ID) was observed on the given occasion (sample time). '.' indicates no detection. Each detector has known x-y coordinates.

	Occasions				
ID	1	2	3	4	5
1	A9	.	.	.	.
2	A12	A12	.	.	.
3	.	.	C6	B5	.
4	.	.	G3	.	F3
etc.					

In SECR, a spatial model of the population and the detection process is fitted to the spatial detection histories. The resulting estimates of population density are unbiased by edge effects and incomplete detection (other sources of bias may remain). Inverse prediction (IP SECR) and maximum likelihood (ML SECR) are alternative methods for fitting the spatial detection model (Efford 2004, Borchers and Efford 2008). Of these, ML SECR is the more flexible, with a caveat for data from single-catch traps. Data augmentation and Markov chain Monte Carlo (McMC) methods have also been used for SECR (Royle et al. 2009), but this approach is not considered here.

State and observation models

Like other methods for estimating animal abundance (Borchers et al. 2002), SECR combines a state model and an observation model. The state model describes the distribution of animal home ranges in the landscape, and the observation model (a spatial detection model) relates the probability of detecting an individual at a particular detector to the distance of the detector from a central point in each animal's home range. The distances are not observed directly (we don't know the range centre), so conventional distance sampling methods do not apply.

Distribution of home-range centres

The distribution of range centres in the population (Borchers and Efford 2008) will usually be treated as a homogeneous Poisson point process (Fig. 1). Density is the sole parameter of a Poisson process. An inhomogeneous distribution may also be fitted and this provides a means to evaluate the effects of habitat variables on density.

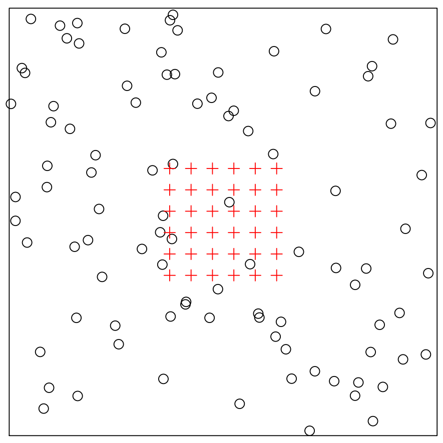

Figure 1: Hypothetical Poisson distribution of range centres near an array of detectors.

Detection functions

A detection model is based on one of several possible parametric forms for the decline in detection probability with distance (d) from the home-range centre (Table 2, Fig. 2). The probability $g(d)$ is for the 'ideal' case of just one animal and one detector; the actual probability may differ (see discussion of traps under Detector Types).

Table 2: Various functions relating the probability of detection to distance (d)

Halfnormal	$g(d) = g_0 \exp\left(\frac{-d^2}{2\sigma^2}\right)$
Exponential	$g(d) = g_0 \exp\left(-\frac{d}{\sigma}\right)$
Hazard-rate	$g(d) = g_0[1 - \exp\{-(\frac{d}{\sigma})^{-z}\}]$

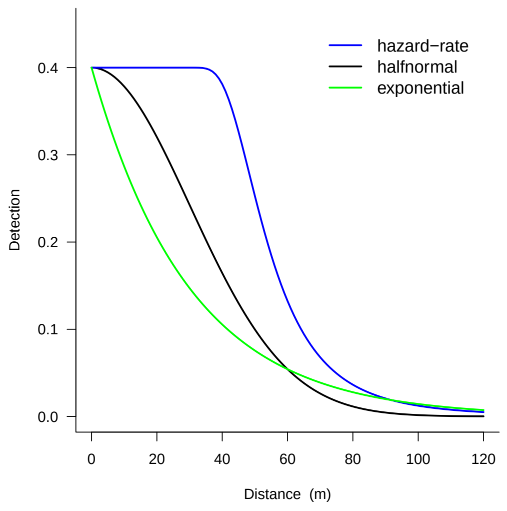

Figure 2: Alternative shapes for a function relating the probability of detection to distance from range centre.

