Package ‘babar’

February 24, 2015

Type Package

Title Bayesian Bacterial Growth Curve Analysis in R

Version 1.0

Date 2014-11-13

Author Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Maintainer Matthew Hartley <Matthew.Hartley@jic.ac.uk>

Description Babar is designed to use nested sampling (a Bayesian analysis technique) to compare possible models for bacterial growth curves, as well as extracting parameters. It allows model evidence and parameter likelihood values to be extracted, and also contains helper functions for comparing distributions as well as direct access to the underlying nested sampling code.

License GPL-2

Suggests knitr, R.rsp

VignetteBuilder knitr, R.rsp

NeedsCompilation no

Repository CRAN

Date/Publication 2015-02-24 23:27:07

R topics documented:

`babar-package` *Bayesian Bacterial Growth Curve Analysis in R*

Description

Babar is designed to use nested sampling (a Bayesian analysis technique) to compare possible models for bacterial growth curves, as well as extracting parameters. It allows model evidence and parameter likelihood values to be extracted, and also contains helper functions for comparing distributions as well as direct access to the underlying nested sampling code.

Details

Package:	babar
Type:	Package
Title:	Bayesian Bacterial Growth Curve Analysis in R
Version:	1.0
Date:	2014-11-13
Author:	Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen
Maintainer:	Matthew Hartley <Matthew.Hartley@jic.ac.uk>
License:	GPL-2
Suggests:	knitr, R.rsp
VignetteBuilder:	knitr, R.rsp

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

`Bayescompare` *Bayescompare*

Description

Perform Bayesian analysis for comparing two bacterial growth curves using the Baranyi model.

Usage

```
Bayescompare(data1, data2, hyp, model, inf.sigma1 = TRUE, inf.sigma2 = TRUE,
    inc.nd1 = FALSE, inc.nd2 = FALSE, sigma1 = 0.3, sigma2 = 0.3,
    threshold1 = NULL, threshold2 = NULL, mumax.prior1 = "Uniform",
    mumax.prior2 = "Uniform", mu.mean1 = NULL, mu.mean2 = NULL,
    mu.sd1 = NULL, mu.sd2 = NULL, tol = 0.1, prior.size = 250)
```

Arguments

`data1`

`data2` Datafiles of the two curves to be fitted. This should consist of two columns, the first for time and second for logc. The bacterial concentration should be given in log_10 cfu and there should be at least 2 data points (the first of which may be undetected). Undetected y values should be represented by "NA".

`hyp` Hypothesis to test. This should be one of "H1" (data curves replicates), "H2" (data curves have same growth rate) and "H3" (all data curve parameters are different)

`model` The growth model to be used. This should be one of "linear", "logistic", "Bar3par" and "Bar4par".

`inf.sigma1`

`inf.sigma2` (TRUE/FALSE) Choose whether or not to infer the noise levels, sigma1 (for curve 1) and sigma2 (for curve 2), as part of the analysis. If FALSE, sigma should be specified (or the default value of sigma, 0.3, will be used).

`inc.nd1`

`inc.nd2` Choose whether or not to include undetected points for curves 1 and 2 respectively as part of the analysis. If TRUE, threshold should be specified.

`sigma1`

`sigma2` The choice of noise levels, sigma1 and sigma2, in log_10 cfu if not inferred as part of the analysis. Default is 0.3.

`threshold1`

`threshold2` Thresholds in log_10 cfu below which values are considered as undetected.

`mumax.prior1`

`mumax.prior2` The type of priors to use for mu_max1 and mu_max2. These should be one of "Uniform", "Gaussian" or "Cauchy" (or the default "Uniform" will be used). If "Gaussian" or "Cauchy" are specified for either, mu.mean and mu.sd should be given.

`mu.mean1`

`mu.mean2` The means to be used when using a Gaussian or Cauchy prior.

`mu.sd1`

`mu.sd2` The standard deviations to be used when using a Gaussian or Cauchy prior.

