

Package ‘extBatchMarking’

May 8, 2026

Type Package

Title Extended Batch Marking Models

Version 1.1.0

Date 2024-10-04

Maintainer Kehinde Olobatuyi <olobatuyikenny@uvic.ca>

Description

A system for batch-marking data analysis to estimate survival probabilities, capture probabilities, and enumerate the population abundance for both marked and unmarked individuals. The estimation of only marked individuals can be achieved through the `batchMarkOptim()` function. Similarly, the combined marked and unmarked can be achieved through the `batchMarkUnmarkOptim()` function. The algorithm was also implemented for the hidden Markov model encapsulated in `batchMarkUnmarkOptim()` to estimate the abundance of both marked and unmarked individuals in the population. The package is based on the paper: “Hidden Markov Models for Extended Batch Data” of Cowen et al. (2017) <[doi:10.1111/biom.12701](https://doi.org/10.1111/biom.12701)>.

License AGPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

LinkingTo Rcpp, RcppArmadillo

Imports doParallel, foreach, optimbase, Rcpp, parallel

Depends R (>= 4.0)

LazyData true

Suggests testthat (>= 3.0.0), knitr

Config/testthat/edition 3

URL https://github.com/Olobatuyi/extBatchMarking_cov

BugReports https://github.com/Olobatuyi/extBatchMarking_cov/issues

NeedsCompilation yes

Author Kehinde Olobatuyi [aut, cre] (ORCID:
<<https://orcid.org/0000-0002-4635-7895>>),
Simon Johns [aut],
Matthew RP Parker [aut] (ORCID:

<<https://orcid.org/0000-0003-3021-7959>>),
 Steve Hof [aut],
 Laura LE Cowen [aut] (ORCID: <<https://orcid.org/0000-0002-0853-1450>>)

Repository CRAN

Date/Publication 2024-10-04 23:40:02 UTC

Contents

batchLL	2
batchLogit	3
batchMarkHmmLL	3
batchMarkOptim	4
batchMarkUnmarkHmmLL	6
batchMarkUnmarkOptim	7
batchUnmark2Viterbi	11
batchUnmarkHmmLL	11
batchUnmarkViterbi	12
dbinpois	13
delta_g	13
gamma_gt	14
plot.batchMarkOptim	14
plot.batchMarkUnmarkOptim	15
print.batchMarkOptim	15
print.batchMarkUnmarkOptim	16
probs	16
WeatherLoach	17
Index	18

batchLL	<i>batchLL function provides the batch marking log-likelihood</i>
---------	---

Description

batchLL function provides the batch marking log-likelihood

Usage

```
batchLL(phi, p, R, begin_g, end_g)
```

Arguments

phi	The probability of surviving and remaining in the population between occasions t and t + 1, given an individual was alive and in the population at occasion t. This must be a number between 0 and 1.
p	The probability of capture at occasion t. This must be a number between 0 and 1.

R	The number of individuals marked and released at sampling occasion g from batch group g; $g = 1, 2, \dots, G$. This must be an integer.
begin_g	The beginning of the occasion.
end_g	The end of the occasion.

Value

fr returns the log sum of the Hidden Markov Model.

batchLogit	<i>batchLogit function</i>
------------	----------------------------

Description

'batchLogit' provides the number between 0 and 1.

Usage

```
batchLogit(x)
```

Arguments

x This is an input numerical value i.e double.

Value

Returns a number between 0 and 1.

batchMarkHmMLL	<i>Log-likelihood function for marked model.</i>
----------------	--

Description

This helps users check whether the function can be optimized at the given initial values before optimizing using [batchMarkOptim](#). After a quick check, if NAN or Inf is returned, the initial values should be revisited.

