An introduction to model specification in **secr**

Murray Efford

2018-02-25

Contents

This document[1] introduces model specification for **secr** 3.1.

What do we mean by 'models' here?

A family of capture–recapture models (e.g. SECR) may include submodels that constrain variation in core parameters and include the effects of covariates. 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, +Inf).

Submodels for spatially explicit capture–recapture in **secr** are defined symbolically using the 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**: D (density), g0, sigma and z. Only the last three real parameters, the ones jointly defining detection probability as a function of location, can be estimated directly when the model is fitted by maximizing the conditional likelihood. D is then a derived parameter. 'z' is a shape parameter used only when the detection function requires three parameters. Other real parameters are used for acoustic models (beta0, beta1; secr-sound.pdf) and for the mixture proportion (pmix) in finite mixture models (secr-finitemixtures.pdf).

Each real parameter has a linear predictor of the form

y = X * beta,

[1] This material was once included in the help pages and an appendix to secr-overview.pdf

where y is vector of parameter values on the link scale, X is a design matrix of predictor values, beta is a vector of coefficients, and '*' stands for matrix multiplication. The elements of beta are coefficients to be estimated when we fit the model; in MARK these are called 'beta' parameters' to distinguish them from the 'real' parameter values in y. X has one column for each element of beta. To repeat: there is an X and a beta for each real parameter; elsewhere in the documentation we use 'beta' to refer to the vector got by concatenating *all* the parameter-specific beta's. We now describe design matrices in more detail.

[Some variations on the basic SECR model do not fit easily into this framework. An example is the choice of detection function (halfnormal vs hazard-rate). These are treated as higher-level choices.]

Design matrices

The design matrix contains a column of 1's (for the constant or intercept term) and additional columns as needed to describe the effects in the submodel. 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`, a design matrix is constructed automatically from the input data (capthist) and the model formula (e.g. model$g0) 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, and more if the formula includes interactions or factors with more than two levels. For a good description of the general approach see the documentation for RMark (Laake and Rexstad 2008). The key point is that the necessary design data can be either extracted from the inputs (capthist and mask) or generated automatically (e.g. indicator of previous capture, mentioned in the previous paragraph).

Real parameters fall into two groups: density (D) and detection (g0, lambda0, sigma and z). Density and detection parameters are subject to different types of effect, so they use different design matrices and are described separately in the following sections.

Detection Models

For spatially explicit capture–recapture estimation of a closed population, we model the detection of individual i on occasion s at detector k. Given n observed individuals on S occasions at K detectors there are therefore $n.S.K$ detection probabilities of interest. We can think of these as elements of a 3-dimensional array. 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 consider them in the array.

In a null (constant) model, all $n.S.K$ detection probabilities are the same. The conventional sources of variation in capture probability (Otis et al. 1978) appear as variation in the n dimension ('individual heterogeneity' h), 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.

Spatially explicit capture–recapture introduces two sorts of additional complexity. Firstly, detection probability is no longer a scalar (even for a particular animal, occasion and detector combination); it is described by the detection function, which may have two parameters (e.g. g0, sigma for half-normal), three parameters (e.g. g0, sigma, z for the hazard-rate function), or potentially more.

Secondly, many more types of variation are possible. Any of the parameters of the detection function may vary with respect to individual (i), occasion (s) or detector (k). For example, there may be a covariate associated with trap location that influences detection probability, and this effect may vary between occasions (see ?timevaryingcov).

The full design matrix for each detection submodel has one row for each combination of i, s and k (animal, occasion and trap). Allowing a distinct probability for each animal (the n dimension) may seem excessive,

as continuous individual-specific covariates are feasible only when a model is fitted by maximizing the conditional likelihood (cf Huggins 1989). However, the full $n.S.K$ array is convenient for coding both group membership (Lebreton et al. 1992, Cooch and White 2008) and experience of capture, even when individual-level heterogeneity cannot be modelled.