`tol` The termination tolerance for nested sampling

`prior.size` The number of prior samples to use for nested sampling

Value

Returns:

posterior: The samples from the posterior, together with their log weights and log likelihoods as a m x n matrix, where m is the number of posterior samples and n is the number of parameters + 2. The log weights are the first column and the log likelihood values are the second column of this matrix. The sum of the log-weights = logZ.

logevidence: The logarithm of the evidence, a scalar.

means: A vector of the mean of each parameter, length = no. of parameters.

vars: A vector of the variance of each parameter, length = no. of parameters.

equalposterior: Equally weighted posterior samples together with their log likelihoods as a m x n matrix, where m is the number of posterior samples and n is the number of parameters + 1. The log likelihood values are the first column of this matrix.

fit.t1,fit.t2: Vectors of time points at which the model is fitted for data1 and data2 respectively.

fit.y1,fit.y2: Matrices of fitted model points, y1 (for data1) and y2 (for data 2), using posterior parameter samples in the model. Each column represents a different posterior sample.

fit.y1mean and fit.y2mean: Vectors of fitted model points, y1 and y2, using the mean of the posterior parameter samples in the model.

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
LmH_411.file <- system.file("extdata", "LmH_411.csv", package = "babar")
LmH_411.data <- read.csv(LmH_411.file, header=TRUE, sep =",",
                         na.strings=c("ND","NA"))
M126_50.file <- system.file("extdata", "M126_50.csv", package = "babar")
M126_50.data <- read.csv(M126_50.file, header=TRUE, sep =",",
                         na.strings=c("ND","NA"))

# Get a quick approximation of the evidence/model parameters.
results.linear.short <- Bayescompare(LmH_411.data, M126_50.data, hyp="H1",
                                     model="linear",tol=100, prior.size=25)

# Compute a better estimate of the evidence/model parameters (slow so not
# run as part of the automated examples).
## Not run:
results.linear.full <- Bayescompare(LmH_411.data, M126_50.data, hyp="H1", model="linear")

## End(Not run)
```

Bayesfit *Bayesfit*

Description

Perform Bayesian analysis for fitting a single bacterial growth curve using the Baranyi model.

Usage

```
Bayesfit(data, model, inf.sigma = TRUE, inc.nd = FALSE, sigma = 0.3,
    threshold = NULL, mumax.prior = "Uniform", mu.mean = NULL,
    mu.sd = NULL, tol = 0.1, prior.size = 100)
```

Arguments

`data` A datafile of the curve to be fitted. This should consist of two columns, the first for time and second for logc. The bacterial concentration should be given in log_10 cfu and there should be at least 2 data points (the first of which may be undetected). Undetected y values should be represented by "NA".

`model` The growth model to be used. This should be one of "linear", "logistic", "Bar3par", "Bar4par" and "Bar6par".

`inf.sigma` (TRUE/FALSE) Choose whether or not to infer the noise level, sigma, as part of the analysis. If FALSE, sigma should be specified (or the default value of sigma, 0.3, will be used).

`inc.nd` Choose whether or not to include undetected points as part of the analysis. If TRUE, threshold should be specified.

`sigma` The choice of noise level, sigma, in log_10 cfu if it is not inferred as part of the analysis. Default is 0.3.

`threshold` Threshold in log_10 cfu below which values are considered as undetected.

`mumax.prior` The type of prior to use for mu_max. This should be one of "Uniform", "Gaussian" or "Cauchy" (or the default "Uniform" will be used). If "Gaussian" or "Cauchy" are specified, mu.mean and mu.sd should be given.

`mu.mean` The mean to be used when using a Gaussian or Cauchy prior.

`mu.sd` The standard deviation to be used when using a Gaussian or Cauchy prior.

`tol` The termination tolerance for the nested sampling

`prior.size` The number of prior samples to use for nested sampling

Value

Returns:

posterior: The samples from the posterior, together with their log weights and log likelihoods as a m x n matrix, where m is the number of posterior samples and n is the number of parameters + 2. The log weights are the first column and the log likelihood values are the second column of this matrix. The sum of the log-weights = logZ.

logevidence: The logarithm of the evidence, a scalar.

means: A vector of the mean of each parameter, length = no. of parameters.

vars: A vector of the variance of each parameter, length = no. of parameters.

equalposterior: Equally weighted posterior samples together with their log likelihoods as a m x n matrix, where m is the number of posterior samples and n is the number of parameters + 1. The log likelihood values are the first column of this matrix.

fit.t: A vector of time points at which the model is fitted.

fit.y: A matrix of fitted model points, y, using posterior parameter samples in the model. Each column represents a different posterior sample.

fit.ymean: A vector of fitted model points, y, using the mean of the posterior parameter samples in the model.

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
B092_1.file <- system.file("extdata", "B092_1.csv", package = "babar")
data <- read.csv(B092_1.file, header=TRUE, sep=",",
                 na.strings=c("ND","NA"))

# Get a quick approximation of the evidence/model parameters.
results.linear.short <- Bayesfit(data,model="linear",inf.sigma=FALSE,
                                 tol=10,prior.size=25)

# Compute a better estimate of the evidence/model parameters (slow so not
# run as part of the automated examples).
## Not run:
results.linear.full <- Bayesfit(data,model="linear",inf.sigma=FALSE)

## End(Not run)
```

