

Package ‘ipsecr’

June 26, 2022

Type Package

Title Spatially Explicit Capture-Recapture by Inverse Prediction

Version 1.1.2

Date 2022-06-22

Depends R (>= 3.5.0), secr (>= 4.5.4)

Imports graphics, grDevices, MASS, utils, parallel, stats, tools,
stringr, nlme, Rcpp (>= 0.12.14)

Suggests FrF2, rmarkdown, knitr, testthat

LinkingTo Rcpp, BH

VignetteBuilder knitr

Description Estimates the density of a spatially distributed animal population sampled with an array of passive detectors, such as traps. Models incorporating distance-dependent detection are fitted by simulation and inverse prediction as proposed by Efford (2004) <[doi:10.1111/j.0030-1299.2004.13043.x](https://doi.org/10.1111/j.0030-1299.2004.13043.x)>.

License GPL (>= 2)

LazyData yes

LazyDataCompression xz

URL <https://github.com/MurrayEfford/ipsecr/>,
<https://www.otago.ac.nz/density/>

NeedsCompilation yes

Author Murray Efford [aut, cre] (<<https://orcid.org/0000-0001-5231-5184>>)

Maintainer Murray Efford <murray.efford@otago.ac.nz>

Repository CRAN

Date/Publication 2022-06-26 21:30:04 UTC

R topics documented:

ipsecr-package	2
details	4

Internal	5
ipsecr.fit	7
ipsecrdemo	10
newdata	11
plot.ipsecr	12
predict.ipsecr	13
print.ipsecr	15
vcov.ipsecr	17

Index	18
--------------	-----------

ipsecr-package	<i>Spatially Explicit Capture–Recapture by Inverse Prediction</i>
----------------	---

Description

Functions to estimate the density and size of a spatially distributed animal population sampled with an array of passive detectors, such as traps. **ipsecr** addresses ‘difficult’ models that strictly cannot be fitted by maximum likelihood in the larger parent package **secr** (Efford 2022). The classic example concerns data from single-catch traps.

Details

Package: ipsecr
 Type: Package
 Version: 1.1.2
 Date: 2022-06-22
 License: GNU General Public License Version 2 or later

Spatially explicit capture–recapture is a set of methods for studying marked animals distributed in space. Data comprise the locations of detectors (described in an object of class ‘traps’), and the detection histories of individually marked animals. Individual histories are stored in an object of class ‘capthist’ that includes the relevant ‘traps’ object.

Models for population density (animals per hectare) and detection are defined in **ipsecr** using symbolic formula notation (but only constant models are allowed for detection parameters in **ipsecr** 1.1) Density models may include spatial trend. Habitat is distinguished from nonhabitat with an object of class ‘mask’.

Models are fitted in **ipsecr** by simulation and inverse prediction (Efford 2004). A model fitted with `ipsecr.fit` is an object of class `ipsecr`. Generic methods (plot, print, summary, etc.) are provided.

A link at the bottom of each help page takes you to the help index. Several vignettes complement the help pages:

The analyses in **ipsecr** extend those available in the software Density (see www.otago.ac.nz/density/ for the most recent version of Density). Help is available on the ‘DENSITY | secr’ forum at

www.phidot.org and the Google group [secrgroup](#). Feedback on the software is also welcome, including suggestions for additional documentation or new features consistent with the overall design.

‘Inverse prediction’ uses methods from multivariate calibration (Brown 1982). The goal is to estimate population density (D) and the parameters of a detection function (usually g_0 or λ_0 and σ) by ‘matching’ statistics from `proxyfn(capthist)` (the target vector) and statistics from simulations of a 2-D population using the postulated detection model. Statistics (see Note) are defined by the proxy function, which should return a vector equal in length to the number of parameters (default `np = 3`). Simulations of the 2-D population use either internal C++ code or `sim.popn`.

The simulated population is sampled with internal C++ code, `sim.capthist`, or a user-specified function. Simulations match the detector type (e.g., ‘single’ or ‘multi’) and detector layout specified in `traps(capthist)`, including allowance for varying effort if the layout has a `usage` attribute.