Usage

```
batchMarkHmMLL(
  par = NULL,
  data,
  covariate_phi = NULL,
  covariate_p = NULL,
  choiceModel = c("model1", "model2", "model3", "model4")
)
```

Arguments

par	Initial values for the parameters to be optimized over.
data	A capture-recapture data matrix or data frame.
covariate_phi	This covariate placeholder for the parameter phi_t
covariate_p	This covariate placeholder for the parameter p_t
choiceModel	This chooses among different models and allows for model selection

Value

Negative Log-likelihood value of the likelihood function

Examples

```
library(extBatchMarking)
# Initial parameter
theta <- c(0, -1)
res1 <- batchMarkHmMLL(par          = theta,
                       data         = WeatherLoach,
                       choiceModel  = "model4",
                       covariate_phi = NULL,
                       covariate_p  = NULL)

res1
```

batchMarkOptim	<i>Marked model only.</i>
----------------	---------------------------

Description

batchMarkOptim function optimizes [batchMarkHmMLL](#) function.

Usage

```
batchMarkOptim(
  par = NULL,
  data,
  covariate_phi = NULL,
  covariate_p = NULL,
  choiceModel = c("model1", "model2", "model3", "model4"),
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B"),
  lowerBound = -Inf,
  control,
  ...
)
```

Arguments

par	Initial values for the parameters to be optimized over.
data	A capture-recapture data matrix or data frame
covariate_phi	This covariate placeholder for the parameter phi_t
covariate_p	This covariate placeholder for the parameter p_t
choiceModel	This chooses among different models and allow for model selection
method	The method to be used. See optim for details.
lowerBound	Lower bounds on the variables for the "L-BFGS-B" method.
control	A list of control parameters. See optim for details.
...	Further arguments to be passed by user which goes into the optim function.

Details

Note that arguments after ... must be matched exactly. [batchMarkOptim](#) depends on [optim](#) function to optimize the parameters of the marked model only. By default [optim](#) performs minimization.

Value

For [batchMarkOptim](#), a list with components:

phi The survival probability and remaining in the population between occasion t and t+1.

p The capture probability at occasion time t.

ll The optimized log-likelihood value of marked model.

SE The standard error for each parameter.

AIC The Akaike Information Criteria for model selection.

References

Laura L. E. Cowen, Panagiotis Besbeas, Byron J. T. Morgan, 2017.: Hidden Markov Models for Extended Batch Data, *Biometrics*, 73, 1321-1331. DOI: 10.1111/biom.12701.

Examples

```
# Load the package
library(extBatchMarking)

# Load the WeatherLoach data from Cowen et al., 2017.
data(WeatherLoach)

# Initial parameter values
theta <- c(0, -1)

mod1 <- batchMarkOptim(
  par      = theta,
  data     = WeatherLoach,
  choiceModel = "model4",
```

```

        method      = "BFGS",
        control      = list(trace = 1),
        covariate_phi = NULL,
        covariate_p  = NULL)

# print(mod1)

# Survival probability
mod1$phi
# Capture probability
mod1$p
# Optimized log-likelihood
mod1$ll
# The Akaike Information Criteria
mod1$AIC

mod2 <- batchMarkOptim(
  par      = theta,
  data     = WeatherLoach,
  choiceModel = "model4",
  method   = "L-BFGS-B",
  control   = list(trace = 1),
  covariate_phi = NULL,
  covariate_p  = NULL)

# print(mod2)
# Survival probability
mod2$phi
# Capture probability
mod2$p
# Optimized log-likelihood
mod2$ll
# The Akaike Information Criteria
mod2$AIC

```

batchMarkUnmarkHmmLL *Log-likelihood function for combined model.*

Description

This helps users check whether the function can be optimized at the given initial values before optimizing using [batchMarkUnmarkOptim](#). After a quick check, if NAN or Inf is returned, the initial values should be revisited.

Usage

```
batchMarkUnmarkHmmLL(
```

```

  par,
  data,
  Umax,
  nBins,
  covariate_phi = NULL,
  covariate_p = NULL,
  choiceModel = c("model1", "model2", "model3", "model4")
)

```

Arguments

par	Initial values for the parameters to be optimized over.
data	A capture-recapture data matrix or data frame.
Umax	The maximum number of the unmarked individuals in the population for capture on any occasion.
nBins	The number of bin size into which the matrix will be divided.
covariate_phi	This covariate placeholder for the parameter phi_t
covariate_p	This covariate placeholder for the parameter p_t
choiceModel	This chooses among different models and allow for model selection.

Value

Negative Log-likelihood value of the likelihood function.