CauchyPrior *CauchyPrior*

Description

Generate samples from the Cauchy prior

Usage

```
CauchyPrior(u, location, scale)
```

Arguments

u	Input scalar/vector of values between 0 and 1
location	The location parameter (the peak of the distribution)
scale	The scale factor (full width at half maximum)

Value

Samples from Cauchy prior

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
samples <- CauchyPrior(runif(100), 5, 2)
summary(samples)
```

`compareDistributions` *compareDistributions*

Description

Compare two sets of normally distributed samples using nested sampling, to determine whether they have the same mean and variance.

Usage

```
compareDistributions(data.first, data.second)
```

Arguments

`data.first`	Samples from the first distriubtion, as a vector of normally distributed values
`data.second`	Samples from the second distribution as a vector of normally distributed values

Value

Bayes factor for the hypothesis that the distributions have the same mean and variance versus the hypothesis that they have different means and variances

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
data.a <- rnorm(10, 1, 1)
data.b <- rnorm(10, 5, 1)
compareDistributions(data.a, data.b)
```

`ExponentialPrior` *ExponentialPrior*

Description

Generate samples from the exponential prior

Usage

```
ExponentialPrior(u, rate)
```

Arguments

`u`	Input scalar/vector of values between 0 and 1
`rate`	The rate paramter (such that the mean of the distribution is 1 / rate)

Value

Samples from the Exponential prior

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
samples <- ExponentialPrior(runif(100), 5)
summary(samples)
```

`GaussianPrior` *GaussianPrior*

Description

Generate samples from normal distribution

Usage

```
GaussianPrior(u, mean, sd)
```

Arguments

`u`	Input scalar/vector of values between 0 and 1
`mean`	mean
`sd`	standard deviation

Value

Samples from normal distribution

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
samples <- GaussianPrior(runif(100), 5, 1)
summary(samples)
```

---

getEqualSamples *getEqualSamples*

Description

Return n equally weighted posterior samples

Usage

```
getEqualSamples(posterior, n = Inf)
```

Arguments

posterior	Matrix from the output of nested sampling. Should have log weights in column 1, log likelihoods in column 2 and then the parameter values in the remaining column(s).
n	Number of samples from the posterior required. If infinity (the deafult) this will return the maximum number of equally weighted samples generated from the posterior it can. Likewise if n is greater than this maximum number of samples.

Value

A set of equally weighted samples from the inferred posterior distribution.

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
mu <- 1
sigma <- 1
data <- rnorm(100, mu, sigma)

transform <- function(params) {
  tParams = numeric(length=length(params))
  tParams[1] = GaussianPrior(params[1], mu, sigma)
  tParams[2] = UniformPrior(params[2], 0, 2 * sigma)
  return(tParams)
}

llf <- function(params) {
  tParams = transform(params)
  mean = tParams[1]
  sigma = tParams[2]
  n <- length(data)
 ll <- -(n/2) * log(2*pi) - (n/2) * log(sigma**2) - (1/(2*sigma**2)) * sum((data-mean)**2)
  return(ll)
}

prior.size <- 25
tol <- 0.5

ns.results <- nestedSampling(llf, 2, prior.size, transform, tolerance=tol)

getEqualSamples(ns.results$posterior)
```

JeffreysPrior *JeffreysPrior*

Description

Generate samples from the Jeffreys prior

Usage

