Package ‘SALTSampler’

Type     Package
Title    Efficient Sampling on the Simplex
Version  1.1.0
Date     2017-08-09
Author   Hannah Director, Scott Vander Wiel, James Gattiker
Maintainer Scott Vander Wiel <scottv@lanl.gov>
Imports  graphics, methods, stats
Depends  R (>= 3.0.0), lattice
Description The SALTSampler package facilitates Monte Carlo Markov Chain (MCMC) sampling of random variables on a simplex. A Self-Adjusting Logit Transform (SALT) proposal is used so that sampling is still efficient even in difficult cases, such as those in high dimensions or with parameters that differ by orders of magnitude. Special care is also taken to maintain accuracy even when some coordinates approach 0 or 1 numerically. Diagnostic and graphic functions are included in the package, enabling easy assessment of the convergence and mixing of the chain within the constrained space.
License  BSD_3_clause + file LICENSE
Suggests knitr, coda
RoxygenNote 6.0.1
NeedsCompilation no
Repository CRAN
Date/Publication 2017-08-10 19:35:03 UTC

R topics documented:

SALTSampler-package .................................................. 2
Diagnostics ............................................................. 3
GenData ................................................................. 4
Logit ..................................................................... 5
LogitScale ............................................................... 5
LogitSum ................................................................. 6
Description

The SALTSampler package facilitates Monte Carlo Markov Chain (MCMC) sampling of random variables on a simplex. A Self-Adjusting Logit Transform (SALT) proposal is used so that sampling is still efficient even in difficult cases, such as those in high dimensions or with parameters that differ by orders of magnitude. Special care is also taken to maintain accuracy even when some coordinates approach 0 or 1 numerically. Diagnostic and graphic functions are included in the package, enabling easy assessment of the convergence and mixing of the chain within the constrained space.

Details

Package: SALTSampler
Type: Package
Version: 0.1
Date: 2015-08-15
License: BSD_3_clause + file LICENSE

The main function for this package is runMh. Using user-defined information, runMh conducts MCMC on a simplex and outputs an object of class mhrun. The function can be used with any target distribution on the simplex defined by the user. Alternatively, two common posteriors types are built into the function and can be specified by the user. For type 'dirichlet', mhrun produces MCMC samples from a specified dirichlet distribution and for type 'multinomial', mhrun uses data to sample the distributional parameters of a multinomial distribution. Additionally, the functions Diagnostics and TriPlot can be used to analyze the output of mhrun.

Author(s)

Hannah Director, Scott Vander Wiel, Jim Gattiker

Examples

```r
###Dirichlet sampling in 3-simplex
dir <- RunMh(center = c(0.7, 0.2, 0.1), B = 2e3, concentration = 10,
             h = c(2, 2, 2), type = 'dirichlet', dat = NULL)
Diagnostics(mhOut = dir)
TriPlot(mhOut = dir)

###Multinomial sampling
```
## User-defined target distribution for a calibration problem

### Known function which we want to calibrate

```r
CalibFn <- function(y, logit = FALSE) {
  if (logit == TRUE) {
    y <- exp(LogPq(y)$logp)
  }
  return(out)
}
```

### Generated data

```r
z <- rnorm(n = 1000, mean = CalibFn(c(1/3, 1/3, 1/3), 2))
```

### User defined target distribution

```r
Target <- function(ycurrent, a, dat, pars = NULL) {
  out <- sum(dnorm(z, CalibFn(ycurrent, logit = TRUE), 2, log = TRUE)) -
  sum(dnorm(z, CalibFn(ycurrent, logit = TRUE), 2, log = TRUE)) +
  sum((a - 1)*LogPq(ycurrent)$logp - LogPq(ycurrent)$logp))
  return(out)
}
```

### Run sampler

```r
inputDist <- RunMh(center = c(1/3, 1/3, 1/3), B = 3e4, concentration = 3,
                    h = c(0.2, 0.2, 0.2), type = 'user', dat = z)
Diagnostics(mhOut = inputDist)
TriPlot(mhOut = inputDist)
```

## Plots and Summaries of RunMh Output

### Description

Taking in a `mhOut` object, this function outputs graphs and summaries to evaluate the performance of an MCMC run on a simplex. In particular, the acceptance rate is outputted for each dimension along with a trace plot. For type 'dirichlet', qqplots of the theoretical versus empirical marginal distributions are also provided for each dimension.
Usage
Diagnostics(mhOut)