Detector types

Properties of the detectors are an important part of the observation model. Some common detectors (camera 'traps' and hair snags for DNA) do not capture animals but provide evidence of their passing, and can be considered to act independently of each other. We call these 'proximity' detectors. With proximity detectors, each animal x occasion 'cell' of a detection history potentially contains several positive records. In the simplest case each such observation is a binary vector coding presence or absence at each detector. Recent extensions include 'proximity count' detectors (a vector of counts, one per detector) and acoustic 'signal strength' detectors (a binary vector supplemented by measurements of signal strength, e.g. from an array of microphones).

Detectors that are true traps do not act independently because capture of an animal in one trap prevents capture in another (until the animal is released). Traps expose animals to competing risks of capture. The per-trap probability of capture may be adjusted for the competing risk from other traps by using an additive hazard model (Borchers and Efford 2008). However, if the detectors are traps that catch only one animal at a time then there is a further level of competition – between animals for traps. Multi-catch and single-catch traps therefore represent distinct detector types. No general adjustment has been found for the per-trap probability of capture in the single-catch case (it's an open research question), and there is strictly no known maximum likelihood estimator. However, density estimates using the multi-catch likelihood for single-catch data appear only slightly biased (Efford, Borchers and Byrom 2009).

Origins and outline of the package ‘secr’

The program DENSITY (Efford et al. 2004, Efford 2009) provides a graphical interface to SECR methods that has been accepted by many biologists. However, DENSITY has significant drawbacks: it requires the Windows operating system and is increasingly difficult to maintain, its algorithms are not always transparent or well-documented, it fits only homogeneous Poisson models, and it omits some recent advances in SECR.

The R package **secr** was written to address these weaknesses and allow for further development. It implements almost all the methods and options described in Borchers and Efford (2008) and Efford et al. (2009), and others yet to be published. **secr** uses external C code for computationally intensive operations. Appendix 2 compares the features of DENSITY and **secr**. The major functions of **secr** are listed in Appendix 3.

How secr works

secr defines a set of R classes[1] and methods for data from detector arrays. The essential classes are:

`traps`	locations of detectors; detector type (‘proximity’, ‘multi’, etc.)
`capthist`	spatial detection histories, including a `traps` object
`mask`	points on habitat mask
`secr`	fitted SECR model.

To perform an SECR analysis you will construct each of these objects in turn, using the functions provided (e.g., `make.grid`[2], `make.capthist`, `secr.fit`). Fig. 3 summarizes the relationships among the core object classes. The classes `traps`, `capthist` and `mask` may optionally store covariates specific to detectors, animals and habitat points respectively. Each set of covariates is saved in a dataframe that is an attribute of the corresponding object; the `covariates` method is used to extract or replace covariates.

Output

The output from the function `secr.fit` is an object of class `secr`. This is an R list with many components. Assigning the output to a named object (such as secr0 or secrb in the example) saves both the fit and the data for further manipulation. Typing the name at the R prompt invokes `print.secr` which formats the key results. Functions are provided for further computations on `secr` objects (e.g., density as a derived parameter, profile-likelihood confidence intervals for beta parameters, AIC model selection, model averaging, likelihood ratio and score tests). Some of these are listed in Appendix 3.

[1] A ‘class’ specifies a particular type of data object and the functions (methods) by which it is manipulated (computed, printed, plotted etc). See the R documentation for further explanation

[2] Text in `teletype` font refers to R objects that are documented in online help for the **secr** package, or in base R. A good place to start is the page for `secr.fit`

