Package 'RZigZag'

April 30, 2018

Type Package

Title Zig-Zag Sampler

Version 0.1.6

Date 2018-04-30

Author Joris Bierkens

Maintainer Joris Bierkens <joris.bierkens@tudelft.nl>

Description Implements the Zig-Zag algorithm with subsampling and control variates (ZZ-CV) of (Bierkens, Fearnhead, Roberts, 2016) <arXiv:1607.03188> as applied to Bayesian logistic regression, as well as basic Zig-Zag for a Gaussian target distribution, and Bouncy Particle Sampler for a Gaussian target.

License GPL (>= 2)

Imports Rcpp (>= 0.12.3)

LinkingTo Rcpp, RcppEigen

RoxygenNote 6.0.1

Encoding UTF-8

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-04-30 11:05:01 UTC

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_iterations, x0, finalTime = -1, refresh_rate = 1,
  unit_velocity = TRUE, n_samples = 0L, n_batches = 0L,
  computeCovariance = FALSE)
```

Arguments

`V`	the inverse covariance matrix of the Gaussian target distribution
`mu`	mean of the Gaussian target distribution
`n_iterations`	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
`x0`	starting point
`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`)
`refresh_rate`	`lambda_refresh`
`unit_velocity`	TRUE indicates velocities uniform on unit sphere, FALSE indicates standard normal velocities
`n_samples`	Number of discrete time samples to extract from the Zig-Zag skeleton.
`n_batches`	If non-zero, estimate effective sample size through the batch means method, with n_batches number of batches.
`computeCovariance`	Boolean indicating whether to estimate the covariance matrix.

Value

Returns a list with the following objects:

`skeletonTimes`: Vector of switching times

`skeletonPoints`: 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.

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

`samples`: If `n_samples > 0`, this is a matrix whose `n_samples` columns are samples along the Zig-Zag trajectory.

`mode`: Not used for a Gaussian target.

`batchMeans`: If `n_batches > 0`, this is a matrix whose `n_batches` columns are the batch means

`means`: If `n_batches > 0`, this is a vector containing the means of each coordinate along the Zig-Zag trajectory

`covariance` :If `n_batches > 0` or `computeCovariance = TRUE`, this is a matrix containing the sample covariance matrix along the trajectory

`asVarEst`: If `n_batches > 0` this is an estimate of the asymptotic variance along each component

`ESS`: If `n_batches > 0` this is an estimate of the effective sample size along each component

Examples

```
V <- matrix(c(3,1,1,3),nrow=2)
mu <- c(2,2)
x0 <- c(0,0)
result <- BPSGaussian(V, mu, 100, x0, n_samples = 10)
plot(result$skeletonPoints[1,], result$skeletonPoints[2,],type='l',asp=1)
points(result$samples[1,], result$samples[2,], col='magenta')
```

`RZigZag` *RZigZag*

Description

Implements the Zig-Zag algorithm with subsampling and control variates (ZZ-CV) of (Bierkens, Fearnhead, Roberts, 2018) https://arxiv.org/abs/1607.03188 as applied to Bayesian logistic regression, as well as basic Zig-Zag for a Gaussian target distribution.

Details

This package currently consists of the following functions: `ZigZagLogistic` for logistic regression, `ZigZagGaussian` for multivariate Gaussian, and `BPSGaussian` for multivariate Gaussian using BPS.

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_iterations, x0, finalTime = -1, n_samples = 0L,
  n_batches = 0L, computeCovariance = FALSE)
```

Arguments

V	the inverse covariance matrix of the Gaussian target distribution
mu	mean of the Gaussian target distribution
n_iterations	Number of algorithm iterations; will result in the equivalent amount of skeleton points in Gaussian case because no rejections are needed.
x0	starting point
finalTime	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of n_iterations)
n_samples	Number of discrete time samples to extract from the Zig-Zag skeleton.
n_batches	If non-zero, estimate effective sample size through the batch means method, with n_batches number of batches.
computeCovariance	Boolean indicating whether to estimate the covariance matrix.

