Package ‘BBRecapture’

February 19, 2015

Type Package

Title Bayesian Behavioural Capture-Recapture Models

Version 0.1

Date 2013-12-18

Author Luca Tardella and Danilo Alunni Fegatelli

Maintainer Danilo Alunni Fegatelli
<danilo.alunnifegatelli@uniroma1.it>

Depends R (>= 3.0.2), HI, locfit, lme4, secr

Description Model fitting of flexible behavioural recapture models based on conditional probability reparameterization and meaningful partial capture history quantification also referred to as meaningful behavioural covariate

License GPL-2

NeedsCompilation no

Repository CRAN

Date/Publication 2013-12-19 12:22:14

R topics documented:

BBRecapture-package *Bayesian Behavioural Capture-Recapture Models*

Description

Model fitting of flexible behavioural recapture models based on conditional probability reparameterization and meaningful partial capture history quantification also referred to as *meaningful behavioural covariate*

Details

Package:	BBRecapture
Type:	Package
Version:	0.1
Date:	2013-12-18
License:	GPL-2

This BBRecap package has been built up to help researchers to fit some relevant classes of capture-recapture models within the framework of Bayesian inference. Special emphasis is given on recently developed tools to take into account flexible behavioral response to capture. The main function developed in the package relies on the generalized linear model framework in the spirit of Huggins (1989) and Alho (1990) for regressing the capture occurrence on previous partial capture histories although shortcuts have been embedded to reduce computational complexity whenever possible. There are also some functions which fit the same class of models maximizing the unconditional likelihood as opposed to the most frequently used approach based on the conditional likelihood (Huggins and Hwang, 2011). There are theoretical arguments related to the so-called likelihood failure (Alunni Fegatelli and Tardella, 2013; Carle and Strub, 1978) which support the use of a Bayesian approach for the estimation of the unknown population size in the presence of behavioral response to capture. Some simulation studies have been also carried out in Alunni Fegatelli (2013) to highlight the occurrence of the likelihood failure pathology and the loss of inferential performance of the conditional likelihood approach even in the absence of failure. In the same circumstances the unconditional likelihood approach should be preferred to the conditional likelihood but both of them are in any case outperformed by the Bayesian approach. Functions in the package are designed to allow minimal efforts by the researcher although optional arguments often allow for a more customized and refined model building.

Author(s)

Luca Tardella and Danilo Alunni Fegatelli

Maintainer: Danilo Alunni Fegatelli <danilo.alunnifegatelli@uniroma1.it>

References

Alho, J.M. (1990). Logistic regression in capture-recapture models. Biometrics, 46, 623–635.

Carle, F.L. and Strub, M.R. (1978) A new method for estimating population size from removal data. Biometrics, 34, 621–630.

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

Huggins, R. and Hwang, HW (2011) A review of the use of conditional likelihood in capture-recapture experiments. International Statistical Review, 79, 385–400

Farcomeni, A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika, 98, 237–242

Alunni Fegatelli, D. (2013) New methods for capture-recapture modelling with behavioural response and individual heterogeneity. PhD Thesis. http://padis.uniroma1.it/bitstream/10805/2085/1/TesiDottorato-AlunniFegatelliDanilo.pdf

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications, 22:45-66 (DOI: 10.1007/S10260-012-0221-4)

Examples

```
data(greatcopper)
out=BBRecap(greatcopper,mod="Mb")
print(out)
```

BBRecap *Bayesian inference for capture-recapture analysis with emphasis on behavioural effect modelling*

Description

Bayesian inference for a large class of discrete-time capture-recapture models under closed population with special emphasis on behavioural effect modelling including also the *meaningful behavioral covariate* approach proposed in Alunni Fegatelli (2013) [PhD thesis]. Many of the standard classical models such as M_0, M_b, M_{c_1}, M_t or M_{bt} can be regarded as particular instances of the aforementioned approach. Other flexible alternatives can be fitted through a careful choice of a meaningful behavioural covariate and a possible partition of its admissible range

Usage

```
BBRecap (data,last.column.count=FALSE, neval = 1000, by.incr = 1,
  mbc.function = c("standard","markov","counts","integer","counts.integer"),
  mod = c("linear.logistic", "M0", "Mb", "Mc", "Mcb", "Mt", "Msubjective.cut",
  "Msubjective"), nsim = 5000, burnin = round(nsim/10),
  nsim.ML = 1000, burnin.ML = round(nsim.ML/10), num.t = 50,
```

```
markov.ord=NULL, prior.N = c("Rissanen","Uniform","one.over.N","one.over.N2"),
 meaningful.mat.subjective = NULL, meaningful.mat.new.value.subjective = NULL,
 z.cut=NULL, output = c("base", "complete", "complete.ML"))
```

Arguments

`data` can be one of the following:

1. an M by t binary matrix/data.frame. In this case the input is interpreted as a matrix whose rows contain individual capture histories for all M observed units
2. a matrix/data.frame with $(t + 1)$ columns. The first t columns contain binary entries corresponding to capture occurrences, while the last column contains non negative integers corresponding to frequencies. This format is allowed only when `last.column.count` is set to `TRUE`
3. a t-dimensional array or table representing the counts of the 2^t contingency table of binary outcomes

M is the number of units captured at least once and t is the number of capture occasions.

`last.column.count` a logical. In the default case `last.column.count=FALSE` each row of the input argument `data` represents the complete capture history for each observed unit. When `last.column.count` is set to `TRUE` in each row the first t entries represent one of the observed complete capture histories and the last entry in the last column is the number of observed units with that capture history

`neval` a positive integer. `neval` is the number of values of the population size N where the posterior is evaluated starting from M. The default value is `neval`=1000.