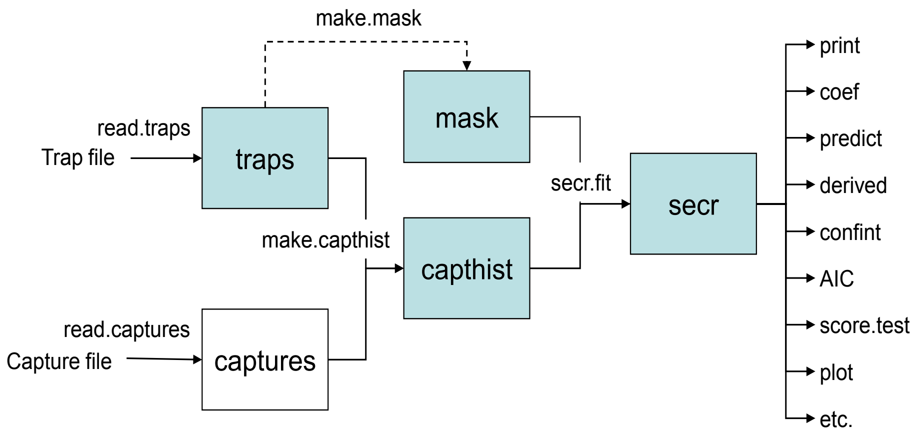

Figure 3: Essentials of the **secr** package. Each object class (shaded box) comes with methods to display and manipulate the data it contains (e.g. `print`, `summary`, `plot`, `rbind`, `subset`). Detector coordinates (`traps`) are stored with attributes such as detector type and usage. Detection data (`captures`) are initially stored in a dataframe with one row per detection, but must be wrapped with the corresponding `traps` in a `capthist` object for analysis. If a habitat mask is not created manually (dashed arrow) it will be generated automatically by `secr.fit`. Any of the objects input to `secr.fit` may include a dataframe of covariates whose names may be used in a model formula. Fitted secr models may be further manipulated with the methods shown on the right. Additional functions (not shown) construct a regular detector array (e.g. `make.grid`, `make.circle`) or simulate detection of a known population (`sim.capthist`).

Documentation

The primary documentation for **secr** is in the help pages that accompany the package. Help for a function is obtained in the usual way by typing a question mark at the R prompt, followed by the function name. Note the 'Index' link at the bottom of each help page.

The consolidated help pages are also distributed as a pdf file that may be accessed from within R using

```
> RShowDoc("secr-manual", package = "secr")
```

Other documentation in the form of pdf files based on Sweave vignettes will be added from time to time. The 'overview' link in the package help index lists available vignettes. To list vignettes for all packages use

```
> vignette()
```

The same function may be used to view a particular vignette, for example

```
> vignette("secr-sound")
```

The web page www.otago.ac.nz/density should be checked for news of bug fixes and new releases.

Models in secr

A family of capture–recapture models, such as the Cormack-Jolly-Seber models for survival, may include submodels[3] that allow for contingent variation in core parameters, including the effects of covariates. Annual survival, for example, may vary with the severity of winter weather, so it often makes sense to include a measure of winter severity as a covariate. Gary White's MARK software has been particularly successful in packaging open-population models for biologists, and **secr** aims for similar flexibility.

The language of generalised linear models is convenient for describing submodels (e.g. Huggins 1989, Lebreton et al. 1992). Each parameter is treated as a linear combination of effects on its transformed ('link') scale. This is useful for combining effects because, given a suitable link function, any combination maps to a feasible value of the parameter. The logit scale has this property for probabilities in (0, 1), and the natural log scale works for positive parameters i.e. $(0, +\infty)$. These are the link functions used most often in **secr**, but there are others, including the identity (null) link.

Submodels are defined symbolically in **secr** using R formula notation. A separate linear predictor is used for each core parameter. Core parameters are 'real' parameters in the terminology of MARK, and **secr** uses that term to reduce confusion. Four real parameters are commonly modelled in **secr** 1.3; these are denoted D (for density), g0, sigma and z. Only the last three real parameters, which jointly define detection probability as a function of location, can be estimated directly when the model is fitted by maximizing the conditional likelihood (CL = TRUE in `secr.fit`). D is then a derived parameter that is computed from an `secr` object with the function `derived`. 'z' is a shape parameter that is used only when the detection function has the 'hazard-rate' form (Hayes and Buckland 1983).

For each real parameter there is a linear predictor of the form $\mathbf{y} = \mathbf{X}\beta$, where $\mathbf{y}$ is a vector of parameter values on the link scale, $\mathbf{X}$ is a design matrix of predictor values, and β is a vector of coefficients. Each element of $\mathbf{y}$ and corresponding row of $\mathbf{X}$ relates to the value of the real parameter in a particular circumstance (e.g. density at a particular point in space, or detection probability of an animal on a particular occasion). The elements of β are coefficients estimated when we fit the model. In MARK these are called 'beta parameters' to distinguish them from the transformed 'real' parameter values in $\mathbf{y}$. **secr** acknowledges this usage, but also refers to beta and real parameters as 'coefficients' and 'fitted values', a usage more consistent with modelling in R. $\mathbf{X}$ has one column for each element of β. Design matrices are described in more detail in the next section.

