Package ‘RZigZag’

July 20, 2019

Type Package

Title Zig-Zag Sampler

Version 0.2.1

Date 2019-07-20

Author Joris Bierkens

Maintainer Joris Bierkens <j.bierkens@vu.nl>

Description Implements the Zig-Zag algorithm (Bierkens, Fearnhead, Roberts, 2016) <arXiv:1607.03188> applied and Bouncy Particle Sampler <arXiv:1510.02451> for a Gaussian target and Student distribution.

License GPL (>= 2)

Imports Rcpp (>= 0.12.3)

LinkingTo Rcpp, RcppEigen

RoxygenNote 6.1.1

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2019-07-20 08:00:04

R topics documented:

BPSGaussian *BPSGaussian*

Description

Applies the BPS Sampler to a Gaussian target distribution, as detailed in Bouchard-Côté et al, 2017. Assume potential of the form

$$U(x) = (x - mu)^T V(x - mu)/2,$$

i.e. a Gaussian with mean vector `mu` and covariance matrix `inv(V)`

Usage

```
BPSGaussian(V, mu, n_iter = -1L, finalTime = -1, x0 = numeric(0),
  v0 = numeric(0), refresh_rate = 1, unit_velocity = FALSE)
```

Arguments

`V`	the inverse covariance matrix (or precision matrix) of the Gaussian target distribution.
`mu`	mean of the Gaussian target distribution
`n_iter`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`finalTime`	If provided and nonnegative, run the BPS sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of `n_iterations`)
`x0`	starting point (optional, if not specified taken to be the origin)
`v0`	starting direction (optional, if not specified taken to be a random vector)
`refresh_rate`	`lambda_refresh`
`unit_velocity`	TRUE indicates velocities uniform on unit sphere, FALSE (default) indicates standard normal velocities

Value

Returns a list with the following objects:

`Times`: Vector of switching times

`Positions`: Matrix whose columns are locations of switches. The number of columns is identical to the length of `skeletonTimes`. Be aware that the skeleton points themselves are NOT samples from the target distribution.

`Velocities`: Matrix whose columns are velocities just after switches. The number of columns is identical to the length of `skeletonTimes`.

Examples

```
V <- matrix(c(3,1,1,3),nrow=2)
mu <- c(2,2)
x0 <- c(0,0)
result <- BPSGaussian(V, mu, n_iter = 100, x0 = x0)
plot(result$Positions[1,], result$Positions[2,],type='l',asp=1)
```

BPSIIDGaussian *BPSIIDGaussian*

Description

Applies the Bouncy Particle Sampler to a IID Gaussian distribution

Usage

```
BPSIIDGaussian(variance, dim = 1L, n_iter = -1L, finalTime = -1,
  x0 = numeric(0), v0 = numeric(0), refresh_rate = 1,
  unit_velocity = FALSE)
```

Arguments

`variance`	scalar indicating variance
`dim`	dimension
`n_iter`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`finalTime`	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of `n_iterations`)
`x0`	starting point (optional, if not specified taken to be the origin)
`v0`	starting direction (optional, if not specified taken to be a random vector)
`refresh_rate`	`lambda_refresh`
`unit_velocity`	TRUE indicates velocities uniform on unit sphere, FALSE (default) indicates standard normal velocities

Value

Returns a list with the following objects:

`Times`: Vector of switching times

`Positions`: Matrix whose columns are locations of switches. The number of columns is identical to the length of `skeletonTimes`. Be aware that the skeleton points themselves are NOT samples from the target distribution.

`Velocities`: Matrix whose columns are velocities just after switches. The number of columns is identical to the length of `skeletonTimes`.