Variation between sessions and between latent classes in a finite mixture adds two further dimensions: in principle there is an $n.S.K$ array for each latent class (classes are numbered $1..M$), and an $n.S.K.M$ array for each session (sessions are numbered $1..R$). The full design matrix has $n.S.K.M.R$ rows. We do not expand on this here.

Potential effects on detection parameters

Effects on parameters of detection probability are specified with R formulae using standard variable names or named covariates supplied by the user. The formula for each detection parameter (g0, sigma, z) may be constant ($\sim$ 1}, the default) or some combination of terms in standard R formula notation (see ?formula).

Variable	Description	Data source	Dim
t	time factor (one level for each occasion)	automatic	S
T	time trend (integer covariate 0:(S-1))	automatic	S
tcov	default time covariate	timecov[,1]	S
kcov	default trap covariate	covariates (traps)[,1]	K
b	learned response	capthist	$n.S$
B	transient (Markovian) response	capthist	$n.S$
bk	animal x site learned response	capthist	$n.S.K$
Bk	animal x site transient response	capthist	$n.S.K$
k	site learned response	capthist	$S.K$
K	site transient response	capthist	$S.K$
g	group	see below	n
h2	2-class mixture	–	2
h3	3-class mixture	–	3
session	session factor (one level for each session)	automatic	R
Session	session number 0:(R-1)	automatic	R
ts	marking vs sighting occasions	automatic	S
tt	capture–recapture vs telemetry occasions	automatic	S
[user]	individual covariate	covariates (capthist)	n
[user]	session covariate	sessioncov	R
[user]	time covariate	timecov	S
[user]	detector covariate	covariates (traps)	K

Detection functions

[to be written]

Learned responses

The classic learned response is a step change following first detection; this is implemented with the predictor variable b which is FALSE up to and including the time of first capture and TRUE afterwards. An alternative is a response that depends only on detection at the last opportunity (B).

The site-specific learned and transient responses bk and Bk imply that an individual becomes trap happy or trap shy in relation to a particular detector, as in the wolverine example of Royle et al. (2011).

Groups

Groups (g) are defined by the interaction of the capthist categorical (factor) individual covariates identified in `secr.fit` argument groups. Groups are redundant with conditional likelihood because individual covariates of whatever sort (continuous or categorical) may be included freely in the model.

Individual heterogeneity

Individual heterogeneity (h in the notation of Otis et al. 1978) may modelled by treating any detection parameter as a 2-part or 3-part finite mixture e.g. g0 ~ h2. See secr-finitemixtures.pdf.

Covariates

Any other variable name appearing in a formula is assumed to refer to a user-defined predictor, also known as a 'covariate'. These will be interpreted by searching for name matches in the dataframes of individual, session, time and trap covariates, in that order (remembering that individual covariates other than groups are allowed only when the model is fitted by maximizing the conditional likelihood). The type of the predictor is inferred from the data frame in which it first occurs. Thus if the model included the formula 'g0 ~ wetness', and 'wetness' was a column in the data frame of time covariates (timecov), then 'wetness' would be interpreted as a time covariate, and a column of the same name in covariates(traps) would be ignored. In this case, renaming the column in timecov would expose the traps covariate, and 'wetness' would be interpreted as an attribute of detectors, rather than sample intervals. This is a good reason to give covariates distinctive names!

The design matrix for detection parameters may also be provided manually in the argument dframe. This feature requires some care and is better avoided.

The submodels for g0 (lambda0), sigma and z are named components of the model argument of `secr.fit`. They are expressed in R formula notation by appending terms to ~. The name of the response may optionally appear on the left hand side of the formula (e.g. g0 ~ b).

```
# constant (null) model
list(g0 = ~1, sigma = ~1)

# both detection parameters change after first capture
list(g0 = ~b, sigma = ~b)

# group-specific parameters; additive time effect on g0
# groups are defined via the groups argument of secr.fit
list(g0 = ~ g + t, sigma = ~ g)

# g0 depends on trap-specific covariate
list(g0 = ~ kcov)
```

The parameter z was previously called 'b'; it was renamed to avoid confusion with the predictor b used in a formula for a learned trap response.