`by.incr` a positive integer. `by.incr` represents the increment on the sequence of possible population sizes N where the posterior is evaluated. The default value is `by.incr`=1. The use of `by.incr`>1 is discouraged unless the range of N values of interest is very large

`mbc.function` a character string with possible entries (see Alunni Fegatelli (2013) for further details)

1. `"standard"` meaningful behavioural covariate in [0,1] obtained through the normalized binary representation of integers relying upon partial capture history
2. `"markov"` slight modification of `"standard"` providing consistency with arbitrary Markov order models when used in conjunction with the options `"Msubjective"` and `z.cut`.
3. `"counts"` covariate in [0,1] obtained by normalizing the integer corresponding to the sum of binary entries i.e. the number of previous captures
4. `"integer"` un-normalized integer corresponding to the binary entries of the partial capture history
5. `"counts.integer"` un-normalized covariate obtained as the sum of binary entries i.e. the number of previous captures

mod — a character. `mod` represents the behavioural model considered for the analysis. `mod="linear.logistic"` is the model proposed in Alunni Fegatelli (2013) based on the meaningful behavioural covariate. `mod="M0"` is the most basic model where no effect is considered and all capture probabilities are the same. `mod="Mb"` is the classical behavioural model where the capture probability varies only once when first capture occurs. Hence it represents an *enduring* effect to capture. `mod="Mc"` is the *ephemeral* behavioural Markovian model originally introduced in Yang and Chao (2005) and subsequently extended in Farcomeni (2011) and reviewed in Alunni Fegatelli and Tardella (2012) where capture probability depends only on the capture status (captured or uncaptured) in the previous `k=markov.ord` occasions. `mod="Mcb"` is an extension of Yang and Chao's model (2005); it considers both *ephemeral* and *enduring* effect to capture. `mod="Mt"` is the standard temporal effect with no behavioural effect. `mod="Msubjective.cut"` is an alternative behavioural model obtained through a specific cut on the meaningful behavioural covariate interpreted as memory effect. `mod="Msubjective"` is a customizable (subjective) behavioural model within the linear logistic model framework requiring the specification of the two additional arguments: the first one is `meaningful.mat.subjective` and contains an M by t matrix of ad-hoc meaningful covariates depending on previous capture history; the second one is `meaningful.mat.new.value.subjective` and contains a vector of length t corresponding to meaningful covariates for a generic uncaptured unit. The default value for `mod` is `"linear.logistic"`.

nsim — a positive integer. `nsim` is the number of iterations for the Metropolis-within-Gibbs algorithm which allows the approximation of the posterior. It is considered only if `mod` is `"linear.logistic"` or `"Msubjective"`. In the other cases closed form evaluation of the posterior is available up to a proportionality constant. The default value is `nsim=10000`.

burnin — a positive integer. `burnin` is the initial number of MCMC samples discarded. It is considered only if `mod` is `"linear.logistic"` or `"Msubjective"`. The default value for `burnin` is `round(nsim/10)`.

nsim.ML — a positive integer. `nsim.ML` is the number of iterations used in the marginal likelihood estimation procedure via power posterior method of Friel and Pettit (2008). This approach is implemented only when `mod="linear.logistic"` or `mod="Msubjective"` and when `output` is set to `"complete.ML"`. The default value is `nsim.ML=500`.

burnin.ML — a positive integer. `burnin.ML` is the initial number of samples discarded for marginal likelihood estimation via power-posterior approach. It is considered only if `mod` is `"linear.logistic"` or `"Msubjective"` and when `output` is set to `"complete.ML"`. The default value is `burnin.ML` is `round(nsim/10)`.

num.t — a positive integer. `num.t` is the number of powers used in the power posterior approximation method for the marginal likelihood evaluation. It is used only when `output="complete.ML"`.

markov.ord — a positive integer. `markov.ord` is the order of Markovian model M_c or M_{cb}. It is considered only if `mod="Mc"` or `mod="Mcb"`.

prior.N — a character. `prior.N` is the label for the prior distribution for N. When `prior.N` is set to `"Rissanen"` (default) the Rissanen prior is used as a prior on N. This distribution was first proposed in Rissanen 1983 as a universal prior on integers. `prior.N="Uniform"` stands for a prior on N proportional to a constant value. `prior.N="one.over.N"` stands for a prior on N proportional to $1/N$. `prior.N="one.over.N2"` stands for a prior on N proportional to $1/N^2$.

`meaningful.mat.subjective`
: `M x t` matrix containing numerical covariates to be used for a customized logistic model approach

`meaningful.mat.new.value.subjective`
: `1 x t` numerical vector corresponding to auxiliary covariate to be considered for unobserved unit

`z.cut`
: numeric vector. `z.cut` is a vector containing the cut point for the memory effect covariate. It is considered only if `mod="Msubjective.cut"`

`output`
: a character. `output` selects the kind of output from a very basic summary info on the posterior output (point and interval estimates for the unknown N) to more complete details including MCMC simulations for all parameters in the model when appropriate

Details

Independent uniform distributions are considered as default prior for the nuisance parameters. If `model="linear.logistic"` or `model="Msubjective"` and `output="complete.ML"` the marginal likelihood estimation is performed through the *power posteriors method* suggested in Friel and Pettit (2008). In that case the `BBRecap` procedure is computing intensive for high values of `neval` and `nsim`.

