Package 'dupiR'

February 19, 2015

Type Package

Title Bayesian inference from count data using discrete uniform priors

Version 1.2

Date 2014-12-29

Depends R (>= 2.15.1), methods, plotrix

Author Federico Comoglio and Maurizio Rinaldi

Maintainer Federico Comoglio <federico.comoglio@bsse.ethz.ch>

Description Inference of population sizes using a binomial likelihood and least informative discrete uniform priors.

License GPL-2

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2014-12-29 18:15:37

R topics documented:

dupiR-package 2
computePosterior 2
Counts 4
getCounts 6
getPosteriorParam 6
newCounts 7
plotPosterior 8

Index **10**

dupiR-package *Bayesian inference using discrete uniform priors with R*

Description

This package implements a Bayesian approach to infer population sizes from count data. The package takes a set of sample counts obtained by sampling fractions of a finite volume containing an homogeneously dispersed population of identical objects and returns the posterior probability distribution of the population size. The algorithm makes use of a binomial likelihood and non-conjugate, discrete uniform priors. dupiR can be applied to both sampling with or without replacement.

Details

Package:	dupiR
Type:	Package
Version:	1.2
Date:	2014-12-29
License:	GPL-2

Author(s)

Federico Comoglio (1) and Maurizio Rinaldi (2)

(1) Department of Biosystems Science and Engineering (D-BSSE), ETH Zurich, Switzerland

(2) Dipartimento di Scienze del Farmaco, Universita' del Piemonte Orientale

Maintainer: Federico Comoglio

<federico.comoglio@bsse.ethz.ch>

References

Comoglio F, Fracchia L and Rinaldi M (2013) Bayesian inference from count data using discrete uniform priors. PLOS ONE, 8(10):e74388. doi:10.1371/journal.pone.0074388

computePosterior *Compute the posterior probability distribution of the population size*

Description

Compute the posterior probability distribution of the population size using a discrete uniform prior and a binomial likelihood (DUP method). When applicable, an approximation using a Gamma prior and a Poisson likelihood is used instead (GP method, see Clough et al).

Usage

```
computePosterior(object, n1, n2, replacement = FALSE, b, alg = "DUP")
```

Arguments

object	An object of class 'Counts'
n1	Left endpoint of the prior support interval (Optional). If not provided and total counts are not zero, computed using the maximum likelihood estimate (mle) of the population size as 0.5 * mle
n2	Right endpoint of the prior support interval (Optional). If not provided and total counts are not zero, computed using the maximum likelihood estimate (mle) of the population size as 2 * mle
replacement	Whether sampling has been performed with replacement. Default to FALSE.
b	Prior rate parameter of the gamma distribution used to compute the posterior with Clough. Default is 1e-10.
alg	Algorithm to be used to perform computations. Default to DUP.

Methods

```
signature(object = "Counts")
```

an object of class `Counts`.

Value

Returns an object of class `Counts`.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Comoglio F, Fracchia L and Rinaldi M (2013) Bayesian inference from count data using discrete uniform priors. PLOS ONE, to appear

Clough HE et al. (2005) Quantifying Uncertainty Associated with Microbial Count Data: A Bayesian Approach. Biometrics 61: 610-616

See Also

`Counts`, `plotPosterior`

Examples

```
K <- newCounts( counts = c(20,30), fractions = c(0.075, 0.10))

#using default parameters (DUP, sampling without replacement and default prior support)
K.dup <- computePosterior(K)
```

```
#using custom prior support (DUP)
K.cust <- computePosterior(K, n1 = 0, n2 = 1e3)

#using a Gamma prior (GP method)
K.gp <- computePosterior(K, alg = 'GP')

#plot the results (compare DUP with GP)
plotPosterior(K.dup, type = 'l', lwd = 3, col = 'blue3', low = 0.025, up = 0.975)
lines(K.gp@posterior, lwd = 3, col = 'red3')

#for sampling with replacement:
computePosterior(K, replacement = TRUE)
```

`Counts` *Class "Counts" – a container for measurements and dupiR inference results*

Description

Definition of an object of this class requires a set of measurements, i.e. a collection of counts and sampling fractions. Inference of the posterior distribution by dupiR (`computePosterior`) and subsequent call to `getPosteriorParam` will fill all additional slots.