Design matrices

A design matrix is specific to a 'real' parameter. Each design matrix $\mathbf{X}$ contains a column of '1's (for the constant or intercept term) and additional columns as

[3]This use of 'submodel' is non-standard – maybe we'll find a better term

needed to describe the effects in the submodel for the parameter. Depending on the model, these may be continuous predictors (e.g. air temperature to predict occasion-to-occasion variation in g0), indicator variables (e.g. 1 if animal i was caught before occasion s, 0 otherwise), or coded factor levels. Within `secr.fit`, each design matrix is constructed automatically from the input data and the model formula in a 2-stage process.

First, a data frame is built containing 'design data' with one column for each variable in the formula. Second, the R function `model.matrix` is used to construct the design matrix. This process is hidden from the user. The design matrix will have at least one more column than the design data; there may be more if the formula includes interactions or factors with more than two levels. For a good description of this general approach see the documentation for RMark (Laake and Rexstad 2008). The necessary design data are either extracted from the inputs or generated automatically as explained in later sections. 'Real' parameters fall into two groups: density (D) and detection (g0, sigma and z). Density and detection parameters are subject to different effects, so they use different design matrices as described in the next three sections.

Detection submodels

For SECR, we want to model the detection of each individual i on occasion s at detector k. Given n observed individuals on S occasions at K detectors, there are therefore nSK detection probabilities of interest. We treat these as elements in a 3-dimensional array. Strictly, we are also interested in the detection probabilities of unobserved individuals, but these are estimated only by extrapolation from those observed so we do not include them in the array.

In a null model, all nSK detection probabilities are assumed to be the same. The conventional sources of variation in capture probability (Otis et al. 1978) appear as variation either in the n dimension ('individual heterogeneity' h), or in the S dimension ('time variation' t), or as a particular interaction in these two dimensions ('behavioural response to capture' b). Combined effects are possible. SECR introduces additional complexity.

Detection probability in SECR is no longer a scalar (even for a particular animal-occasion-detector combination); it is described by a 'detection function'. The detection function may have two parameters (e.g. g0, sigma for a half-normal function), three parameters (e.g. g0, sigma, z for the Hayes and Buckland hazard-rate function), or potentially more. Any of the parameters of the detection function may vary with respect to individual (subscript i), occasion (subscript s) or detector (subscript k).

The full design matrix for each detection submodel has one row for each combination of i, s and k. Allowing a distinct probability for each animal (the n dimension) may seem excessive, and truly individual-specific covariates are feasible only when a model is fitted by maximizing the conditional likelihood (cf Huggins 1989). However, the full nSK array is convenient for coding both group membership (Lebreton et al. 1992, Cooch and White 2008) and experience of capture, even when pure individual-level heterogeneity cannot be modelled.

The programming gets even more complex. Analyses may combine data from several independent samples, dubbed 'sessions'. This adds a fourth dimension of length equal to the number of sessions. When finite mixture models are used for detection parameters there is even a fifth dimension, with the preceding structure being replicated for each mixture class. Fortunately, **secr** handles all this and the user need be concerned only with the model specification, which we describe next.

Specifying effects on detection parameters

Effects on parameters of detection probability are specified with R formulae. The variable names used in formulae are either names for standard effects (Table 3) or the names of user-supplied covariates.

Table 3: Automatically generated predictor variables used in detection models

Variable	Description	Notes
g	group	interaction of the capthist individual covariates listed in argument `groups` of `secr.fit`
t	time factor	one level for each occasion
T	time trend	linear trend over occasions on link scale
b	learned response	step change in parameter after first detection of animal
B	transient response	parameter depends on detection at previous occasion (Markovian response)
session	session factor	one level for each session
h2	2-class mixture	finite mixture model with 2 latent classes

Any name in a formula that is not in Table 3 is assumed to refer to a user-supplied covariate. `secr.fit` looks for user-supplied covariates in data frames embedded in the `capthist` argument or supplied in the `timecov` and `sessioncov` arguments, using the first match (Table 4).