Value

`Model`
: model considered

`Prior`
: prior distribution for N

`N.hat.mean`
: posterior mean for N

`N.hat.median`
: posterior median for N

`N.hat.mode`
: posterior mode for N

`N.hat.RMSE`
: minimizer of a specific loss function connected with the Relative Mean Square Error

`HPD.N`
: 95% highest posterior density interval estimate for N

`log.marginal.likelihood`
: log marginal likelihood

`N.range`
: values of N considered

`posterior.N`
: values of the posterior distribution for each N considered

`z.matrix`
: meaningful behavioural covariate matrix for the observed data

`vec.cut`
: cut point used to set up meaningful partitions the set of the partial capture histories according to the value of the value of the meaningful behavioural covariate

`N.vec`
: simulated values from the posterior marginal distribution of N

`mean.a0`
: posterior mean of the parameter a0

`mean.a0`	highest posterior density interval estimate of the parameter a0
`a0.vec`	simulated values from the posterior marginal distribution of a0
`mean.a1`	posterior mean of the parameter a1
`mean.a1`	highest posterior density interval estimate of the parameter a1
`a1.vec`	simulated values from the posterior marginal distribution of a1

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Friel, N. and Pettitt, A. N. (2008) Marginal likelihood estimation via power posteriors. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 70(3):589–607

Farcomeni A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika 98(1):237–242

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

See Also

`BBRecap.custom.part`, `LBRecap`

Examples

```
## Not run:
data(greatcopper)

mod.Mb=BBRecap(greatcopper,mod="Mb")
str(mod.Mb)

## End(Not run)
```

`BBRecap.all` *Comparative Bayesian analysis of alternative flexible behavioural and time effect models*

Description

Comparative point and interval estimates for the population size N obtained fitting many alternative behavioural and time effect capture-recapture models. Log marginal likelihood is reported for each alternative model.

Usage

```
BBRecap.all(data, last.column.count=FALSE, neval=1000, by.incr=1,nsim=10000,
  burnin=round(nsim/10),nsim.ML=500,burnin.ML=round(nsim.ML/10), num.t = 50,
  prior.N = c("Rissanen","Uniform","one.over.N","one.over.N2"),
  which.mod=c("all","standard"), sort=c("default","log.ML"))
```

Arguments

`data` can be one of the following:

1. an M by t binary matrix/data.frame. In this case the input is interpreted as a matrix whose rows contain individual capture histories for all M observed units
2. a matrix/data.frame with $(t + 1)$ columns. The first t columns contain binary entries corresponding to capture occurrences, while the last column contains non negative integers corresponding to frequencies. This format is allowed only when `last.column.count` is set to `TRUE`
3. a t-dimensional array or table representing the counts of the 2^t contingency table of binary outcomes

M is the number of units captured at least once and t is the number of capture occasions.

`last.column.count` a logical. In the default case `last.column.count=FALSE` each row of the input argument `data` represents the complete capture history for each observed unit. When `last.column.count` is set to `TRUE` in each row the first t entries represent one of the observed complete capture histories and the last entry in the last column is the number of observed units with that capture history

`neval` a positive integer. `neval` is the number of values of the population size N where the posterior is evaluated starting from M. The default value is `neval`=1000.

`by.incr` a positive integer. `by.incr` represents the increment on the sequence of possible population sizes N where the posterior is evaluated. The default value is `by.incr`=1. The use of `by.incr`>1 is discouraged unless the range of N values of interest is very large

`nsim` a positive integer. `nsim` is the number of iterations for the Metropolis-within-Gibbs algorithm which allows the approximation of the posterior. It is considered only if `mod` is `"linear.logistic"` or `"Msubjective"`. In the other cases closed form evaluation of the posterior is available up to a proportionality constant. The default value is `nsim=10000`.

`burnin` a positive integer. `burnin` is the initial number of MCMC samples discarded. It is considered only if `mod` is `"linear.logistic"` or `"Msubjective"`. The default value for `burnin` is `round(nsim/10)`.

`nsim.ML` a positive integer. Whenever MCMC is needed `nsim.ML` is the number of iterations used in the marginal likelihood estimation procedure via power posterior method of Friel and Pettit (2008)

`burnin.ML` a positive integer. Whenever MCMC is needed `burnin.ML` is the initial number of samples discarded for marginal likelihood estimation via power-posterior approach. The default value is `burnin.ML` is `round(nsim/10)`.

`num.t`	a positive integer. Whenever MCMC is needed `num.t` is the number of powers used in the power posterior approximation method for the marginal likelihood evaluation.
`prior.N`	a character. `prior.N` is the label for the prior distribution for N. When `prior.N` is set to `"Rissanen"` (default) the Rissanen prior is used as a prior on N. This distribution was first proposed in Rissanen 1983 as a universal prior on integers. `prior.N="Uniform"` stands for a prior on N proportional to a constant value. `prior.N="one.over.N"` stands for a prior on N proportional to $1/N$. `prior.N="one.over.N2"` stands for a prior on N proportional to $1/N^2$.
`which.mod`	a character. `which.mod` selects which models are fitted and compared. In the default setting `which.mod="all"` all alternative models are fitted including new behavioural models based on alternative meaningful covariates (see Details). When `which.mod="standard"` the function only fits classical behavioural models with either enduring effects as in M_b, M_{c_1b}, M_{c_2b} or ephemeral effects as in purely Markovian M_{c_1} and M_{c_2}
`sort`	character. `sort` selects the order of models.

Details

The available models are: M_0, M_b, M_t, M_{c_1}, M_{c_1b}, M_{c_2}, M_{c_2b}, M_{mc}, $M_{mc_{int}}$, $M_{mc_{count}}$ and $M_{mc_{count.int}}$. This function `BBRecap.all` can be computing intensive for high values of `neval` and `nsim`.

Value

A dataframe with one row corresponding to each model and the following columns:

model: model considered

npar: number of parameters

log.marginal.likelihood: log marginal likelihood

Nhat: estimate of population size

Ninf: lower 95% highest posterior density interval

Nsup: upper 95% highest posterior density interval

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Otis D. L., Burnham K. P., White G. C, Anderson D. R. (1978) Statistical Inference From Capture Data on Closed Animal Populations, Wildlife Monographs.