The simulated 2-D distribution of animals is Poisson by default.

Simulations are usually conducted on a factorial experimental design in parameter space - i.e. at the vertices of a cuboid ‘box’ centred on the working values of the parameters, plus an optional number of centre points.

A multivariate linear model is fitted to predict each vector of simulated proxies from the known parameter values. The number of simulations at each design point is increased until a specified precision is reached, up to the limit `max.nsim`.

Once a model with sufficient precision has been obtained, a new working vector of parameter estimates is ‘predicted’ by inverting the linear model and applying it to the target vector. A working vector is accepted as the final estimate when it lies within the box; this reduces the bias from using a linear approximation to extrapolate a nonlinear function. If the working vector lies outside the box then a new design is centred on value for each parameter in the working vector.

Once a final estimate is accepted, further simulations are conducted to estimate the variance-covariance matrix. These also provide a parametric bootstrap sample to evaluate possible bias. Set `var.nsim = 0` to suppress the variance step.

See Efford et al. (2004) for another description of the method, and Efford et al. (2005) for an application.

If not provided, the starting values are determined automatically with the `**secr**` function `makeStart`.

Linear measurements are assumed to be in metres and density in animals per hectare (10 000 m²).

If `ncores > 1` the **parallel** package is used to create processes on multiple cores (see [Parallel](#) for more).

Author(s)

Murray Efford <murray.efford@otago.ac.nz>

References

- Brown, P. J. (1982) Multivariate calibration. *Journal of the Royal Statistical Society, Series B* **44**, 287–321.
- Efford, M. G. (2004) Density estimation in live-trapping studies. *Oikos* **106**, 598–610.
- Efford, M. G. (2022) `secr`: Spatially explicit capture–recapture models. R package version 4.5.4. <https://CRAN.R-project.org/package=secr/>

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

Efford, M. G., Dawson, D. K. and Robbins C. S. (2004) DENSITY: software for analysing capture-recapture data from passive detector arrays. *Animal Biodiversity and Conservation* **27**, 217–228.

Efford, M. G., Warburton, B., Coleman, M. C. and Barker, R. J. (2005) A field test of two methods for density estimation. *Wildlife Society Bulletin* **33**, 731–738.

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**.

See Also

[proxy.ms ipsecr.fit](#), [secr.fit](#), [capthist](#), [mask](#)

details

Detail Specification for ipsecr.fit

Description

The function `ipsecr.fit` allows many options. Some of these are used infrequently and have been bundled as a single argument `details` to simplify the documentation. They are described here.

Detail components

`details$contrasts` may be used to specify the coding of factor predictors. The value should be suitable for the `'contrasts.arg'` argument of `model.matrix`. See 'Trend across sessions' in [secr-multisession.pdf](#) for an example.

`details$debug` is used only for debugging. In ordinary use it should not be changed from the default (FALSE).

`details$distribution` specifies the distribution of the number of individuals detected n ; this may be conditional on the number in the masked area ("binomial") or unconditional ("poisson"). `distribution` affects the sampling variance of the estimated density. The default is "poisson". The component 'distribution' may also take a numeric value larger than `nrow(capthist)`, rather than "binomial" or "poisson".

`details$factorial` "full" or "fractional" design. Fractional requires the package `**FrF2**` (Groemping 2014).

`details$FrF2args` list of arguments defining a fractional design

`details$ignorenontarget = TRUE` causes the function to ignore non-target information (the attribute `'nontarget'`).

`details$ignoreusage = TRUE` causes the function to ignore usage (varying effort) information in the traps component. The default (`details$ignoreusage = FALSE`) is to include usage in the model.

`details$popmethod` chooses between internal C++ code, the `secr` function `sim.popn`, and a user-provided R function with arguments `'mask'`, `'D'` (density per cell of mask) and `'N'` (number of individuals to simulate).

details\$CHmethod chooses between internal C++ code, the **secr** function `sim.caphist`, and a user-provided R function with arguments 'traps', 'popn', 'detectfn', 'detectpar', and 'noccasions'.