Spatial variation not allowed

The structure of **secr** precludes certain types of model. **secr** does not allow spatial variation in detection parameters (g0, lambda0, sigma etc.) to be modelled directly (i.e. as a function of habitat mask covariates). This applies whether the variation is continuous or discrete (e.g. a function of habitat class). However, there is a workaround for spatially varying sigma, as shown in secr-noneuclidean.pdf.

Density Models

SECR can fit an inhomogeneous Poisson model to describe the distribution of animals. This may be viewed as a surface of expected density across the study area.

The log likelihood is evaluated in `secr.fit` by summing values at points on a 'habitat mask'. Each point in a habitat mask represents a grid cell of potentially occupied habitat (their combined area may be almost any shape and may include disjunct patches).

The density model may take one of two forms: a user-provided R function or a linear model on the link scale (see the link argument of `secr.fit`; the default link for density is 'log'). User-provided functions are described in the accompanying vignette secr-densitysurfaces.pdf. Here we focus on linear models.

The full design matrix for density (D) has one row for each point in the mask. 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', 'y', 'x2', 'y2', 'xy', 'session', 'Session' and 'g' are provided automatically. Other covariates should be named columns in the 'covariates' attribute of the habitat mask.

Variable	Description	Data source
x	x-coordinate	automatic
y	y-coordinate	automatic
x2	x-coordinate^2	automatic
y2	y-coordinate^2	automatic
xy	x-coordinate * y-coordinate	automatic
session	session factor	automatic
Session	session number 0:(R-1)	automatic
g	group factor	automatic
[user]	mask covariate	covariates(mask) as named in formula

The submodel for density (D) is a named component of the list used in the model argument of `secr.fit`. It is expressed in R formula notation by appending terms to ∼.

Density surfaces resulting from the fitting of SECR models are manipulated in **secr** as objects of class Dsurface. See the vignette secr-densitysurfaces.pdf for details and examples, including functions for prediction and plotting.

Note that no density model is fitted when `secr.fit` is called with CL = TRUE.

See also `?secr.fit`, `?Dsurface`, `?predictDsurface`, `plot.Dsurface`.

```
D = ~ 1       # constant density (homogeneous Poisson)
D = ~ x       # east-west trend
D = ~ cover   # requires 'cover' as a mask covariate
```

[smooth terms]

References

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) (2018) *Program MARK: A Gentle Introduction* 6th edition. Available online at http://www.phidot.org.

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.

Royle, J. A., Magoun, A. J., Gardner, B., Valkenburg, P. and Lowell, R. E. (2011) Density estimation in a wolverine population using spatial capture–recapture models. *Journal of Wildlife Management* **75**, 604–611.

Appendix. More on models in secr

A family of capture–recapture models, such as the Cormack-Jolly-Seber models for survival, may include submodels[2] that allow for variation in core ('real') 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 predictor variables 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. Set link functions with the 'link' argument of `secr.fit`.

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 because it will be familiar to biologists. Three real parameters are commonly modelled in **secr**; these are denoted D (for density), g0, and sigma. Only the last two real parameters, which jointly define the model for 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 a fitted secr object with the function `derived` or one of its siblings (`derivedCluster` etc.).

Other 'real' parameters appear in particular contexts. 'z' is a shape parameter that is used only when the detection function has three parameters (annular halfnormal, cumulative gamma, hazard-rate etc. – see `?detectfn`). 'lambda0' substitutes for 'g0' when the detection function is defined in terms of cumulative hazard. 'pmix' represents the mixing proportion in finite mixture models (or e.g., the sex ratio in hybrid mixture models with 'hcov').

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 parameters as 'coefficients' and real parameters as 'fitted values', a usage more in line with other statistical modelling in R. $\mathbf{X}$ has one column for each element of β. Design matrices are described in more detail in the next section.

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

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 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 2014). 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), or three parameters (e.g. g0, sigma, z). 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 2014) and experience of capture, even when individual-specific covariates 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 out of view: as a user you only need to know how to specify the detection model.