The formula for any detection parameter (g0, sigma, z) may be constant ($\sim$1, the default) or some combination of terms in standard R formula notation (see help(formula)). For example, g0 $\sim$ b + T specifies a model with a learned response and a linear time trend in g0; the effects are additive on the link scale. See Table 5 for other examples.

For other effects, the design matrix for detection parameters may also be provided manually in the argument `dframe` of `secr.fit`. This feature is untested.

Table 4: User-provided covariates used in detection models. The names of columns in the respective dataframes may be used in model formulae

Covariate type	Data source	Notes
Individual	`covariates(capthist)`	conditional likelihood
Time	`timecov` argument	
Detector	`covariates(traps(capthist))`	
Session	`sessioncov` argument	

Table 5: Some examples of the `model` argument in `secr.fit`

Model	Description
`g0 ~ 1`	g0 is constant across animals, occasions and detectors
`g0 ~ b`	learned response affects g0
`list(g0~b, sigma~b)`	learned response affects both g0 and sigma
`g0 ~ h2`	2-class finite mixture for heterogeneity in g0
`g0 ~ b + T`	learned response in g0 combined with trend over occasions
`sigma ~ g`	detection scale sigma differs between groups
`sigma ~ g*T`	group-specific trend in sigma
`D ~ cover`	density varies with 'cover' given in `covariates(mask)`
`list(D~g, g0~g)`	both density and g0 differ between groups
`D ~ session`	session-specific density

Density submodels

The SECR log likelihood is evaluated by summing values at points on a 'habitat mask' (the `mask` argument of `secr.fit`). Each point in a habitat mask represents a grid cell of potentially occupied habitat (their combined area may be almost any shape). The full design matrix for density (D) has one row for each point in the mask. As for the detection submodels, the design matrix has one column for the intercept (constant) term and one for each predictor.

Predictors may be based on Cartesian coordinates (e.g. 'x' for an east-west trend), a continuous habitat variable (e.g. vegetation cover) or a categorical (factor) habitat variable. Predictors must be known for all points in the mask (non-habitat excluded). The variables 'x' and 'y' are the coordinates of the habitat mask and are automatic. Other spatial covariates should be named columns in the `covariates` attribute of the habitat mask.

Model fitting and estimation

Models are fitted in `secr.fit` by numerically maximising the likelihood. The likelihood involves integration over the unknown locations of the animals' range centres. This is achieved in practice by summation over points in the habitat mask, which has some implications for the user. Computation may be slow, especially if there are many points in the mask, and estimates may be sensitive to the particular choice of mask (either explicitly in `make.mask` or implicitly via the `buffer` argument).

The default maximisation algorithm is Newton-Raphson in the function `stats::nlm`. By default, all reported variances, covariances, standard errors and confidence limits are asymptotic and based on a numerical estimate of the information matrix. Use `confint.secr` for profile likelihood intervals and `simulate.secr` for parametric bootstrap intervals (slow).

References

Borchers, D. L., Buckland, S. T. and Zucchini, W. (2002) *Estimating animal abundance: closed populations.* Springer, London.

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

Cooch, E. and White, G. (eds) (2008) *Program MARK: A Gentle Introduction.* 6th edition. Available online at http://www.phidot.org.

Efford, M. G. (2004) Density estimation in live-trapping studies. *Oikos* **106**, 598–610.

Efford, M. G. (2009) *DENSITY 4.4: software for spatially explicit capture–recapture.* Department of Zoology, University of Otago, Dunedin, New Zealand http://www.otago.ac.nz/density.

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, M. J. Conroy (eds) *Modeling Demographic Processes in Marked Populations.* Springer. Pp 255–269.

Efford, M. G., Dawson, D. K. and Borchers, D. L. (2009) Population density estimated from locations of individuals on a passive detector array. *Ecology* **90**, 2676–2682.

Hayes, R. J. and Buckland, S. T. (1983) Radial-distance models for the line-transect method. *Biometrics* **39**, 29–42.

Huggins, R. M. (1989) On the statistical analysis of capture experiments. *Biometrika* **76**, 133–140.

Laake, J. and Rexstad E. (2008) Appendix C. RMark - an alternative approach to building linear models in MARK. In: Cooch, E. and White, G. (eds) *Program MARK: A Gentle Introduction.* 6th edition. Available online at http://www.phidot.org.