Defaults

- `boxsize1` scalar or vector of length `np` for size of design
- `boxsize2` as for `boxsize1`; used from second box onwards
- `centre` number of centre points in simulation design
- `min.nsim` minimum number of simulations per point
- `max.nsim` maximum number of simulations per point
- `dev.max` tolerance for precision of points in predictor space
- `var.nsim` number of additional simulations to estimate variance-covariance matrix
- `min.nbox` minimum number of attempts to 'frame' solution
- `max.nbox` maximum number of attempts to 'frame' solution
- `max.ntries` maximum number of attempts at each simulation

References

Groemping, U. (2014). R Package FrF2 for Creating and Analyzing Fractional Factorial 2-Level Designs. *Journal of Statistical Software*, **56**, 1–56. <https://www.jstatsoft.org/article/view/v056i01>.

See Also

[ipsecr.fit](#)

Internal

Internal Functions

Description

Functions called internally by **ipsecr**.

Usage

```
proxy.ms(caphist, model = list(D = ~1, NT = ~1), trapdesigndata = NULL, ...)
```

```
proxyfn0(caphist, N.estimator = c("n", "null", "zippin", "jackknife"), ...)
```

```
proxyfn1(caphist, N.estimator = c("n", "null", "zippin", "jackknife"), ...)
```

```
simpop(mask, D, N, distribution)
```

```
simCH(traps, popn, detectfn, detectpar, NT, noccasions, details = list())
```

Arguments

capthist	secr capthist object
model	named list of model formulae (see ipsecr.fit)
trapdesigndata	dataframe with one row for each detector and session
...	other arguments (not used)
N.estimator	character name of closed-population estimator
mask	secr mask object
D	numeric density in each mask cell
N	integer number of animals to simulate
distribution	character 'even' or 'poisson'
traps	detector locations as secr traps object
popn	animal locations as secr popn object
detectfn	integer code for detection function (see detectfn)
detectpar	named list of parameter values
NT	numeric hazard of non-target interference at each detector
noccasions	integer number of sampling occasions
details	list with optional additional named arguments

Details

proxy.ms is the default proxyfn used by [ipsecr.fit](#).

simpop is used by [ipsecr.fit](#) for popmethod 'internal'. It is faster and simpler than the **secr** function [sim.popn](#).

simCH is used by [ipsecr.fit](#) for CHmethod 'internal'. It is faster and simpler than the **secr** function [sim.capthist](#), and optionally simulates non-target interference.

D and NT are matrices with one column per session.

Value

proxy.ms – a numeric vector of length ≥ 3 corresponding to proxies for multi-session density and non-target interference models.

proxyfn0, proxyfn1 – a numeric vector of length 3 corresponding to proxies for population size, capture probability intercept and scale of detection.

simCH – a single-session capthist object.

References

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

See Also

[ipsecr.fit](#)

Examples

```
proxy.ms(captdata)
```

```
ipsecr.fit
```

Spatially Explicit Capture–Recapture by Inverse Prediction

Description

Estimate population density by simulation and inverse prediction (Efford 2004; Efford, Dawson & Robbins 2004). A restricted range of SECR models may be fitted.

Usage

```
ipsecr.fit(capthist, proxyfn = proxy.ms, model = list(D ~ 1, g0 ~ 1, sigma ~ 1),
  mask = NULL, buffer = 100, detectfn = "HN", binomN = NULL, start = NULL,
  link = list(), fixed = list(), timecov = NULL, sessioncov = NULL,
  details = list(), verify = TRUE, verbose = TRUE, ncores = NULL,
  seed = NULL, ...)
```

Arguments

capthist	secr capthist object including capture data and detector (trap) layout
proxyfn	function to compute proxy from capthist for each coefficient (beta parameter)
model	list with optional components each symbolically defining a linear predictor for one real parameter using formula notation
mask	mask object
buffer	scalar mask buffer radius in metres if mask not specified
detectfn	integer code or character string for shape of detection function 0 = halfnormal, 1 = hazard rate etc. – see detectfn
binomN	integer code for distribution of counts (see Details)
start	vector of initial values for beta parameters, or ipsecr object from which they may be derived
link	list with optional components corresponding to ‘real’ parameters (e.g., ‘D’, ‘g0’, ‘sigma’), each a character string in {"log", "logit", "identity", "sin"} for the link function of one real parameter
fixed	list with optional components corresponding to real parameters giving the scalar value to which the parameter is to be fixed
timecov	optional dataframe of values of time (occasion-specific) covariate(s). NOT USED
sessioncov	optional dataframe of values of session-specific covariate(s)