Objects from the Class

Objects should be created with calls to `newCounts`.

Slots

`counts`: Object of class `"integer"`. An integer vector of positive numbers (counts). Mandatory for object initialization.

`fractions`: Object of class `"numeric"`. A numeric vector of the corresponding sampling fractions (real numbers in (0,1]). Mandatory for object initialization.

`n1`: Object of class `"numeric"`. Left endpoint of the prior support interval. If not provided and total counts are not zero, computed using the maximum likelihood estimate (mle) of the population size as 0.5 * mle.

`n2`: Object of class `"numeric"`. Right endpoint of the prior support interval. If not provided and total counts are not zero, computed using the maximum likelihood estimate (mle) of the population size as 2 * mle.

`X`: Object of class `"numeric"`. The product of (1 - fractions).

`mle`: Object of class `"numeric"`. The maximum likelihood estimate of the population size, computed as K/R, where K is the total counts and R is the total sampling fraction.

`nconst`: Object of class `"numeric"`. The normalization constant (see Corollary 1 in the reference for details).

`posterior`: Object of class `"ANY"`. A vector of posterior probabities over the prior support. It containes either the PMF or a logical value used to obtain posterior parameters with a Gamma approximation (see reference for details).

`map.p`: Object of class `"numeric"`. The maximum posterior probability.

`map.idx`: Object of class `"numeric"`. The index of the prior support corresponding to the maximum a posteriori (MAP), i.e. MAP - n1 + 1.

`map`: Object of class `"numeric"`. The MAP of the population size.

`qlow.p`: Object of class `"numeric"`. The probability of the left endpoint (q1) of the credible interval. Default confidence level 95%.

`qlow.idx`: Object of class `"integer"`. The index of the prior support corresponding to q1.

`qlow`: Object of class `"numeric"`. The left endpoint (q1) of the credible interval.

`qlow.cum`: Object of class `"numeric"`. The cumulative posterior probability from n1 to q1, i.e. the left tail.

`qup.p`: Object of class `"numeric"`. The probability of the right endpoint (q2) of the credible interval. Default confidence level 95%.

`qup.idx`: Object of class `"integer"`. The index of the prior support corresponding to q2.

`qup`: Object of class `"numeric"`. The right endpoint (q2) of the credible interval.

`qup.cum`: Object of class `"numeric"`. The cumulative posterior probability from q2 to n2, i.e. the right tail.

`gamma`: Object of class `"logical"`. TRUE if the posterior was computed using a Gamma approximation (see reference for details).

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Comoglio F, Fracchia L and Rinaldi M (2013) Bayesian inference from count data using discrete uniform priors. PLOS ONE, to appear

See Also

`newCounts`

Examples

```
# create an object of class 'Counts' by using new
new('Counts', counts = c(30, 35), fractions = c(0.075, 0.1))

#or by means of the constructor
newCounts(counts = c(30, 35), fractions = c(0.075, 0.1))
```

getCounts *Accessors for the 'counts' and 'fractions' slots of a Counts object.*

Description

Each measurement consists of an integer count and a corresponding sampling fraction. These values are required to defined an object of class `Counts` and are subsequently stored in the counts and fractions slots. The counts slot is an integer vector of counts. The fractions slot is a numeric vector of matched sampling fractions.

Methods

`signature(object = "Counts")` an object of class Counts.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

`Counts`

Examples

```
K <- newCounts( counts = c(20,30), fractions = c(0.075, 0.1))

getCounts(K)
getFractions(K)
```

getPosteriorParam *Compute posterior probability distribution parameters*

Description

Obtain statistical parameters from the posterior probability distribution. Particularly, this function computes credible intervals at a given confidence level (default to 95%).