Examples

```
result <- BPSIIDGaussian(1, 2, 1000)
plot(result$Positions[2,], result$Positions[1,],type='l',asp=1)
```

BPSStudentT *BPSStudentT*

Description

Applies the Zig-Zag Sampler to a Student T distribution (IID or spherically symmetric)

Usage

```
BPSStudentT(dof, dim = 1L, n_iter = -1L, finalTime = -1,
  x0 = numeric(0), v0 = numeric(0), sphericallySymmetric = TRUE,
  refresh_rate = 1, unit_velocity = FALSE)
```

Arguments

`dof`	scalar indicating degrees of freedom
`dim`	dimension
`n_iter`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`finalTime`	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of `n_iterations`)
`x0`	starting point (optional, if not specified taken to be the origin)
`v0`	starting direction (optional, if not specified taken to be a random vector)
`sphericallySymmetric`	boolean. If false, sample iid Student T distribution, if true (default) sample spherically summetric Student t dsitribution.
`refresh_rate`	lambda_refresh
`unit_velocity`	TRUE indicates velocities uniform on unit sphere, FALSE (default) indicates standard normal velocities

Value

Returns a list with the following objects:

`Times`: Vector of switching times

`Positions`: Matrix whose columns are locations of switches. The number of columns is identical to the length of `skeletonTimes`. Be aware that the skeleton points themselves are NOT samples from the target distribution.

`Velocities`: Matrix whose columns are velocities just after switches. The number of columns is identical to the length of `skeletonTimes`.

Examples

```
dim = 2
dof = 4
result <- BPSStudentT(dof, dim, n_iter=1000,sphericallySymmetric = TRUE)
plot(result$Positions[1,], result$Positions[2,],type='l',asp=1)
```

DiscreteSamples *DiscreteSamples*

Description

Extract discrete samples from a skeleton

Usage

```
DiscreteSamples(skeletonList, n_samples, coordinate = -1L)
```

Arguments

skeletonList	a piecewise deterministic skeleton (consisting of Times, Points and Velocities) returned by a sampler
n_samples	number of samples to obtain
coordinate	if specified, only obtain samples of the specified coordinate, otherwise obtain samples of all coordinates

Value

Returns a list containing the extracted samples and the times (on the continuous time scale) at which the samples are extracted

---

EstimateCovarianceMatrix *EstimateCovarianceMatrix*

Description

Estimates the covariance matrix of a piecewise deterministic skeleton

Usage

```
EstimateCovarianceMatrix(skeletonList, coordinate = -1L,
  zeroMeans = FALSE)
```

Arguments

`skeletonList`	a piecewise deterministic skeleton (consisting of Times, Points and Velocities) returned by a sampler
`coordinate`	if specified, only estimate the variance of the specified coordinate, otherwise estimate the covariance matrix of all coordinates
`zeroMeans`	if TRUE do not estimate means but assume a centered distribution

Value

Returns a list containing the estimated moment

`EstimateESS` *EstimateESS*

Description

Estimates the effective sample size (ESS) of a piecewise deterministic skeleton

Usage

```
EstimateESS(skeletonList, n_batches = 100L, coordinate = -1L,
  zeroMeans = FALSE)
```

Arguments

`skeletonList`	a piecewise deterministic skeleton (consisting of Times, Points and Velocities) returned by a sampler
`n_batches`	optional argument indicating the number of batches to use in the batch means estimation method
`coordinate`	if specified, only estimate the ESS of the specified coordinate, otherwise estimate the ESS of all coordinates
`zeroMeans`	if TRUE do not estimate means but assume a centered distribution

Value

Returns a list containing the estimated asymptotic variance, ESS and estimated covariance matrix

`EstimateMoment` *EstimateMoment*

Description

Estimates the p-th moment of a piecewise deterministic skeleton

Usage

```
EstimateMoment(skeletonList, p, coordinate = -1L)
```

Arguments

`skeletonList`	a piecewise deterministic skeleton (consisting of Times, Points and Velocities) returned by a sampler
`p`	moment to estimate
`coordinate`	if specified, only estimate the ESS of the specified coordinate, otherwise estimate the ESS of all coordinates