Arguments

mhOut Object outputted by the function RunMH which summarizes a Metropolis Hasting run on a simplex

Examples

# Dirichlet run and diagnostic plots
dir <- RunMH(center = c(0.7, 0.2, 0.1), B = 2e3, concentration = 10,
h = c(2, 2, 2), type = 'dirichlet', dat = NULL)
Diagnostics(mhOut = dir)

GenData

Synthetic Data From a Multinomial Distribution

Description
This function generates a synthetic data set representing multiple draws from a multinomial distribution with user-specified parameters. A matrix of $n$ rows corresponding to each draw is outputted where the entry in the $i$th column and the $j$th row gives the number of the items that were in the $i$th bin on the $j$th trial.

Usage
GenData(center, n, size)

Arguments

center Vector of numeric values defining the parameters of a multinomial distribution. The $i$th value corresponds to the likelihood of a random variable being drawn from the $i$th bin

n The $n$ argument for the rmultinom function in base R which is defined to be the "number of random vectors to draw"

size The $Size$ argument for the rmultinom function in base R which is defined to be an "integer, say $N$, specifying the total number of objects that are put into $K$ boxes in the typical multinomial experiment."

References

Examples

#Generate sample data from a multinomial distribution
GenData(center = c(0.2, 0.3, 0.5), n = 10, size = 20)

Logit

Description

Returns the logit of a vector of probabilities, p. When logp is set to TRUE, the second argument contains natural logs of probabilities.

Usage

Logit(p, logp = FALSE)

Arguments

p Vector of probabilities or log probabilities
logp Boolean which is FALSE when the first argument contains probabilities and TRUE when the first argument contains log probabilities

Examples

#Find logit on natural scale
a <- c(0.4, 0.4, 0.1, 0.1)
Logit(p = a)

#Find logit on log scale
b <- c(log(1e-4), log(1e-6), log(1 - 1e-6))
b <- b/sum(b)
Logit(p = b, logp = FALSE)

LogitScale

Description

For \( x = \logit(p) \) and \( l = \log(s) \), this function returns \( \logit(s) \).

Usage

LogitScale(x, l)
Arguments

\( x \)  
Logit \( (p) \) where \( p \) is a vector of probabilities

\( l \)  
\( \exp(s) \) where \( 0 \leq s \leq 1 / \sum p_i \) to produce a well-defined return value.

Examples

# Calculates logit(x1) for p = (0.4, 0.3):
# x = (logit(0.4), logit(0.3)) and l = 0.7
LogitScale(x = logit(c(0.4, 0.3)), l = 0.7)

---

LogitSum  
\( \log \text{ of the Sum of Probabilities} \)

Description

For \( x = \text{logit}(p) \), this function returns \( s = \log(\sum p) \) where the sum of \( p \) is less than or equal to 1. Calculations are designed to preserve accuracy even for values numerically near 0 or 1.

Usage

LogitSum(x)

Arguments

\( x \)  
A vector of probabilities whose sum is less than or equal to 1

Examples

# Find logit sum for a single value
LogitSum(x = 0.1)

# Find logit sum for a vector of values
LogitSum(x = c(0.1, 0.4, 0.2))

---

LogPq  
\( \text{Computes } \log(p) \text{ and } \log(1 - p) \)

Description

For \( x = \text{logit}(p) \), this function returns \( \log(p) \) and \( \log(1 - p) \). Special care is taken to ensure accuracy when coordinates are numerically close to 0 or 1.

Usage

LogPq(x)
Arguments

\[ \text{Logit}(p) \text{ where } p \text{ is a vector of probabilities} \]

Examples

\[
\begin{align*}
\text{#Find log(p) and log(q) for } x &= \text{logit}(0.2) \\
   a &\leftarrow \text{log}(0.2/(1 - 0.2)) \\
   \text{LogPq}(x = a) \\

\text{#Find log(p) and log(q) for } x &= \text{logit}(1e-4) \\
   b &\leftarrow \text{log}(1e-4/(1 - 1e-4)) \\
   \text{LogPq}(x = b)
\end{align*}
\]

PropStep

\[ \text{PropStep} \quad \text{Draw a Proposal on a Simplex} \]

Description

Given a logit-scaled simplex point \( y \), this function draws a new logit-scaled simplex point. For a specified element, \( i \), a new point is drawn with Gaussian standard deviation \( h \). Then all other elements are rescaled such that they remain on the simplex. The returned value also includes a detailed balance term, \( \text{dbt} \), as an attribute.