Examples

```

library(extBatchMarking)
theta <- c(0.1, 0.1, 7, -1.5)
res3 <- batchMarkUnmarkHmMLL(par      = theta,
                             data      = WeatherLoach,
                             choiceModel = "model4",
                             Umax      = 1800,
                             nBins     = 600,
                             covariate_phi = NULL,
                             covariate_p  = NULL)

res3

```

batchMarkUnmarkOptim *Combined Marked and Unmarked models.*

Description

batchMarkUnmarkOptim function optimizes [batchMarkUnmarkHmMLL](#) function.

Usage

```
batchMarkUnmarkOptim(
  par = NULL,
  data,
  choiceModel = c("model1", "model2", "model3", "model4"),
  covariate_phi = NULL,
  covariate_p = NULL,
  method = c("Nelder-Mead", "BFGS", "CG", "L-BFGS-B"),
  Umax = 1800,
  nBins = 20,
  popSize = c("Horvitz_Thompson", "Model-Based"),
  lowerBound = -Inf,
  control,
  ...
)
```

Arguments

par	Initial values for the parameters to be optimized over.
data	A capture-recapture data matrix or data frame
choiceModel	This chooses among different models and allow for model selection
covariate_phi	This covariate placeholder for the parameter phi_t
covariate_p	This covariate placeholder for the parameter p_t
method	The method to be used. See <code>optim</code> for details.
Umax	The maximum number of the unmarked individuals in the population for capture on any occasion.
nBins	The number of bin size into which the matrix will be divided.
popSize	The Horvitz_Thompson method or Model-Based to compute population size.
lowerBound	Lower bounds on the variables for the "L-BFGS-B" method.
control	a list of control parameters. See <code>optim</code> for details.
...	Further arguments to be passed by user which goes into the <code>optim</code> function.

Details

Note that arguments after ... must be matched exactly.

`batchMarkUnmarkOptim` depends on `optim` function to optimize the parameters of the combined model. By default `optim` performs minimization.

Example on `Umax` and `nBins`: `Umax = 1800` has a matrix of 1801 x 1801 and `nBins = 20`, reduces the matrix to 90 x 90. This is done in Cowen et al., 2017 to reduce the computing time when dealing with large matrix.

Value

A list of the following optimized parameters will be returned.

phi The survival probability and remaining in the population between occasion t and $t+1$.

p The capture probability at occasion time t .

ll The optimized log-likelihood value of marked model.

SE The standard error for each parameter.

AIC The Akaike Information Criteria for model selection.

lambda Initial mean abundance at occasion $t = 1$.

gam Recruitment rate of individual into the unmarked population.

M Total number of marked individual in the population.

U Total number of unmarked individuals in the population available for capture at occasion $t = 1, \dots, T$.

N Total population size at time $t = 1, \dots, T$.

References

Laura L. E. Cowen, Panagiotis Besbeas, Byron J. T. Morgan, 2017.: Hidden Markov Models for Extended Batch Data, *Biometrics*, 73, 1321-1331. DOI: 10.1111/biom.12701.

Examples

```
# Load the package
library(extBatchMarking)

# Load the WeatherLoach data from Cowen et al., 2017.
data(WeatherLoach)

# Initial parameter values
theta <- c(0.1, 0.1, 7, -1.5)

mod1 <- batchMarkUnmarkOptim(
  par      = theta,
  data     = WeatherLoach,
  Umax    = 1800,
  nBins   = 600,
  covariate_phi = NULL,
  covariate_p  = NULL,
  choiceModel = "model4",
  popSize    = "Horvitz_Thompson",
  method     = "CG",
  control    = list(trace = 1))

# Survival probability
mod1$phi
# Capture probability
mod1$p
```

```

# Optimized log-likelihood
mod1$l1
# The Akaike Information Criteria
mod1$AIC
# The initial mean abundance
mod1$lambda
# Recruitment rate into the population
mod1$gam
# The estimated abundance of unmarked animals
mod1$U
# The estimated abundance of marked animals
mod1$M
# The estimated total abundance of marked and unmarked animals
mod1$N