Value

Returns a list containing the estimated moment

`RZigZag` *RZigZag*

Description

Implements various piecewise deterministic Monte Carlo methods, including the Zig-Zag Sampler (Bierkens, Fearnhead, Roberts, 2019, `https://arxiv.org/abs/1607.03188`) and the Bouncy Particle Sampler (BPS, Bouchard-Côté et al., 2017, `https://arxiv.org/abs/1510.02451`). Typical usage consists of first creating a "skeleton" consisting of "events", which can be used directly for plotting trajectories. The skeleton may be post-processed to extract information, such as as moment and covariance estimates, discrete samples at fixed time intervals along the trajectory, effective sample size and asymptotic variance.

Details

This package currently consists of the following functions for generating skeletons: `ZigZagLogistic` for logistic regression, `ZigZagGaussian` for multivariate Gaussian, `ZigZagIIDGaussian` for a IID Gaussian target using Zig-Zag, `ZigZagStudentT` for spherically symmetric or factorized Student-t distribution, `BPSGaussian` for multivariate Gaussian using BPS, `BPSIIDGaussian` for a IID Gaussian target using BPS, `BPSStudentT` for BPS applied to a spherically symmetric or factorized Student-t distribution. Furthermore the package contains the following functions for post-processing: `EstimateESS` (to estimate asymptotic variance and effective sample size for individual coordinates), `EstimateMoment`, `EstimateCovarianceMatrix` and `DiscreteSamples`.

Author(s)

Joris Bierkens

With thanks to Matt Moores, `https://mattstats.wordpress.com/`, for his help in getting from C++ code to a CRAN-ready Rcpp based package.

`ZigZagGaussian` *ZigZagGaussian*

Description

Applies the Zig-Zag Sampler to a Gaussian target distribution, as detailed in Bierkens, Fearnhead, Roberts, The Zig-Zag Process and Super-Efficient Sampling for Bayesian Analysis of Big Data, 2016. Assume potential of the form

$$U(x) = (x - mu)^T V (x - mu)/2,$$

i.e. a Gaussian with mean vector `mu` and covariance matrix `inv(V)`

Usage

```
ZigZagGaussian(V, mu, n_iter = -1L, finalTime = -1, x0 = numeric(0),
  v0 = numeric(0))
```

Arguments

`V`	the inverse covariance matrix (or precision matrix) of the Gaussian target distribution.
`mu`	mean of the Gaussian target distribution
`n_iter`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`finalTime`	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of `n_iterations`)
`x0`	starting point (optional, if not specified taken to be the origin)
`v0`	starting direction (optional, if not specified taken to be +1 in every component)

Value

Returns a list with the following objects:

`Times`: Vector of switching times

`Positions`: Matrix whose columns are locations of switches. The number of columns is identical to the length of `skeletonTimes`. Be aware that the skeleton points themselves are NOT samples from the target distribution.

`Velocities`: Matrix whose columns are velocities just after switches. The number of columns is identical to the length of `skeletonTimes`.

Examples

```
V <- matrix(c(3,1,1,3),nrow=2)
mu <- c(2,2)
result <- ZigZagGaussian(V, mu, 100)
plot(result$Positions[1,], result$Positions[2,],type='l',asp=1)
```

`ZigZagIIDGaussian` *ZigZagIIDGaussian*

Description

Applies the Zig-Zag Sampler to a IID Gaussian distribution

Usage

```
ZigZagIIDGaussian(variance, dim = 1L, n_iter = -1L, finalTime = -1,
  x0 = numeric(0), v0 = numeric(0))
```

Arguments

`variance`	scalar indicating variance
`dim`	dimension
`n_iter`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`finalTime`	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of `n_iterations`)
`x0`	starting point (optional, if not specified taken to be the origin)
`v0`	starting direction (optional, if not specified taken to be +1 in every component)

Value

Returns a list with the following objects:

`Times`: Vector of switching times

`Positions`: Matrix whose columns are locations of switches. The number of columns is identical to the length of `skeletonTimes`. Be aware that the skeleton points themselves are NOT samples from the target distribution.