Yang H.C., Chao A. (2005) Modeling animals behavioral response by Markov chain models for capture-recapture experiments, Biometrics 61(4), 1010-1017

N. Friel and A. N. Pettitt. Marginal likelihood estimation via power posteriors. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 70(3):589, 607–2008

Farcomeni A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika 98(1):237–242

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

Alunni Fegatelli D. (2013) New methods for capture-recapture modelling with behavioural response and individual heterogeneity.

See Also

BBRecap,

Examples

```
## Not run:
data(hornedlizard)
BBRecap.all(hornedlizard,neval=200)

## End(Not run)
```

BBRecap.custom.part — *Bayesian inference for behavioural effect models based on a partition of the set of all partial capture histories*

Description

Bayesian inference for a general model framework based on the capture probabilities conditioned on each possible partial capture history. As suggested in Alunni Fegatelli and Tardella (2012) the conditional approach originally proposed in Farcomeni (2011) [saturated reparameterization] is reviewed in terms of partitions into equivalence classes of conditional probabilities. In this function the user can directly provide the model as a partition.

Usage

```
BBRecap.custom.part (data,last.column.count=FALSE, partition, neval = 1000,
   by.incr = 1, prior.N = c("Rissanen", "Uniform", "one.over.N", "one.over.N2"),
   output = c("base", "complete"))
```

Arguments

`data` can be one of the following:

1. an M by t binary matrix/data.frame. In this case the input is interpreted as a matrix whose rows contain individual capture histories for all M observed units

2. a matrix/data.frame with $(t + 1)$ columns. The first t columns contain binary entries corresponding to capture occurrences, while the last column contains non negative integers corresponding to frequencies. This format is allowed only when `last.column.count` is set to `TRUE`
3. a t-dimensional array or table representing the counts of the 2^t contingency table of binary outcomes

M is the number of units captured at least once and t is the number of capture occasions.

`last.column.count`
: a logical. In the default case `last.column.count=FALSE` each row of the input argument `data` represents the complete capture history for each observed unit. When `last.column.count` is set to `TRUE` in each row the first t entries represent one of the observed complete capture histories and the last entry in the last column is the number of observed units with that capture history

`partition`
: list. `partition` represents a partition of the set of all partial capture histories.

`neval`
: a positive integer. `neval` is the number of values of the population size N where the posterior is evaluated starting from M. The default value is `neval`=1000.

`by.incr`
: a positive integer. `by.incr` represents the increment on the sequence of possible population sizes N where the posterior is evaluated. The default value is `by.incr`=1. The use of `by.incr`>1 is discouraged unless the range of N values of interest is very large

`prior.N`
: a character. `prior.N` is the label for the prior distribution for N. When `prior.N` is set to `"Rissanen"` (default) the Rissanen prior is used as a prior on N. This distribution was first proposed in Rissanen 1983 as a universal prior on integers. `prior.N="Uniform"` stands for a prior on N proportional to a constant value. `prior.N="one.over.N"` stands for a prior on N proportional to $1/N$. `prior.N="one.over.N2"` stands for a prior on N proportional to $1/N^2$.

`output`
: a character. `output` selects the kind of output from a very basic summary info on the posterior output (point and interval estimates for the unknown N) to more complete details

Details

Uniform prior distribution is considered for the nuisance parameters.

Value

`Prior`
: prior distribution for `N`.

`N.hat.mean`
: posterior mean for N

`N.hat.median`
: posterior median for N

`N.hat.mode`
: posterior mode for N

`N.hat.RMSE`
: minimizer of a specific loss function connected with the Relative Mean Square Error.

`HPD.N`
: 95% highest posterior density interval estimate for N.

`log.marginal.likelihood`
: log marginal likelihood.

N.range	values of N considered.
posterior.N	values of the posterior distribution for each N considered
partition	partition of the set H

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

Farcomeni A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika 98(1):237–242

See Also

partition.ch, LBRecap.custom.part, BBRecap

Examples

```
data(greatcopper)
partition.Mc1=partition.ch(quant.binary,t=ncol(greatcopper),breaks=c(0,0.5,1))
mod.Mc1=BBRecap.custom.part(greatcopper,partition=partition.Mc1)
str(mod.Mc1)
```

---

greatcopper *Great-Copper butterfly data*

Description

Great Copper butterflies in Willamette Valley of Oregon. Once presumed to be extinct from western Oregon since 1970 (R. A. Pyle, 2002, Butterflies of Cascadia, Seattle Audubon Society), the great copper (Lycaena xanthoides Lycaenidae), was rediscovered in several wetland prairie remnants in the summer of 2004 (see Severns and Villegas 2005 for an account). Until this rediscovery, the habitat and host plants were unknown for western Oregon populations, and the great copper was conspicuously sparse in collections, with a total of twelve specimens captured between 1896 and 1970. A current estimate of the extant populations is alarmingly low, about 100 individuals across three subpopulations in the Willamette Valley. The great copper occurs in much larger numbers through much of California. http://people.oregonstate.edu/~wilsomar/Persp_GrtCop.htm

This dataset has been first analyzed in Ramsey F, Severns P (2010) and later re-analyzed in Farcomeni (2011) and Alunni Fegatelli and Tardella (2012)

Usage

```
greatcopper
```

Format

A matrix with 45 rows (observed butterflies) and 8 columns (capture occasions)

References

Severns, P.M. and Villegas, S. (2005) Butterflies Hanging on to Existence in the Willamette Valley: A Relict Population of the Great Copper (Lycaena xanthoides Boisduval), Northwest Science, Vol. 79, No. 1, 77–80.

Severns, P.M. and Villegas, S. (2006) Conserving a wetland butterfly: quantifying early lifestage survival through seasonal flooding, adult nectar, and habitat preference, Journal of Insect Conservation 10 (4), 361-370

Ramsey F, Severns P (2010) Persistence models for mark-recapture. Environmental and Ecological Statistics 17(1):97–109

Farcomeni A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika 98(1):237–242

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

Examples

```
data(greatcopper)
```

`hornedlizard` *Flat-tailed Horned Lizard Dataset*

Description

Data from multiple searches for flat-tailed horned lizards (*Phrynosoma mcalli*) on a plot in Arizona, USA.