details	list of additional settings, to control estimation (see Details)
verify	logical, if TRUE the input data are checked with verify
verbose	logical, if TRUE then messages are output during execution
ncores	integer number of cores to use for parallel processing
seed	either NULL or an integer that will be used in a call to <code>set.seed</code>
...	other arguments passed to proxy function

Details

The vignette should be consulted for a full exposition.

`details` is used for various specialized settings listed below. Some of these are described separately - see [details](#).

Name	Default	Description
<code>boxsize1</code>	0.2	scalar or vector of length <code>np</code> for size of design
<code>boxsize2</code>	0.05	as for <code>boxsize1</code> ; used from second box onwards
<code>boxtype</code>	'absolute'	'absolute' or 'relative'
<code>centre</code>	3	number of centre points in simulation design
<code>dev.max</code>	0.002	tolerance for precision of points in predictor space
<code>var.nsim</code>	1000	number of additional simulations to estimate variance-covariance matrix
<code>min.nsim</code>	10	minimum number of simulations per point
<code>max.nsim</code>	2000	maximum number of simulations per point
<code>min.nbox</code>	2	minimum number of attempts to 'frame' solution
<code>max.nbox</code>	5	maximum number of attempts to 'frame' solution
<code>max.ntries</code>	2	maximum number of attempts at each simulation
<code>distribution</code>	'poisson'	'poisson', 'binomial' or 'even'
<code>even</code>	FALSE	'binomial' or 'even' population distribution in main simulations
<code>binomN</code>	0	integer code for distribution of counts (unused)
<code>ignorenontarget</code>	FALSE	override <code>nontarget</code> attribute of <code>capthist</code>
<code>ignoreusage</code>	FALSE	override usage in traps object of <code>capthist</code>
<code>debug</code>	FALSE	stop at arbitrary points in execution (varies)
<code>savecall</code>	TRUE	optionally suppress saving of call
<code>newdetector</code>	NULL	detector type to override <code>detector(traps(capthist))</code>
<code>contrasts</code>	NULL	coding of factor predictors
<code>popmethod</code>	'internal'	'internal' or 'sim.popn' or a user-provided function
<code>CHmethod</code>	'internal'	'internal' or 'sim.capthist' or a user-provided function
<code>factorial</code>	'full'	'full' or 'fractional' design
<code>FrF2args</code>	NULL	arguments for <code>FrF2</code> when <code>factorial = 'fractional'</code>

Value

An object of class 'ipsecr', a list comprising:

<code>call</code>	the function call
<code>capthist</code>	input
<code>mask</code>	input

detectfn	input
timecov	input
start	input
link	input
fixed	input
model	input
details	input
designD	list of design data for density
design	list of design data for detection parameters
design0	list of design data for detection parameters (naive animal)
parindx	mapping of coefficients (beta parameters) to real parameters
vars	names of covariates in model
betanames	names of coefficients
realnames	names of 'real' parameters
code	integer completion code: 1 successful, 2 target not within final box, 3 exceeded maximum simulations
beta	estimates of coefficients on link scale
beta.vcov	variance-covariance matrix of estimates
designbeta	vertices of final box (design points)
ip.nsim	total number of simulations
var.nsim.OK	number of successful variance simulations
variance.bootstrap	dataframe summarising simulations for variance estimation
version	package version
starttime	time execution started
proctime	processor time (seconds)

(The order and composition of the output list is subject to change).

References

- Efford, M. G. (2004) Density estimation in live-trapping studies. *Oikos* **106**, 598–610.
- Efford, M. G., Dawson, D. K. and Robbins C. S. (2004) DENSITY: software for analysing capture-recapture data from passive detector arrays. *Animal Biodiversity and Conservation* **27**, 217–228.