`Velocities`: Matrix whose columns are velocities just after switches. The number of columns is identical to the length of `skeletonTimes`.

Examples

```
result <- ZigZagIIDGaussian(1, 2, 1000)
plot(result$Positions[2,], result$Positions[1,],type='l',asp=1)
```

ZigZagLogistic *ZigZagLogistic*

Description

Applies the Zig-Zag Sampler to logistic regression, as detailed in Bierkens, Fearnhead, Roberts, The Zig-Zag Process and Super-Efficient Sampling for Bayesian Analysis of Big Data, 2019.

Usage

```
ZigZagLogistic(dataX, dataY, n_iter = -1L, finalTime = -1,
  x0 = numeric(0), v0 = numeric(0), cv = FALSE)
```

Arguments

`dataX`	Design matrix containing observations of the independent variables x. The i-th row represents the i-th observation with components x_i,1, ..., x_i,d.
`dataY`	Vector of length n containing 0, 1-valued observations of the dependent variable y.
`n_iter`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`finalTime`	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of `n_iterations`)
`x0`	starting point (optional, if not specified taken to be the origin)
`v0`	starting direction (optional, if not specified taken to be +1 in every component)
`cv`	optional boolean to indicate the use of subsampling with control variates

Value

Returns a list with the following objects:

`Times`: Vector of switching times

`Positions`: Matrix whose columns are locations of switches. The number of columns is identical to the length of `skeletonTimes`. Be aware that the skeleton points themselves are NOT samples from the target distribution.

`Velocities`: Matrix whose columns are velocities just after switches. The number of columns is identical to the length of `skeletonTimes`.

Examples

```
require("RZigZag")

generate.logistic.data <- function(beta, n.obs) {
  dim <- length(beta)
  dataX <- cbind(rep(1.0,n.obs), matrix(rnorm((dim -1) * n.obs), ncol = dim -1));
  vals <- dataX %*% as.vector(beta)
```

```
    generateY <- function(p) { rbinom(1, 1, p)}
  dataY <- sapply(1/(1 + exp(-vals)), generateY)
    return(list(dataX = dataX, dataY = dataY))
}

beta <- c(1,2)
data <- generate.logistic.data(beta, 1000)
result <- ZigZagLogistic(data$dataX, data$dataY, 1000)
plot(result$Positions[1,], result$Positions[2,],type='l',asp=1)
```

`ZigZagStudentT` *ZigZagStudentT*

Description

Applies the Zig-Zag Sampler to a Student T distribution (IID or spherically symmetric)

Usage

```
ZigZagStudentT(dof, dim = 1L, n_iter = -1L, finalTime = -1,
  x0 = numeric(0), v0 = numeric(0), sphericallySymmetric = TRUE)
```

Arguments

`dof`	scalar indicating degrees of freedom
`dim`	dimension
`n_iter`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`finalTime`	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of `n_iterations`)
`x0`	starting point (optional, if not specified taken to be the origin)
`v0`	starting direction (optional, if not specified taken to be +1 in every component)
`sphericallySymmetric`	boolean. If false, sample iid Student T distribution, if true (default) sample spherically summetric Student t dsitribution.

Value

Returns a list with the following objects:

`Times`: Vector of switching times

`Positions`: Matrix whose columns are locations of switches. The number of columns is identical to the length of `skeletonTimes`. Be aware that the skeleton points themselves are NOT samples from the target distribution.

`Velocities`: Matrix whose columns are velocities just after switches. The number of columns is identical to the length of `skeletonTimes`.

Examples

```
dim = 2
dof = 4
result <- ZigZagStudentT(dof, dim, n_iter=1000, sphericallySymmetric = TRUE)
plot(result$Positions[1,], result$Positions[2,],type='l',asp=1)
```