Usage

```
getPosteriorParam(object, low = 0.025, up = 0.975, ...)
```

Arguments

`object`	An object of class 'Counts'
`low`	The left tail posterior probability
`up`	1 - the right tail posterior probability
`...`	Additional arguments, for compatibility with `plotPosterior`

Value

An object of class Counts.

Methods

signature(object = "Counts")

an object of class Counts.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

Counts, plotPosterior

Examples

```
K <- newCounts( counts = c(20,30), fractions = c(0.075, 0.10))

#using default parameters (DUP, sampling without replacement and default prior support)
K.dup <- computePosterior(K)

getPosteriorParam(K.dup)
```

---

newCounts *Construct an object of class Counts*

Description

Construct an object of class Counts

Usage

```
newCounts(counts, fractions)
```

Arguments

counts	An integer vector of positive numbers (counts). Mandatory for object initialization.
fractions	A numeric vector of the corresponding sampling fractions (real numbers in (0,1]). Mandatory for object initialization.

Value

Returns an object of class Counts.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

References

Comoglio F, Fracchia L and Rinaldi M (2013) Bayesian inference from count data using discrete uniform priors. PLOS ONE, to appear

See Also

Counts

Examples

```
K <- newCounts( counts = c(20,30), fractions = c(0.075, 0.10))
K
#or (a little bit more informative):
summary(K)
```

---

plotPosterior *Plot posterior probability distributions*

Description

Produces publication-level plots of posterior probability distributions computed using `computePosterior`. A data summary, credible intervals at a given confidence level, maximum a posteriori (and more) are indicated.

Usage

```
plotPosterior(object, low = 0.025, up = 0.975, xlab, step, ...)
```

Arguments

`object`	An object of class 'Counts'
`low`	The left tail posterior probability. Default to 0.025.
`up`	1 - the right tail posterior probability. Default to 0.975.
`xlab`	The x-axis label. If not provided, default to 'n'.
`step`	An integer defining the increment for x-axis labels (i.e. the separation between two consecutive tick marks).
`...`	Additional arguments to be passed to the `plot` function

Value

Called for its effects.

Methods

`signature(object = "Counts")` an object of class Counts.

Author(s)

Federico Comoglio, federico.comoglio@bsse.ethz.ch

See Also

Counts-class, computePosterior

Examples

```
K <- newCounts( counts = c(20,30), fractions = c(0.075, 0.10))

#using default parameters (DUP, sampling without replacement and default prior support)
K.dup <- computePosterior(K)

plotPosterior(K.dup, type = 'l', lwd = 3, col = 'blue3', low = 0.025, up = 0.975)
```

Index

∗Topic **class**
Counts, 4
∗Topic **functions**
newCounts, 7
∗Topic **methods**
computePosterior, 2
getCounts, 6
getPosteriorParam, 6
plotPosterior, 8
∗Topic **package**
dupiR-package, 2

computePosterior, 2, *9*
computePosterior,Counts-method *(*computePosterior*)*, 2
computePosterior-methods *(*computePosterior*)*, 2
Counts, *3*, 4, *6–8*
Counts-class *(*Counts*)*, 4

dupiR *(*dupiR-package*)*, 2
dupiR-package, 2

getCounts, 6
getCounts,Counts-method *(*getCounts*)*, 6
getCounts-methods *(*getCounts*)*, 6
getFractions *(*getCounts*)*, 6
getFractions,Counts-method *(*getCounts*)*, 6
getFractions-methods *(*getCounts*)*, 6
getPosteriorParam, 6
getPosteriorParam,Counts-method *(*getPosteriorParam*)*, 6
getPosteriorParam-methods *(*getPosteriorParam*)*, 6

newCounts, *4, 5*, 7

plot,Counts-method *(*plotPosterior*)*, 8
plot-method *(*plotPosterior*)*, 8
plotPosterior, *3*, *7*, 8
plotPosterior,Counts-method *(*plotPosterior*)*, 8
plotPosterior-methods *(*plotPosterior*)*, 8

summary,Counts-method *(*Counts*)*, 4