See Also

[proxy.ms](#), [predict.ipsecr](#), [summary.ipsecr](#)

Examples

```
ipsecrdemo <- ipsecr.fit(captdata, ncores = 1, buffer = 100, detectfn = 14, seed = 1237)
```

ipsecrdemo

*SECR Model Fitted to Demonstration Data***Description**

Demonstration data from program Density are provided as a `capthist` object (`captdata`) ready for input to `ipsecr.fit`.

The fitted models are objects of class `ipsecr` formed by

```
ipsecrdemo <- ipsecr.fit(captdata, ncores = 1, detectfn = 14, seed = 1237)
```

Usage

```
data(ipsecrdemo)
```

Details

The raw data are 235 fictional captures of 76 animals over 5 occasions in 100 single-catch traps 30 metres apart on a square grid with origin at (365,365).

The fitted model uses a hazard halfnormal detection function and default values of other arguments.

Object	Description
<code>ipsecrdemo</code>	fitted <code>ipsecr</code> model – null

References

Efford, M. G. (2012) *DENSITY 5.0: software for spatially explicit capture–recapture*. Department of Mathematics and Statistics, University of Otago, Dunedin, New Zealand. <https://www.otago.ac.nz/density/>.

See Also

[capthist](#), [read.capthist](#), [secrdemo](#)

Examples

```
predict(ipsecrdemo)
```

newdata	<i>Create Default Design Data</i>
---------	-----------------------------------

Description

Internal function used to generate a dataframe containing design data for the base levels of all predictors in an `ipsecr` object.

Usage

```
## S3 method for class 'ipsecr'  
makeNewData(object, all.levels = FALSE, ...)
```

Arguments

<code>object</code>	fitted <code>ipsecr</code> model object
<code>all.levels</code>	logical; if TRUE then all levels of factors are included
<code>...</code>	other arguments (not used)

Details

`makeNewData` is used by `predict` in lieu of user-specified 'newdata'. There is seldom any need to call the function `makeNewData` directly.

Value

A dataframe with one row for each session and group, and columns for the predictors used by `object$model`.

See Also

[predict.ipsecr](#), [ipsecr.fit](#)

Examples

```
## from previously fitted model  
makeNewData(ipsecrdemo)
```

plot.ipsecr

Plot Detection Functions

Description

Plot detection functions using estimates of parameters in an ipsecr object.

Usage

```
## S3 method for class 'ipsecr'
plot(x, newdata = NULL, add = FALSE,
      sigmatick = FALSE, rgr = FALSE, limits = FALSE, alpha = 0.05,
      xval = 0:200, ylim = NULL, xlab = NULL, ylab = NULL, ...)
```

Arguments

x	an ipsecr object
newdata	dataframe of data to form estimates
add	logical to add curve(s) to an existing plot
sigmatick	logical; if TRUE the scale parameter sigma is shown by a vertical line
rgr	logical; if TRUE a scaled curve $r.g(r)$ is plotted instead of $g(r)$
limits	logical; if TRUE pointwise confidence limits are drawn
alpha	alpha level for confidence intervals
xval	vector of distances at for which detection to be plotted
ylim	vector length 2 giving limits of y axis
xlab	label for x axis
ylab	label for y axis
...	arguments to pass to lines

Details

newdata is usually NULL, in which case one curve is plotted for each session and group. Otherwise, predict.ipsecr is used to form estimates and plot a curve for each row in newdata.

If axis labels are not provided they default to 'Distance (m)' and 'Detection probability' or 'Detection lambda'.

Approximate confidence limits for $g(r)$ are calculated using a numerical first-order delta-method approximation to the standard error at each xval. The distribution of $g(r)$ is assumed to be normal on the logit scale for non-hazard functions (detectfn 0:13). For hazard detection functions (detectfn 14:18) the hazard is assumed (from version 3.1.1) to be distributed normally on the log scale. Limits are back-transformed to the probability scale $g(r)$.

Value

plot.ipsecr invisibly returns a dataframe of the plotted values (or a list of dataframes in the case that newdata has more than one row).