Lebreton, J.-D., Burnham, K. P., Clobert, J., and Anderson, D. R. (1992) Modeling survival and testing biological hypotheses using marked animals: a unified approach with case studies. *Ecological Monographs* **62**, 67–118.

Otis, D. L., Burnham, K. P., White, G. C. and Anderson, D. R. (1978) Statistical inference from capture data on closed animal populations. *Wildlife Monographs* **62**.

Royle, J. A., Nichols, J. D., Karanth, K. U. and Gopalaswamy, A. M. (2009). A hierarchical model for estimating density in camera-trap studies. *Journal of Applied Ecology* **46**, 118–127.

Appendix 1. A simple secr analysis

A simple analysis might look like this. We start by loading the package and constructing an object `myCH` that contains both the captures and the trap locations.

```
> library(secr)
> mytraps <- make.grid(nx = 10, ny = 10, spacing = 30,
     originxy = c(365, 365))
> mycapt <- read.captures(file = "capt.txt")
> myCH <- make.capthist(mycapt, mytraps, fmt = "XY")
```

Next we fit two simple models and compare them with AIC. We set `trace = FALSE` to reduce the volume of output, but the default `trace = TRUE` is usually better.

```
> secr0 <- secr.fit(myCH, model = g0 ~ 1, trace = FALSE)
> secrb <- secr.fit(myCH, model = g0 ~ b, trace = FALSE)
> AIC(secr0, secrb)

                 model   detectfn npar    logLik      AIC     AICc
secr0 D~1 g0~1 sigma~1 halfnormal    3 -759.0198 1524.040 1524.373
secrb D~1 g0~b sigma~1 halfnormal    4 -759.0106 1526.021 1526.584
      dAICc  AICwt
secr0 0.000 0.7513
secrb 2.211 0.2487
```

A model with learned trap response (g0∼b) showed no improvement in fit over a null model (g0∼1). In this instance the estimates of density from the two models were also very close (not shown) and we rely on the null model for estimation.

```
> secr0

secr.fit( capthist = myCH, model = g0 ~ 1, trace = FALSE )
secr 1.3.0, 20:08:48 11 Mar 2010

Detector type     multi
Detector number   100
Average spacing   30 m
x-range           365 635 m
y-range           365 635 m
N animals       :  76
N detections    :  235
N occasions     :  5
Mask area       :  22.09 ha

Model           :  D~1 g0~1 sigma~1
Fixed (real)    :  none
Detection fn    :  halfnormal
```

```
Distribution    :  poisson
N parameters    :  3
Log likelihood  :  -759.0198
AIC             :  1524.040
AICc            :  1524.373

Beta parameters (coefficients)
            beta    SE.beta       lcl        ucl
D      1.7008900 0.11763036  1.470339  1.9314413
g0    -0.9786458 0.13625081 -1.245692 -0.7115991
sigma  3.3799591 0.04444661  3.292845  3.4670729

Variance-covariance matrix of beta parameters
                  D            g0         sigma
D      0.0138369005  0.0001609861 -0.0009948866
g0     0.0001609861  0.0185642841 -0.0033465283
sigma -0.0009948866 -0.0033465283  0.0019755014

Fitted (real) parameters evaluated at base levels of covariates
       link   estimate SE.estimate        lcl        ucl
D       log  5.4788216  0.64671155  4.3507088  6.8994472
g0    logit  0.2731606  0.02705176  0.2234467  0.3292456
sigma   log 29.3695698  1.30602285 26.9193494 32.0428111
```

The density estimate is 5.48 ha^{-1} (95% confidence interval 4.35–6.90 ha^{-1}). The calculation used a default habitat mask with a buffer of 100 m around the detectors; this is reasonable in the light of the estimate of sigma (29.4 m).

Appendix 2. Software feature comparisons

● full implementation; ○ incomplete or inferior implementation.