Usage

```
hornedlizard
```

Format

A matrix with 68 rows (observed lizards) and 14 columns (capture occasions)

Details

The flat-tailed horned lizard (*Phrynosoma mcalli*) is a desert lizard found in parts of southwestern Arizona, southeastern California and northern Mexico. There is considerable concern about its conservation status. The species is cryptically colored and has the habit of burying under the sand when approached, making it difficult or impossible to obtain a complete count (Grant and Doherty 2007).

A total of 68 individuals were captured 134 times. Exactly half of the individuals were recaptured exactly only once. This dataset is also included in the `secr` package.

References

Grant, T. J. and Doherty, P. F. (2007) Monitoring of the flat-tailed horned lizard with methods incorporating detection probability. *Journal of Wildlife Management* **71**, 1050–1056

See Also

secr

Examples

```
data(hornedlizard)
```

LBRecap — *Unconditional (complete) likelihood inference for capture-recapture analysis with emphasis on behavioural effect modelling*

Description

Unconditional (complete) likelihood inference for a large class of discrete-time capture-recapture models under closed population with special emphasis on behavioural effect modelling including also the *meaningful behavioral covariate* approach proposed in Alunni Fegatelli (2013) [PhD thesis]. Many of the standard classical models such as M_0, M_b, M_{c_1}, M_t or M_{bt} can be regarded as particular instances of the aforementioned approach. Other flexible alternatives can be fitted through a careful choice of a meaningful behavioural covariate and a possible partition of its admissible range

Usage

```
LBRecap ( data,last.column.count = FALSE, neval = 1000, startadd=0, by.incr = 1,
   mbc.function = c("standard","markov","counts","integer","counts.integer"),
   mod = c("linear.logistic", "M0", "Mb", "Mc", "Mcb", "Mt", "Msubjective.cut",
   "Msubjective"), heterogeneity=FALSE, markov.ord=NULL, z.cut=c(),
   meaningful.mat.subjective = NULL, meaningful.mat.new.value.subjective = NULL,
   td.cov = NULL, td.cov.formula ="", verbose = FALSE, graph = FALSE,
   output = c( "base", "complete" ) )
```

Arguments

`data` can be one of the following:

1. an M by t binary matrix/data.frame. In this case the input is interpreted as a matrix whose rows contain individual capture histories for all M observed units
2. a matrix/data.frame with $(t + 1)$ columns. The first t columns contain binary entries corresponding to capture occurrences, while the last column contains non negative integers corresponding to frequencies. This format is allowed only when `last.column.count` is set to `TRUE`

3. a t-dimensional array or table representing the counts of the 2^t contingency table of binary outcomes

M is the number of units captured at least once and t is the number of capture occasions.

last.column.count
: a logical. In the default case `last.column.count=FALSE` each row of the input argument `data` represents the complete capture history for each observed unit. When `last.column.count` is set to `TRUE` in each row the first t entries represent one of the observed complete capture histories and the last entry in the last column is the number of observed units with that capture history

neval
: a positive integer. `neval` is the number of values of the population size N where the posterior is evaluated starting from M. The default value is `neval`=1000.

startadd
: a positive integer. The likelihood evaluation is started from M+`startadd` that is from a value N which is strictly greater than the number of observed units. This can be useful when the likelihood has to be evaluated in a large range of N values and a reduced grid is called for.

by.incr
: a positive integer. `by.incr` represents the increment on the sequence of possible population sizes N where the posterior is evaluated. The default value is `by.incr`=1. The use of `by.incr`>1 is discouraged unless the range of N values of interest is very large

mbc.function
: a character string with possible entries (see Alunni Fegatelli (2013) for further details)

1. `"standard"` meaningful behavioural covariate in [0,1] obtained through the normalized binary representation of integers relying upon partial capture history
2. `"markov"` slight modification of `"standard"` providing consistency with arbitrary Markov order models when used in conjunction with the options `"Msubjective"` and `z.cut`.
3. `"counts"` covariate in [0,1] obtained by normalizing the integer corresponding to the sum of binary entries i.e. the number of previous captures
4. `"integer"` un-normalized integer corresponding to the binary entries of the partial capture history
5. `"counts.integer"` un-normalized covariate obtained as the sum of binary entries i.e. the number of previous captures

mod
: a character. `mod` represents the behavioural model considered for the analysis. `mod="linear.logistic"` is the model proposed in Alunni Fegatelli (2013) based on the meaningful behavioural covariate. `mod="M0"` is the most basic model where no effect is considered and all capture probabilities are the same. `mod="Mb"` is the classical behavioural model where the capture probability varies only once when first capture occurs. Hence it represents an *enduring* effect to capture. `mod="Mc"` is the *ephemeral* behavioural Markovian model originally introduced in Yang and Chao (2005) and subsequently extended in Farcomeni (2011) and reviewed in Alunni Fegatelli and Tardella (2012) where capture probability depends only on the capture status (captured or uncaptured) in the previous `k=markov.ord` occasions. `mod="Mcb"` is an extension of Yang and Chao's model (2005); it considers both *ephemeral* and *enduring* effect to

capture. `mod="Mt"` is the standard temporal effect with no behavioural effect. `mod="Msubjective.cut"` is an alternative behavioural model obtained through a specific cut on the meaningful behavioural covariate interpreted as memory effect. `mod="Msubjective"` is a customizable (subjective) behavioural model within the linear logistic model framework requiring the specification of the two additional arguments: the first one is `meaningful.mat.subjective` and contains an M by t matrix of ad-hoc meaningful covariates depending on previous capture history; the second one is `meaningful.mat.new.value.subjective` and contains a vector of length t corresponding to meaningful covariates for a generic uncaptured unit. The default value for `mod` is `"linear.logistic"`.