```

```

mod2 <- batchMarkUnmarkOptim(
  par      = theta,
  data     = WeatherLoach,
  Umax     = 1800,
  nBins    = 600,
  choiceModel = "model4",
  covariate_phi = NULL,
  covariate_p  = NULL,
  popSize     = "Model-Based",
  method      = "L-BFGS-B",
  control     = list(trace = 1))

# print(mod2)
# plot(mod2)
# Survival probability
mod2$phi
# Capture probability
mod2$p
# Optimized log-likelihood
mod2$l1
# The Akaike Information Criteria
mod2$AIC
# The initial mean abundance
mod2$lambda
# Recruitment rate into the population
mod2$gam
# The estimated abundance of unmarked animals
mod2$U
# The estimated abundance of marked animals
mod2$M
# The estimated total abundance of marked and unmarked animals
mod2$N

```

batchUnmark2Viterbi *batchUnmark2Viterbi function provides a wrapper for the batchUnmarkViterbi to compute the population abundance*

Description

batchUnmark2Viterbi function provides a wrapper for the batchUnmarkViterbi to compute the population abundance

Usage

```
batchUnmark2Viterbi(
  par,
  data,
  Umax,
  nBins,
  choiceModel = c("model1", "model2", "model3", "model4")
)
```

Arguments

par	Initial values for the parameters to be optimized over.
data	A capture-recapture data matrix or data frame.
Umax	The maximum number of the unmarked individuals in the population for capture on any occasion.
nBins	The number of bin size into which the matrix will be divided.
choiceModel	This chooses among different models and allows for model selection

Value

Negative Log-likelihood value of the likelihood function

batchUnmarkHmmLL *batchUnmarkHmmLL function provides the unmarked function to be optimized*

Description

batchUnmarkHmmLL function provides the unmarked function to be optimized

Usage

```
batchUnmarkHmmLL(phi, p, lambda, gam, Umax, nBins, u)
```

Arguments

phi	The probability of surviving and remaining in the population between occasions t and $t+1$, given an individual was alive and in the population at occasion t . This must be a number between 0 and 1.
p	The probability of capture at occasion t . This must be a number between 0 and 1.
lambda	The initial mean abundance (at occasion 1) for the unmarked population.
gam	The recruitment rate into the unmarked population.
Umax	The maximum number of the unmarked individuals in the population for capture on any occasion.
nBins	The number of bin size into which the matrix will be divided.
u	The number of individuals captured at sampling occasion t that were not marked; $t = 1, \dots, T$.

Value

Negative Log-likelihood value of the likelihood function

batchUnmarkViterbi *batchUnmarkViterbi function provides the implementation of the Viterbi algorithm for the unmarked model*

Description

batchUnmarkViterbi function provides the implementation of the Viterbi algorithm for the unmarked model

Usage

```
batchUnmarkViterbi(phi, p, lambda, gam, Umax, nBins, u)
```

Arguments

phi	The probability of surviving and remaining in the population between occasions t and $t+1$, given an individual was alive and in the population at occasion t . This must be a number between 0 and 1.
p	The probability of capture at occasion t . This must be a number between 0 and 1.
lambda	the initial mean abundance (at occasion 1) for the unmarked population.
gam	The recruitment rate into the unmarked population
Umax	The maximum number of the unmarked individuals in the population for capture on any occasion.
nBins	The number of bin size into which the matrix will be divided.
u	The number of individuals captured at sampling occasion t that were not marked; $t = 1, \dots, T$.

Value

Negative Log-likelihood value of the likelihood function

dbinpois	<i>Convolution of Poisson and Binomial for Batch</i>
----------	--

Description

This is the convolution of Poisson and Binomial distributions

Usage

```
dbinpois(z, n, par)
```

Arguments

z	This is the vector of numerical values
n	The nrow of capture-recapture data matrix or data frame
par	This is the vector of parameter values: average from Poisson distribution and probability of success from Binomial distribution

Details

The convolution of Poisson and Binomial distribution helps us to compute the number of individuals that have survived from t-1 to t in the combined model while simultaneously computing the number of individuals recruited into the population at occasion t.