```
JeffreysPrior(u, log10lowerBound, log10upperBound)
```

Arguments

u	Input scalar/vector of values between 0 and 1
log10lowerBound	Base 10 logarithm of the lower bound
log10upperBound	Base 10 logarithm of the upper bound

Value

Samples from Jeffreys prior

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
samples <- JeffreysPrior(runif(100), -2, 8)
summary(samples)
```

`LogNormalPrior` *LogNormalPrior*

Description

Generate samples from the log normal distribution

Usage

```
LogNormalPrior(u, meanlog, sdlog)
```

Arguments

`u`	Input scalar/vector of values between 0 and 1
`meanlog`	Log of mean
`sdlog`	Log of standard deviation

Value

Samples from log normal distribution

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
samples <- LogNormalPrior(runif(100), 2, 2)
summary(samples)
```

nestedSampling *nestedSampling*

Description

Perform nested sampling for the given model (expressed through the log likelihood function).

Usage

```
nestedSampling(llFun, numberOfParameters, prior.size, transformParams,
    exploreFn = ballExplore, tolerance = 0.1)
```

Arguments

`llFun` The loglikelihood function, which should be of the form: llFun(params) where params is a vector containing a single instance of parameters, and should return the log likelihood for those parameters.

`numberOfParameters` The number of parameters to be inferred. Should always equal the number of parameters in the likelihood function.

`prior.size` The number of initial prior objects to sample. This is the size of the active set of samples maintained throughout the procedure.

`transformParams` Function to transform parameters from the unit hypercube into the correct distribution

`exploreFn` Preferred exploration routine. Defaults to a dodgy implementation currently but ballExplore likely to be more robust.

`tolerance` The tolerance at which the routine should stop. Changing this will affect how long it takes to run nested sampling. Larger values will terminate sooner but may give inaccurate evidence scores. This is the expected maximum remaining contribution to logZ from the points in the active set. delta Z_i = L_max*X_i < tolerance.

Value

logevidence: The logarithm of the evidence, a scalar.

posterior.samples: The samples from the posterior, together with their log weights and log likelihoods as a m x n matrix, where m is the number of posterior samples and n is the number of parameters + 2. The log weights are the first column and the log likelihood values are the second column of this matrix. The sum of the log-weights = logZ.

logZerror: An estimate of the numerical uncertainty of the log evidence.

parameterMeans: A vector of the mean of each parameter, length = no. of parameters.

parameterVariances: A vector of the variance of each parameter, length = no. of parameters.

entropy: The information — the natural logarithmic measure of the prior-to-posterior shrinkage.

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
mu <- 1
sigma <- 1
data <- rnorm(100, mu, sigma)

transform <- function(params) {
  tParams = numeric(length=length(params))
  tParams[1] = GaussianPrior(params[1], mu, sigma)
  tParams[2] = UniformPrior(params[2], 0, 2 * sigma)
  return(tParams)
}

llf <- function(params) {
  tParams = transform(params)
  mean = tParams[1]
  sigma = tParams[2]
  n <- length(data)
  ll <- -(n/2) * log(2*pi) - (n/2) * log(sigma**2) - (1/(2*sigma**2)) * sum((data-mean)**2)
  return(ll)
}

prior.size <- 25
tol <- 0.5

ns.results <- nestedSampling(llf, 2, prior.size, transform, tolerance=tol)
```

`UniformPrior` *UniformPrior*

Description

Generate samples from uniform distribution

Usage

```
UniformPrior(u, lowerBound, upperBound)
```

Arguments

`u`	Input scalar/vector of values between 0 and 1
`lowerBound`	lower bound
`upperBound`	upper bound

Value

Samples from uniform distribution within given bounds

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
samples <- UniformPrior(runif(1000), -5, 5)
summary(samples)
```

`WeibullPrior` *WeibullPrior*

Description

Generate samples from the Weibull prior

Usage

```
WeibullPrior(u, shape, scale)
```

Arguments

`u`	Input scalar/vector of values between 0 and 1
`shape`	The shape parameter of the distriubtion
`scale`	The scale parameter of the distribution

Value

Samples from the Weibull prior

Author(s)

Lydia Rickett, Matthew Hartley, Richard Morris and Nick Pullen

Examples

```
samples <- WeibullPrior(runif(100), 5, 5)
summary(samples)
```

Index