`heterogeneity` a logical. If `TRUE` individual heterogeneity effect is considered in the model

`markov.ord` a positive integer. `markov.ord` is the order of Markovian model M_c or M_{cb}. It is considered only if `mod="Mc"` or `mod="Mcb"`.

`meaningful.mat.subjective`
`M x t` matrix containing numerical covariates to be used for a customized logistic model approach

`meaningful.mat.new.value.subjective`
`1 x t` numerical vector corresponding to auxiliary covariate to be considered for unobserved unit

`z.cut` numeric vector. `z.cut` is a vector containing the cut point for the memory effect covariate. It is considered only if `mod="Msubjective.cut"`

`td.cov` data frame or matrix with k columns and t rows with each column corresponding to a time-dependent covariate to be used at each capture occasion for any captured/uncaptured unit

`td.cov.formula` a character string to be used as additional component in the `glm/glmer` formula. Names of each column of `td.cov` are forced to be `X1`, `X2`, ... See examples when covariates have to be considered as factors.

`verbose` a logical. If `TRUE` the percentage of likelihood evaluation is printed out while running.

`graph` a logical. If `TRUE` a plot with the likelihood evaluations is sent to the graphical device. This helps to verify the possible presence of an almost flat profile likelihood for N

`output` character. `output` select the kind of output from a very basic summary info on the posterior output (point and interval estimates for the unknown `N`) to more complete details including the estimates of the nuisance parameters and other features of the fitted model

Details

The `LBRecap` procedure is computing intensive for high values of `neval`.

Value

(if `output="complete"`) the function `LBRecap` returns a list of:

1. Modelmodel considered.

2. N.hatunconditional maximum likelihood estimate for N
3. CIinterval estimate for N
4. AICAkaike information criterion.
5. L.FailureLikelihood Failure condition
6. N.rangevalues of N considered.
7. log.likvalues of the log-likelihood distribution for each N considered
8. z.matrixmeaningful behavioural covariate matrix for the observed data
9. vec.cutcut point used to set up meaningful partitions the set of the partial capture histories according to the value of the value of the meaningful behavioural covariate.

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

Farcomeni A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika 98(1):237–242

See Also

`LBRecap.custom.part`, `LBRecap.all`, `BBRecap`

Examples

```
data(greatcopper)
mod.Mb=LBRecap(greatcopper,mod="Mb")
str(mod.Mb)
```

`LBRecap.all` *Standard behavioural and time effect models via unconditional (complete) likelihood approach*

Description

Comparative point and interval estimates for the population size N obtained fitting many alternative behavioural and time effect capture-recapture models. AIC index is reported for each alternative model.

Usage

```
LBRecap.all(data, last.column.count=FALSE, neval=1000, by.incr=1,
   which.mod=c("all","standard"), sort=c("default","AIC"))
```

Arguments

`data` can be one of the following:

1. an M by t binary matrix/data.frame
2. a matrix/data.frame with $(t+1)$ columns according to the value of `last.column.count`
3. a t-dimensional array or table representing the counts of the 2^t contingency table of binary outcomes M is the number of units captured at least once and t is the number of capture occasions.

`last.column.count`
: a logical. In the default case `last.column.count=FALSE` each row of `data` represents the complete capture history for each observed unit. When code-last.column.count=TRUE in each row the first t entries represent one of the possible observed complete capture histories and the last entry (last column) is the number of observed units with that capture history

`neval` a positive integer. `neval` is the number of alternative values of the population size N where the likelihood is evaluated and then maximized. They run from the minimum value M and they are increased by `by.incr` (see below the description of the `by.incr` argument). The default value is `neval`=1000.

`by.incr` a positive integer. `by.incr` represents the increment on the sequence of evaluated values for N. The default value is `by.incr`=1.

`which.mod` a character. `which.mod` selects which models are fitted and compared. In the default setting `which.mod="all"` all alternative models are fitted including new behavioural models based on alternative meaningful covariates (see Details). When `which.mod="standard"` the function only fits classical behavioural models with either enduring effects as in M_b, M_{c_1b}, M_{c_2b} or ephemeral effects as in purely Markovian M_{c_1} and M_{c_2}

`sort` character. `sort` selects the order of models.

Details

The available models are: M_0, M_b, M_t, M_{c_1}, M_{c_1b}, M_{c_2}, M_{c_2b}, M_{mc},$M_{mc_{int}}$, $M_{mc_{count}}$ and $M_{mc_{count.int}}$. This function `LBRecap.all` can be computing intensive for high values of `neval`.

Value

A dataframe with one row corresponding to each model and the following columns:

model: model considered

npar: number of parameters

AIC: Akaike's information criterion

Nhat: estimate of population size

Ninf: lower 95% confidence limit

Nsup: upper 95% confidence limit

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Alunni Fegatelli D. (2013) New methods for capture-recapture modelling with behavioural response and individual heterogeneity.

Alunni Fegatelli D., Tardella L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

Farcomeni A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika 98(1):237–242

Otis D. L., Burnham K. P., White G. C, Anderson D. R. (1978) Statistical Inference From Capture Data on Closed Animal Populations, Wildlife Monographs.

Yang H.C., Chao A. (2005) Modeling animals behavioral response by Markov chain models for capture-recapture experiments, Biometrics 61(4), 1010-1017

See Also

LBRecap,

Examples

```
## Not run:
data(greatcopper)
LBRecap.all(greatcopper)

## End(Not run)
```

LBRecap.custom.part — *Unconditional likelihood inference for behavioural effect models based on an ad-hoc partition of the set of all partial capture histories*

Description

Unconditional likelihood inference for a general model framework based on the capture probabilities conditioned on each possible partial capture history. As suggested in Alunni Fegatelli and Tardella (2012) the conditional approach originally proposed in Farcomeni (2011) [saturated reparameterization] is reviewed in terms of partitions into equivalence classes of conditional probabilities. In this function the user can directly provide the model as a partition.