Usage

\[ \text{PropStep}(y, i, h) \]

Arguments

\( y \) \quad \text{Vector of simplex points on the logit scale}

\( i \) \quad \text{Index value for the coordinate in the simplex point vector that should be modified initially}

\( h \) \quad \text{Gaussian standard deviation for the proposal distribution}

Value

\( \text{dbt} \) \quad \text{Detailed balance term}

Examples

\[
\begin{align*}
\text{#Propose new step from } y &= c(0.2, 0.3, 0.5) \\
   y &\leftarrow c(0.2, 0.3, 0.5) \\
   \text{PropStep}(y = \text{Logit}(y), i = 1, h = c(2, 2, 2))
\end{align*}
\]
**RunMh**

*Metropolis Hasting Algorithm Constrained on a Simplex*

**Description**

This function runs the Metropolis Hasting algorithm constrained on a simplex. The function can be used with any target distribution on the simplex defined by the user. Alternatively, two common target distributions are built into the function and can be specified by the user. The function is designed to continue to perform well in difficult cases, such as those in high dimensions or with parameters that differ by orders of magnitude. Care is also taken to ensure accuracy even when some coordinates are numerically close to 0 or 1.

**Usage**

```
RunMh(center, B, concentration = 1, h, type = 'user', dat = NULL, pars = NULL)
```

**Arguments**

- **center**: Vector of numeric values summing to 1 that define the center of the distributional parameters of the posterior. For type 'dirichlet', the parameter $a_i$ is defined such that $a_i$ is the $i$th element of center times concentration. For type 'multinom', the multinomial distribution parameter, $p_i$, is the $i$th value of center.
- **B**: Number of iterations to run the chain.
- **concentration**: This argument specifies the concentration parameter where $a_i$ is defined such that $a_i$ is the $i$th element of center times concentration. This is typically used with type 'dirichlet', but can also be used in a user-defined function. This argument defaults to 1, so has no effect if it is not specified.
- **h**: Vector of step sizes. Length of vector must match length of center.
- **type**: Specifies the target distribution. Select type 'user' if a target distribution has already been defined (see details). Select type 'dirichlet' for a Dirichlet distribution and type 'multinom' for a multinomial distribution.
- **dat**: A matrix or vector passing data to the sampler. For type 'multinom', this is a matrix giving data from repeated multinomial draws where the data is formatted in the same way as data obtained via GenData. The number of the items in the $i$th bin on the $j$th multinomial trial should be in the $i$th column and the $j$th row of the matrix. For type 'user', any matrix or vector of data can be used to match the form specified in the user's target function. If unspecified, this argument defaults to NULL.
- **pars**: A list of additional parameters that can be passed to the user-specified target function for type 'user' if desired. Argument defaults to NULL.
Details

Any target distribution on the simplex can be used with this function by defining a target distribution function in the environment prior to running RunMh. The function should be named Target and should take in parameters ycand and ycurrent, which are the current and proposed samples on the logit scale, and parameter a, which is center times concentration. Parameters dat and pars can be set to NULL. Alternatively, dat can be used to provide data to the target function and/or pars can be used to provide a list of additional parameters to the the target function. The target function should output the ratio of the log-likelihood of the posterior distribution for the proposal, \( \theta = ycand \), to the log-likelihood of the posterior for the current value, \( \theta = ycurrent \). For simple cases, there are built-in target distributions. For type 'dirichlet', RunMh uses a Dirichlet distribution as a posterior distribution. For type 'multinomial', RunMh samples the distributional parameters of a multinomial distribution that would have generated the data inputted for dat.

Value

An object of class mhOut. mhOut has 12 attributes.