See Also

[Detection functions](#), [plot](#), [ipsecr](#), [detectfnplot](#)

Examples

```
plot(ipsecrdemo, xval = 0:100, ylim = c(0, 0.4))
```

predict.ipsecr	<i>SECR Model Predictions</i>
----------------	-------------------------------

Description

Evaluate a spatially explicit capture–recapture model. That is, compute the ‘real’ parameters corresponding to the ‘beta’ parameters of a fitted model for arbitrary levels of any variables in the linear predictor.

Usage

```
## S3 method for class 'ipsecr'
predict(object, newdata = NULL, type = c("response", "link"),
        se.fit = TRUE, alpha = 0.05, savenew = FALSE, ...)
```

Arguments

object	ipsecr object output from ipsecr.fit
newdata	optional dataframe of values at which to evaluate model
type	character; type of prediction required. The default ("response") provides estimates of the ‘real’ parameters.
se.fit	logical for whether output should include SE and confidence intervals
alpha	alpha level for confidence intervals
savenew	logical for whether newdata should be saved
...	other arguments passed to newdata

Details

The variables in the various linear predictors are described in [seccr-models.pdf](#) and listed for the particular model in the vars component of object.

Optional newdata should be a dataframe with a column for each of the variables in the model (see 'vars' component of object). If newdata is missing then a dataframe is constructed automatically.

Default newdata are for a naive animal on the first occasion; numeric covariates are set to zero and factor covariates to their base (first) level. The argument 'all.levels' may be passed to newdata; if TRUE then the default newdata includes all factor levels.

realnames may be used to select a subset of parameters.

Standard errors for parameters on the response (real) scale are by the delta method (Lebreton et al. 1992), and confidence intervals are backtransformed from the link scale.

The value of newdata is optionally saved as an attribute.

Value

When `se.fit = FALSE`, a dataframe identical to newdata except for the addition of one column for each 'real' parameter. Otherwise, a list with one component for each row in newdata. Each component is a dataframe with one row for each 'real' parameter (density, g0, sigma, b) and columns as below

link	link function
estimate	estimate of real parameter
SE.estimate	standard error of the estimate
lcl	lower 100(1-alpha)% confidence limit
ucl	upper 100(1-alpha)% confidence limit

When newdata has only one row, the structure of the list is 'dissolved' and the return value is one data frame.

For detectpar, a list with the estimated values of detection parameters (e.g., g0 and sigma if detectfn = "halfnormal"). In the case of multi-session data the result is a list of lists (one list per session).

Note

[predictDsurface](#) should be used for predicting density at many points from a model with spatial variation. This deals automatically with scaling of x- and y-coordinates, and is much faster than predict.ipsecr. The resulting Dsurface object has its own plot method.

The argument 'scaled' was removed from both predict methods in version 2.10 as the scaleg0 and scalesigma features had been superceded by other parameterisations.

References

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.

See Also

[ipsecr.fit](#), [predictDsurface](#)

Examples

```
predict (ipsecrdemo)
```

```
print.ipsecr          Print or Summarise ipsecr Object
```

Description

Print results from fitting a spatially explicit capture–recapture model or generate a list of summary values.

Usage

```
## S3 method for class 'ipsecr'
print(x, newdata = NULL, alpha = 0.05, call = TRUE, ...)
## S3 method for class 'ipsecr'
summary(object, newdata = NULL, alpha = 0.05, ...)
```

Arguments

x	ipsecr object output from ipsecr.fit
object	ipsecr object output from ipsecr.fit
newdata	optional dataframe of values at which to evaluate model
alpha	alpha level
call	logical; if TRUE the call is printed
...	other arguments (not used)

Details

Results from `print.ipsecr` are potentially complex and depend upon the analysis (see below). Optional `newdata` should be a dataframe with a column for each of the variables in the model. If `newdata` is missing then a dataframe is constructed automatically. Default `newdata` are for a naive animal on the first occasion; numeric covariates are set to zero and factor covariates to their base (first) level. Confidence intervals are 100 (1 – alpha) % intervals.