Value

Returns a list with the following objects:

skeletonTimes: Vector of switching times

skeletonPoints: 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.

skeletonDirections: Matrix whose columns are directions just after switches. The number of columns is identical to the length of skeletonTimes.

samples: If n_samples > 0, this is a matrix whose n_samples columns are samples along the Zig-Zag trajectory.

mode: Not used for a Gaussian target.

batchMeans: If n_batches > 0, this is a matrix whose n_batches columns are the batch means

means: If n_batches > 0, this is a vector containing the means of each coordinate along the Zig-Zag trajectory

covariance :If n_batches > 0 or computeCovariance = TRUE, this is a matrix containing the sample covariance matrix along the trajectory

asVarEst: If n_batches > 0 this is an estimate of the asymptotic variance along each component

ESS: If n_batches > 0 this is an estimate of the effective sample size along each component

Examples

```
V <- matrix(c(3,1,1,3),nrow=2)
mu <- c(2,2)
x0 <- c(0,0)
result <- ZigZagGaussian(V, mu, 100, x0, n_samples = 10)
plot(result$skeletonPoints[1,], result$skeletonPoints[2,],type='l',asp=1)
points(result$samples[1,], result$samples[2,], col='magenta')
```

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, 2016.

Usage

```
ZigZagLogistic(dataX, dataY, n_iterations, x0 = numeric(0), finalTime = -1,
  subsampling = TRUE, controlvariates = TRUE, n_samples = 0L,
  n_batches = 0L, computeCovariance = FALSE, upperbound = FALSE)
```

Arguments

dataX	Matrix containing the independent variables x. The i-th column represents the i-th observation with components x_1,i, ..., x_d,i.
dataY	Vector of length n containing 0, 1-valued observations of the dependent variable y.
n_iterations	Integer indicating the number of iterations, i.e. the number of proposed switches.
x0	Optional argument indicating the starting point for the Zig-Zag sampler
finalTime	If provided and nonnegative, run the sampler until a trajectory of continuous time length finalTime is obtained (ignoring the value of n_iterations)
subsampling	Boolean. Use Zig-Zag with subsampling if TRUE.

controlvariates
: Boolean. Use Zig-Zag with subsampling combined with control variates if TRUE (overriding any value of subsampling).

n_samples
: Number of discrete time samples to extract from the Zig-Zag skeleton.

n_batches
: If non-zero, estimate effective sample size through the batch means method, with n_batches number of batches.

computeCovariance
: Boolean indicating whether to estimate the covariance matrix.

upperbound
: Boolean. If TRUE, sample without subsampling and using a constant upper bound instead of a linear Hessian dependent upper bound

Value

Returns a list with the following objects:

skeletonTimes: Vector of switching times

skeletonPoints: 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.

skeletonDirections: Matrix whose columns are directions just after switches. The number of columns is identical to the length of skeletonTimes.

samples: If n_samples > 0, this is a matrix whose n_samples columns are samples at fixed intervals along the Zig-Zag trajectory.

mode: If controlvariates = TRUE, this is a vector containing the posterior mode obtained using Newton's method.

batchMeans: If n_batches > 0, this is a matrix whose n_batches columns are the batch means

means: If n_batches > 0, this is a vector containing the means of each coordinate along the Zig-Zag trajectory

covariance: If n_batches > 0 or computeCovariance = TRUE, this is a matrix containing the sample covariance matrix along the trajectory

asVarEst: If n_batches > 0 this is an estimate of the asymptotic variance along each component

ESS: If n_batches > 0 this is an estimate of the effective sample size along each component

Examples

```
require("RZigZag")
generate.logistic.data <- function(beta, n.obs) {
  dim <- length(beta)
  dataX <- rbind(rep(1, n.obs), matrix(rnorm((dim -1) * n.obs), nrow = dim -1));
  vals <- colSums(dataX * as.vector(beta))
  generateY <- function(p) { rbinom(1, 1, p)}
  dataY <- sapply(1/(1 + exp(-vals)), generateY)
  return(list(dataX, dataY))
}

beta <- c(1,2)
data <- generate.logistic.data(beta, 1000)
```

```
result <- ZigZagLogistic(data[[1]], data[[2]], 1000, n_samples = 100)
plot(result$skeletonPoints[1,], result$skeletonPoints[2,],type='l',asp=1)
points(result$samples[1,], result$samples[2,], col='magenta')
```

Index