Usage

```
LBRecap.custom.part (data,last.column.count=FALSE, partition, neval = 1000,
    by.incr = 1, output = c("base", "complete"))
```

Arguments

`data` can be one of the following:

1. an M by t binary matrix/data.frame
2. a matrix/data.frame with $(t+1)$ columns according to the value of `last.column.count`
3. a t-dimensional array or table representing the counts of the 2^t contingency table of binary outcomes M is the number of units captured at least once and t is the number of capture occasions.

`last.column.count` a logical. In the default case `last.column.count=FALSE` each row of `data` represents the complete capture history for each observed unit. When code-last.column.count=TRUE in each row the first t entries represent one of the possible observed complete capture histories and the last entry (last column) is the number of observed units with that capture history

`partition` list. `partition` represents a partition of the set of all partial capture histories.

`neval` a positive integer. `neval` is the number of values evaluated for the population size N. The default value is `neval`=1000.

`by.incr` a positive integer. `by.incr` represents the increment on the sequence of evaluated values for N. The default value is `by.incr`=1.

`output` character. `output` select the kind of output from a very basic summary info on the posterior output (point and interval estimates for the unknown `N`) to more complete details.

Details

The unconditional likelihood is evaluated by means of `glm/glmer` for each value of the `N` parameter and it is then maximized.

Value

(if `output="complete"`) the function `LBRecap` returns a list of:

1. N.hatunconditional maximum likelihood estimate for N
2. CIinterval estimate for N
3. pH.hatpoint estimate of nuisance parameters (conditional probabilities)
4. AICAkaike information criterion.
5. L.FailureLikelihood Failure condition
6. N.rangesequence of N values considered
7. log.likvalues of the log-likelihood distribution for each N considered
8. partitionslist of subsets of partial capture histories corresponding to equivalence classes

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

Farcomeni A. (2011) Recapture models under equality constraints for the conditional capture probabilities. Biometrika 98(1):237–242

See Also

`partition.ch`, `BBRecap.custom.part`, `LBRecap`

Examples

```
data(greatcopper)
partition.Mc1=partition.ch(quant.binary,t=ncol(greatcopper),breaks=c(0,0.5,1))
mod.Mc1=LBRecap.custom.part(greatcopper,partition=partition.Mc1)
str(mod.Mc1)
```

`list.historylabels` *List all the observable partial capture histories*

Description

This function returns a list of all the observable partial capture histories which can be recorded in a discrete-time capture-recapture setting with t consecutive trapping occasions. The observable partial capture histories are $2^t - 1$

Usage

```
list.historylabels(t,t.max=15)
```

Arguments

`t`	a positive integer representing the total number of trapping occasions
`t.max`	a positive integer representing upper bound on the total number of trapping occasions allowed.

Details

For obvious computing/memory reasons t is not allowed to be arbitrarily large. With `t.max=15` there are 32767 possible partial capture histories. If `t>t.max` the function stops with an error message.

Value

A list of all the observable partial capture histories which can be recorded in a discrete-time capture-recapture setting with t consecutive trapping occasions. If `t>t.max` the function stops with an error message.

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

See Also

partition.ch

Examples

```
list.historylabels(t=4)
```

---

lizard *Lizard data*

Description

The giant day gecko (Phelsuma madagascariensis grandis) is a tropical reptile naturally occurring in the northern part of Madagascar. Data represent an extensive capture-recapture experiment at the Masoala rainforest exhibit, Zurich Zoo. The researchers used the individual color patterns of this gecko species for photo recognition. Due to the high number of sampling sessions, this study is based on an unusually good dataset. Moreover, it is a captive population and, therefore, we can be sure that the closed population assumption is valid. However, the dataset is comparable to natural conditions because of the dimensions of the Masoala rainforest exhibit; it is currently the second largest tropical exhibit in the world. Additional variation in the dataset results from time-dependent variation in recapture rates (obvious dependence of gecko activity on daily weather) and the photographic capture method (it is harder to spot and photograph juvenile compared to adult geckos).

Usage

```
lizard
```

Format

A matrix with 68 rows (observed units) and 30 columns (capture occasions)

References

Wanger T.C. et al. (2009) How to monitor elusive lizards: comparison of capture-recapture methods on giant day geckos (Gekkonidae, Phelsuma madagascariensis grandis) in the Masoala rainforest exhibit, Zurich Zoo. Ecological Research 24(2):345–353.

Examples

```
data(lizard)
```

`mouse` *Mouse Dataset*

Description

Mouse (Microtus Pennsylvanicus) Dataset

Usage

```
mouse
```

Format

A matrix with 104 rows (observed animals) and 5 columns (capture occasions)

Details

The mouse (Microtus pennsylvanicus) data were first discussed in Nichols, Pollock and Hines (1984). The original live-trapping experiment was conducted monthly from June to December, 1980. During each month, the capture-recapture procedure was repeated for 5 consecutive days. The detailed data are given in Williams, Nichols and Conroy (2002, pp. 525-528). We use the data collected in June. A total of 104 distinct mice were caught in the experiment.

References

J. D. Nichols, K. H. Pollock and J. E. Hines, The Use of a Robust Capture-Recapture Design in Small Mammal Population Studies: A Field Example with Microtus Pennsylvanicus, Acta Theriologica, vol. 29. 30:357-365, 1984

Examples

```
data(mouse)
```

partition.ch — *Partition of partial capture histories according to equivalence classes of numerical quantification corresponding to supplied intervals*

Description

All the possible partial capture histories observable during a capture-recapture experiment with t sampling occasions can be partitioned according to numerical values corresponding to some meaningful covariate (quantification of binary sequences corresponding to partial capture histories). Each subset of the partition corresponds to all partial capture histories which returns numerical values of the quantification within one of the intervals represented by two consecutive values in the optional argument vector `breaks`.