call	the function call (optional)
version,time	ipsecr version, date and time fitting started, and elapsed time
Detector type	'single', 'multi', 'proximity' etc.
Detector number	number of detectors
Average spacing	

x-range	
y-range	
New detector type	as fitted when details\$newdetector specified
N animals	number of distinct animals detected
N detections	number of detections
N occasions	number of sampling occasions
Mask area	
Model	model formula for each 'real' parameter
Fixed (real)	fixed real parameters
Detection fn	detection function type (halfnormal or hazard-rate)
Distribution	spatial model (details\$distribution)
N parameters	number of parameters estimated
Design points	number of vertices and centre points
Simulations per box	total number
Beta parameters	coef of the fitted model, SE and confidence intervals
vcov	variance-covariance matrix of beta parameters
Real parameters	fitted (real) parameters evaluated at base levels of covariates

Value

The summary method constructs a list of outputs similar to those printed by the `print` method, but somewhat more concise and re-usable:

versiontime	ipsecr version, and date and time fitting started
traps	detector summary
capthist	capthist summary
mask	mask summary
modeldetails	miscellaneous model characteristics
coef	table of fitted coefficients with CI
predicted	predicted values ('real' parameter estimates)

See Also

[ipsecr.fit](#)

Examples

```
## load & print previously fitted null (constant parameter) model
print(ipsecrdemo)

summary(ipsecrdemo)
```

vcov.ipsecr	<i>Variance - Covariance Matrix of SECR Parameters</i>
-------------	--

Description

Variance-covariance matrix of beta or real parameters from fitted ipsecr model.

Usage

```
## S3 method for class 'ipsecr'
vcov(object, realnames = NULL, newdata = NULL,
      byrow = FALSE, ...)
```

Arguments

object	ipsecr object output from the function ipsecr.fit
realnames	vector of character strings for names of ‘real’ parameters
newdata	dataframe of predictor values
byrow	logical for whether to compute covariances among ‘real’ parameters for each row of new data, or among rows for each real parameter
...	other arguments (not used)

Details

By default, returns the matrix of variances and covariances among the estimated model coefficients (beta parameters).

If `realnames` and `newdata` are specified, the result is either a matrix of variances and covariances for each ‘real’ parameter among the points in predictor-space given by the rows of `newdata` or among real parameters for each row of `newdata`. Failure to specify `newdata` results in a list of variances only.

Value

A matrix containing the variances and covariances among beta parameters on the respective link scales, or a list of among-parameter variance-covariance matrices, one for each row of `newdata`, or a list of among-row variance-covariance matrices, one for each ‘real’ parameter.

See Also

[vcov](#), [ipsecr.fit](#), [print.ipsecr](#)

Examples

```
## previously fitted ipsecr model
vcov(ipsecrdemo)
```

Index

- * **datasets**
 - ipsecrdemo, 10
- * **hplot**
 - plot.ipsecr, 12
- * **models**
 - details, 4
 - newdata, 11
 - predict.ipsecr, 13
 - vcov.ipsecr, 17
- * **package**
 - ipsecr-package, 2
- * **print**
 - print.ipsecr, 15

capthist, 4, 10

details, 4, 8

detectfn, 6, 7

detectfnplot, 13

fixedbeta (details), 4

Internal, 5

ipsecr, 13

ipsecr (ipsecr-package), 2

ipsecr-package, 2

ipsecr.fit, 2, 4–6, 7, 11, 15–17

ipsecrdemo, 10

makeNewData.ipsecr (newdata), 11

makeStart, 3

mask, 4, 7

model.matrix, 4

newdata, 11, 13

Parallel, 3

param (details), 4

plot, 13

plot.ipsecr, 12

predict.ipsecr, 9, 11, 13

predictDsurface, 14, 15

print.ipsecr, 15, 17

proxy.ms, 4, 9

proxy.ms (Internal), 5

proxyfn0 (Internal), 5

proxyfn1 (Internal), 5

read.capthist, 10

secr.fit, 4

secrdemo, 10

sim.capthist, 3, 6

sim.popn, 3, 6

simCH (Internal), 5

simpop (Internal), 5

summary.ipsecr, 9

summary.ipsecr (print.ipsecr), 15

usage, 3

vcov, 17

vcov.ipsecr, 17

verify, 8