Feature	DENSITY 4.4	**secr** 1.2	**secr** 1.3
General			
Graphical interface	●	○	○
Inverse prediction (IP SECR)	●	●	●
Maximum likelihood estimation (ML SECR)	●	●	●
Non-spatial open-population models (CJS etc.)	●		
Simulation of spatial sampling	●	○	○
Build detector arrays	●	○	○
Control of random number generator	○	●	●
ML SECR			
Jackknife confidence intervals	○		
Profile likelihood confidence intervals	●	●	●
Set of detectors used may vary with occasion	●	●	●
Fixed parameters	○	●	●
Parametric bootstrap	○	●	●
Between-session models	●	●	●
Mixture models for individual heterogeneity	●		●
Confidence ellipses	●		●
Formula-based model notation		●	●
Density models (inhomogeneous 2-D Poisson)		●	●
Terminology consistent with MARK[1]		●	●
Groups (e.g. males & females)		●	●
Score tests for model selection		●	●
Model averaging		●	●
Within-session variation in hazard-rate z		●	●
'pdot' criterion for region of integration		●	●
Structural relationships between real parameters		●	●
Detector types			
Single-catch trap (not MLE)	○	○	●
Multi-catch trap	●	●	●
Proximity	●	●	●
Signal strength (acoustic)[2]			●
Count[2]			●
Polygon[3]			○
Transect[3]			○

[1]'groups', 'real' parameters, 'beta' parameters
[2]Efford, Dawson & Borchers (2009) *Ecology* **90**, 2676–2682
[3]Efford in prep.

Appendix 3. Functions in secr arranged according to use

This list groups the main functions of **secr** 1.3. Many functions for data manipulation and plotting are omitted. S3 methods are marked with an asterisk *

Manipulate core objects

`make.grid`	construct detector array
`read.traps`	input detector locations from text file
`read.captures`	input detection (capture) data in Density format
`make.capthist`	form `capthist` from `traps` and detection data
`make.mask`	construct habitat mask (mesh)
`sim.capthist`	simulate capture histories
`verify`*	check `capthist`, `traps` or `mask` for internal consistency

Extract or replace attributes of `traps` *object*

`covariates`*	detector-level covariates
`detector`*	detector type ('multi', 'proximity' etc.)
`usage`*	disable detectors (occasion- and detector-specific)

Extract or replace attributes of `capthist` *object*

`covariates`*	individual-level covariates, including grouping factors
`session`*	session identifier(s)
`traps`*	embedded `traps` object(s)

Fit SECR model

`secr.fit`	maximum likelihood fit; result is a fitted `secr` object
`ip.secr`	fit simple SECR model by simulation & inverse prediction

Operate on fitted `secr` *object(s)*

`AIC`*	model selection, model weights
`coef`*	'beta' parameters
`collate`	tabulate estimates from several models
`confint`*	profile likelihood confidence intervals
`derived`	density from conditional likelihood models
`deviance`*	model deviance
`df.residual`*	degrees of freedom for deviance
`LR.test`	likelihood-ratio test of two models
`model.average`	combine estimates using AICc weights
`plot`*	plot detection functions with confidence bands
`predict`*	'real' parameters for arbitrary levels of predictor variables
`score.test`	model selection with score statistic using observed information
`simulate`*	generate realisations of fitted model
`sim.secr`	parametric bootstrap
`vcov`*	variance-covariance matrix of 'beta' or 'real' parameters

Miscellaneous

`autoini`	generate starting values of D, g0 and sigma for `secr.fit`
`counts`	summary data from `capthist` object
`dbar`	a simple (and unreliable) home-range measure
`distancetotrap`	from an arbitrary set of points
`nearesttrap`	from an arbitrary set of points
`pdot`	location-specific net probability of detection
`RPSV`	another simple and unreliable home-range measure

Datasets [restore with `data(xxx)`*]*

`rawdata`	dataframes of raw data
`captdata`	`rawdata` as a `capthist` object
`secrdemo`	`secr.fit` applied to some simulated data
`ovenbird`	multi-year mist-netting study of ovenbirds *Seiurus aurocapilla* at a site in Maryland, USA.
`ovensong`	acoustic detections of ovenbirds (Dawson & Efford *Journal of Applied Ecology* **46**, 1201–1209)
`possum`	brushtail possum *Trichosurus vulpecula* live trapping at Waitarere, North Island, New Zealand April 2002 (Efford et al. 2005 *Wildlife Society Bulletin* **33**, 731–738)