The survival is modeled as Binomial distribution and the recruitment as the Poisson distribution

Value

f This is the output of the convolution from the Binomial and Poisson distributions

delta_g	<i>initial probability function</i>
---------	-------------------------------------

Description

initial probability function

Usage

```
delta_g(R)
```

Arguments

R The number of individuals marked and released at sampling occasion g from batch group g ; $g = 1, 2, \dots, G$. This must be an integer.

Value

A vector of initial value with 1 at the observed position

gamma_gt	<i>Transition State Probability 'gamma_gt' computes the transition probability matrix</i>
----------	---

Description

Transition State Probability 'gamma_gt' computes the transition probability matrix

Arguments

R integer number of marked individuals released per occasion
 phi double number. Survival probability of individuals
 cores The number of cores on your machine.

Value

pR Returns the transition matrix

plot.batchMarkOptim	<i>Plot Method for batchMarkUnmarkOptim Objects</i>
---------------------	---

Description

This function defines how objects of class "Employee" are printed.

Usage

```
## S3 method for class 'batchMarkOptim'
plot(x, ...)
```

Arguments

x An object of class "Employee".
 ... Additional arguments passed to the print method.

`plot.batchMarkUnmarkOptim`

Plot Method for batchMarkUnmarkOptim Objects

Description

This function defines how objects of class "Employee" are printed.

Usage

```
## S3 method for class 'batchMarkUnmarkOptim'  
plot(x, ...)
```

Arguments

`x` An object of class "Employee".
`...` Additional arguments passed to the print method.

`print.batchMarkOptim` *Print Method for batchMarkOptim Objects*

Description

This function defines how objects of class "Employee" are printed.

Usage

```
## S3 method for class 'batchMarkOptim'  
print(x, ...)
```

Arguments

`x` An object of class "Employee".
`...` Additional arguments passed to the print method.

```
print.batchMarkUnmarkOptim
```

Print Method for batchMarkUnmarkOptim Objects

Description

This function defines how objects of class "Employee" are printed.

Usage

```
## S3 method for class 'batchMarkUnmarkOptim'
print(x, ...)
```

Arguments

x	An object of class "Employee".
...	Additional arguments passed to the print method.

```
probs
```

State-dependent probability function

Description

'probs' computes the state-dependent transition matrix

Usage

```
probs(r, p, R)
```

Arguments

r	The number of individuals from batch group "g" recaptured at recapture occasion t; $g = 1, 2, \dots, G$, $t = g+1, \dots, T$. This must be an integer.
p	The probability of capture at occasion t. This must be a number between 0 and 1.
R	The number of individuals marked and released at sampling occasion g from batch group g; $g = 1, 2, \dots, G$. This must be an integer.

Value

PR diagonal matrix of the state-dependent probability.

WeatherLoach

Weather Loach data

Description

Data from marked individuals captured on multiple occasions. The weather-loach study was described in detail by Huggin (). Different colored batch tags were given to a random sample of unmarked individuals at each occasion.

Usage

WeatherLoach

Format**'Weather_loach':**

A data frame with 10 rows indicating number of captures and 11 columns indicating recaptures

Weather Loach Data

Index

* datasets

WeatherLoach, [17](#)

[batchLL](#), [2](#)

[batchLogit](#), [3](#)

[batchMarkHmmLL](#), [3](#), [4](#)

[batchMarkOptim](#), [3](#), [4](#), [5](#)

[batchMarkUnmarkHmmLL](#), [6](#), [7](#)

[batchMarkUnmarkOptim](#), [6](#), [7](#)

[batchUnmark2Viterbi](#), [11](#)

[batchUnmarkHmmLL](#), [11](#)

[batchUnmarkViterbi](#), [12](#)

[dbinpois](#), [13](#)

[delta_g](#), [13](#)

[gamma_gt](#), [14](#)

[optim](#), [5](#), [8](#)

[plot.batchMarkOptim](#), [14](#)

[plot.batchMarkUnmarkOptim](#), [15](#)

[print.batchMarkOptim](#), [15](#)

[print.batchMarkUnmarkOptim](#), [16](#)

[probs](#), [16](#)

[WeatherLoach](#), [17](#)