- Y: Matrix of MCMC samples on logit scale
- S: Matrix of MCMC samples on true scale
- runTime: Summary of the MCMC runtime. The first entry gives the total user CPU time, the second entry gives the system CPU time, and the third entry gives the true elapsed time
- moveCount: Number of steps where the proposal value was accepted
- p: Length of center vector
- center: Vector of numeric values summing to 1 that help to define distributional parameters. For type 'dirichlet', the parameter a is defined such that \( a_i \) is the ith element of center times concentration. For type 'multinomial', the multinomial distribution parameter, \( p_i \), is the ith value of center
- B: Number of iterations to run the chain
- concentration: For type 'dirichlet', this argument specifies the concentration parameter where \( a \) is defined such that \( a_i \) is the ith element of center times concentration. Otherwise, this argument takes on its default value of 1 and has no effect
- h: Vector of step sizes. Length of vector must match length of center
- type: Specifies the target distribution. Select type 'user' if a target distribution has already been defined (see details). Select type 'dirichlet' for a Dirichlet distribution and type 'multinom' for a multinomial distribution
- dat: A matrix or vector passing data to the sampler. For type 'multinom', a matrix giving data from repeated multinomial draws where the data is formatted in the same way as data obtained via GenData. The number of the items in the ith bin on the jth multinomial trial should be in the ith column and the jth row of the matrix. For type 'user', any matrix or vector of data can be used to match the form specified in the user's target function. If unspecified, this argument defaults to NULL
- a: Dirichlet distribution parameters, \( a \), where \( a_i \) is the ith element of center times concentration
Examples

###Dirichlet sampling in 3-simplex

dir <- RunMh(center = c(0.7, 0.2, 0.1), B = 2e3, concentration = 10,
  h = c(2, 2, 2), type = 'dirichlet', dat = NULL)

###Multinomial sampling

# Not run:
sampData <- GenData(center = c(0.2, 0.3, 0.5), n = 100, size = 10)
multinom <- RunMh(center = c(0.2, 0.3, 0.5), B = 1e4, h = c(2,2,2),
  type = 'multinom', dat = sampData)

###User-defined target distribution for a calibration problem

# Not run:

#Known function which we want to calibrate
CalibFn <- function(y, logit = FALSE) {
  if (logit == TRUE) {
    y <- exp(LogPq(y)$logp)
  }
  return(out)
}

#Generate data
z <- rnorm(n = 1000, mean = CalibFn(c(1/3, 1/3, 1/3), 2))

#User defined target distribution
Target <- function(y cand, y current, a, dat, pars = NULL) {
  out <- sum(dnorm(dat, CalibFn(y cand, logit = TRUE), 2, log = TRUE)) -
    sum(dnorm(dat, CalibFn(y current, logit = TRUE), 2, log = TRUE)) +
    sum((a - 1)*LogPq(y cand)$logp - LogPq(y current)$logp))
  return(out)
}

#Run sampler
inputDist <- RunMh(center = c(1/3, 1/3, 1/3), B = 3e4, concentration = 3,
  h = c(0.2, 0.2, 0.2), type = 'user', dat = z)

## End(Not run)

---

**TriPlot**

*Plots MCMC Samples on a 3-Simplex*

###Description

This function plots samples from a 3-simplex projected into two dimensions. If `sumStat` is true, numerical summaries are also plotted on the graph. In particular, the theoretical mean is calculated under the assumption that the initial values entered by the user for `center` in the `runMh` function are...
correct. For type 'dirichlet' the theoretical mode is also calculated under the assumption that the initial values entered by the user for center in the runMh function are correct. These values are plotted along with the samples in the projected space.

**Usage**

TriPlot(mhOut, sumStat = FALSE)

**Arguments**

mhOut  
Output of the RunMh function

sumStat  
Boolean indicating whether or not summary statistics should be plotted on the graph

**Note**

If two or more parameter values are near zero, this plot may not be useful. In such cases, all samples may overlap in a single corner of the triangle, limiting the useful visual information provided by this plot.

**Examples**

```r
#Dirichlet triangle plot
dir <- RunMh(center = c(0.7, 0.2, 0.1), B = 2e3, concentration = 10,
             h = c(2, 2, 2), type = 'dirichlet', dat = NULL)
TriPlot(mhOut = dir, sumStat = TRUE)
```
Index

Diagnostics, 3
GenData, 4
Logit, 5
LogitScale, 5
LogitSum, 6
LogPq, 6
PropStep, 7
RunMh, 8
SALTSampler (SALTSampler-package), 2
SALTSampler-package, 2
TriPlot, 10