Usage

```
partition.ch(quantify.ch.fun, t, breaks, include.lowest = T,
   type = c("list", "index"), ...)
```

Arguments

`quantify.ch.fun`	a function which returns a numerical value for each possible partial capture history
`t`	an integer. `t` is number of trapping occasions
`breaks`	a vector of numerical values which are used as bounds for the interval of numerical values corresponding to partial capture histories that belongs to the same partition
`include.lowest`	a logical, indicating if an x[i] equal to the lowest (or highest, when right = FALSE) breaks value should be included
`type`	a character string. It can be either `"list"` or `"index"`. See examples.
`...`	additional arguments to be passed to `quantify.ch.fun`

Details

It is useful in conjunction with `LBRecap.custom.part`. See examples.

Value

If the argument `type="list"` a list is returned. If `type="index"` a numerical index corresponding to the numeric integer equivalent of the consecutive interval according to the convention used in objects of class `factor`

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

See Also

LBRecap.custom.part, BBRecap.custom.part

Examples

```
data(mouse)
head(mouse)
t=ncol(mouse)

Mc1.partition=partition.ch(quantify.ch.fun=quant.binary,t=t,breaks=c(0,0.5,1))
Mc1.partition

mod.Mc1.cust=BBRecap.custom.part(mouse,partition=Mc1.partition)
mod.Mc1.cust

mod.Mc1.easy=BBRecap(mouse,mod="Mc",markov.ord=1,output="complete")

mod.Mc1.easy$N.hat.RMSE
mod.Mc1.easy$HPD.N
mod.Mc1.easy$log.marginal.likelihood

# the two functions give the same results!
```

---

pch *Partial Capture Histories from a Binary Data Matrix.*

Description

pch is used to obtain all the observed partial capture histories corresponding to an observed binary data matrix.

Usage

```
pch(data.matrix)
```

Arguments

data.matrix	a binary data matrix

Value

pch returns a matrix of mode "character" where each element represents the partial capture history associated to the respective element of the input binary data matrix.

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

See Also

`BBRecap.custom.part` and `LBRecap.custom.part`

Examples

```
data(greatcopper) # load greatcopper data
pch(greatcopper)
```

`quant.binary` *Quantification of binary capture histories*

Description

The `quant.binary` family of functions allow to quantify binary capture histories (partial or complete) in terms of a meaningful quantity which can be interpreted as a possibly meaningful behavioral covariate (like memory persistence of previous capture history)

Usage

```
quant.binary(x)
quant.binary.markov(x, markov.ord)
quant.binary.integer(x)
quant.binary.counts(x)
quant.binary.counts.integer(x)
```

Arguments

`x`	either a character string or a numeric vector exclusively made by binary entries 0 or 1.
`markov.ord`	a positive integer representing the order of the Markovian structure which one is willing to reproduce with suitable partition of the unit interval and the quantification of capture history standardized in the unit interval

Details

For a more detailed description of instances of meaningful behavioral covariates see Alunni Fegatelli and Tardella (2012) and Alunni Fegatelli (2013)[PhD Thesis]

Value

For `quant.binary` it returns a numeric value within the unit interval [0,1] and `quant.binary.markov`. For `quant.binary.integer` it returns an integer value .

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Alunni Fegatelli, D. and Tardella, L. (2012) Improved inference on capture recapture models with behavioural effects. Statistical Methods & Applications Applications Volume 22, Issue 1, pp 45-66 10.1007/s10260-012-0221-4

Alunni Fegatelli, D. (2013) New methods for capture-recapture modelling with behavioural response and individual heterogeneity. PhD Thesis. http://padis.uniroma1.it/bitstream/10805/2085/1/TesiDottorato-AlunniFegatelliDanilo.pdf

Examples

```
## Example of quantification with character input

capt.hist="0110"

quant.binary(capt.hist)
quant.binary.markov(capt.hist,markov.ord=2)
quant.binary.integer(capt.hist)
quant.binary.counts(capt.hist)
quant.binary.counts.integer(capt.hist)

## Example of quantification with numeric input

ch=c(0,1,1,0)

quant.binary(ch)
quant.binary.markov(ch,markov.ord=2)
quant.binary.integer(ch)
quant.binary.counts(ch)
quant.binary.counts.integer(ch)
```

---

rissanen — *Rissanen's universal prior for integers*

Description

It returns (up to normalizing constant) the mass assigned to each positive integer or a vector of integers by Rissanen's universal prior for positive integers

Usage

```
rissanen(n)
```

Arguments

n	a vector of positive integers

Details

Rissanen's universal prior on positive integers is one of the default options for eliciting a noninformative prior distribution on the unknown population size N. It is a proper prior with tails of the order between $1/N$ and $1/N^2$

Value

The mass assigned to each positive integer in the input vector of integers `n` by Rissanen's universal prior for positive integers

$$Q(n) = 2^{-\log^*(n)} \qquad n > 0$$

where $\log^*(x) = \log(x) + \log(\log(x)) + \log(\log(\log(x)))....$ where the sum involves only the non-negative terms. Notice that masses are not normalized hence they do not add to one but to a finite positive real constant

$$c = \sum_{n=1}^{\infty} Q(n)$$

Author(s)

Danilo Alunni Fegatelli and Luca Tardella

References

Rissanen, J. (1983) A universal prior for integers and estimation by minimum description length. Ann. Statist. 11, no. 2, 416-431

Examples

```
# Notice that masses are not normalized hence they do not add to one but to a finite
# positive real constant c

rissanen(1:5)
```

Index