Package 'ars'

December 16, 2015

Type Package

Title An R Implementation Of The Adaptive Rejection Sampling (ARS) Algorithm

Version 1.0

Maintainer Shamindra Shrotriya <shamindra@berkeley.edu>

LazyData true

URL https://github.com/shamindras/ars.git

Imports numDeriv,
knitr,
rmarkdown

Description An R implementation of the adaptive rejection sampling (ARS) algorithm. This is based on the P. Wild and W. R. Gilks (1992) paper ``Adaptive rejection sampling for Gibbs sampling".

License GPL-2

RoxygenNote 5.0.1

VignetteBuilder knitr

R topics documented:

ars *Main function to carry out simulation*

Description

Main function to carry out simulation

Usage

```
ars(n, g, D, k = 100)
```

Arguments

n	A number indicates how many accepted points user wants to generate.
g	A function user wants to generate samples from.
D	A vector with two elements indicates the domain of the sample generation.
k	The number of inital x points to start the algorithm. Default is 100.

Value

A numeric vector of length n, each element of which is a value sampled from g.

References

https://stat.duke.edu/~cnk/Links/tangent.method.pdf

See Also

Gilks et al

Examples

```
sample 1000 points from the standard normal distribution using adaptive
# rejection sample

set.seed(0)
dnorm1000 <- ars(1000,g=dnorm,D=c(-Inf,Inf))
hist(dnorm1000, breaks=30, main="1000 points sampled from N(0,1)")

# sample 500 point from the chisquare distribution with df=5

set.seed(123)
dchisq500 <- ars(500,g=function(x) dchisq(x,df=5), D=c(0,Inf))
hist(dchisq500, breaks=30)

# sample 1000 points from the exponential distribution

set.seed(0)
dexp1000 <- ars(1000,function(x) exp(-x), c(0,Inf))
hist(dexp1000,breaks=30)
```

faux_CheckLogConcavity *Helper function to check the log concavity*

Description

Helper function to check the log concavity

Usage

```
faux_CheckLogConcavity(inp_gfun, inp_Dvec)
```

Arguments

inp_gfun	A function user wants to generate samples from. This function is used to calculate $h(x) = \log(g(x))$.
inp_Dvec	A numeric vector of length 2 indicating the support of inp_gfun.

Value

A logical vector of length 1: TRUE if inp_gfun is log-concave, FALSE if inp_gfun is not log-concave.

faux_findmode *Helper function to find mode of a given univariate function $g(x)$*

Description

Helper function to find mode of a given univariate function $g(x)$

Usage

```
faux_findmode(optim_intervalvec, inp_gfun)
```

Arguments

optim_intervalvec	A vector with two elements indicates the support domain of the sample generation.
inp_gfun	A function of x which the user wants to generate samples from. This function is used to calculate $h(x) = \log(g(x))$.

Value

A list with three elements.

min_int	The minimum of the parameter optim_intervalvec.
superStarSeed	The point within optim_intervalvec at which the optimization routine begins.
faux_findmode_par	the mode of the function inp_gfun.

faux_hPrimex *Helper function to get first derivative of $h(x)$*

Description

Helper function to get first derivative of $h(x)$

Usage

```
faux_hPrimex(inp_gfun, inp_xvec)
```

Arguments

inp_gfun	A function of x which the user wants to generate samples from. This function is used to calculate $h(x) = \log(g(x))$.
inp_xvec	A number indicates the x-axis of the point

Value

A numeric vector of length equal to inp_xvec. The elements of the returned value are equal to the first derivative of $h(x)$ at the points in inp_xvec.

faux_hx *Helper function to get $h(x) = \log(g(x))$.*

Description

Helper function to get $h(x) = \log(g(x))$.

Usage

```
faux_hx(inp_gfun)
```

Arguments

inp_gfun	A function of x which the user wants to generate samples from. This function is used to calculate $h(x) = \log(g(x))$.

Value

The function $h(x) = \log(g(x))$. It takes as input the same input to inp_gfun.

faux_InitChoose *Helper function to choose two starting points*

Description

Helper function to choose two starting points

Usage

```
faux_InitChoose(inp_gfun, inp_Dvec, inp_Initnumsampvec = 2)
```

Arguments

inp_gfun	A function user wants to generate samples from. This function is used to calculate $h(x) = \log(g(x))$
inp_Dvec	A vector with two elements indicates the support domain of the sample generation.
inp_Initnumsampvec	An even integer determining the number of points to initially sample - should be even

Value

A list with 7 elements.

init_sample_points

num_sample_pts_mode

support_classify
: Based on the support function, determine the type of bounds specified: e.g. (-Inf, Inf) then = "negInf_posInf" e.g. (-Inf, 10) then = "negInf_posBnd" e.g. (-10, Inf) then = "negBnd_posInf" e.g. (-13, 55) then = "negBnd_posBnd"

mode
: This is a single element vector returning the mode of the log of the function inp_gfun

.

support
: The support of the function inp_gfun. Equivalent to inp_Dvec but in ascending order.

faux_Lkx *Helper function to get the lower bound linear function l_k(x).*

Description

Helper function to get the lower bound linear function $l_k(x)$.

Usage

```
faux_Lkx(inp_xvec, inp_gfun)
```

Arguments

inp_xvec	A vector of x values of all points and we should be able to get the index of x
inp_gfun	A function user wants to generate samples from. This function is use to calculate $h(x) = \log(g(x))$

Value

A list of functions. The length of the list is one greater than the length of the input `inp_xvec`. Each element of the list is a piece of the piecewise function $l_k(x)$, which forms the lower hull of the function $h(x)$.

faux_SampleSkx *Helper function to sample a value x^* from s_k(x)*

Description

Helper function to sample a value x^* from $s_k(x)$

Usage

```
faux_SampleSkx(inp_uintervallist, inp_sfunlist)
```

Arguments

inp_uintervallist	A list of intervals between the z points
inp_sfunlist	A list of functions which form the $s_k(x)$ function

Value

A named numeric vector of length 1.

faux_SampleSkx_out	Value sampled from the function $s_k(x)$

faux_Skx *Helper function to create piecewise function $s_k(x)$*

Description

Helper function to create piecewise function $s_k(x)$

Usage

```
faux_Skx(inp_uintervallist, inp_ufunlist)
```

Arguments

inp_uintervallist	A list of intervals between z values, as in the output from uInterval.
inp_ufunlist	A list of functions, the output from uFun.

Value

A list of functions. The length of the list is equal to the length of the inputs `inp_uintervallist` and `inp_ufunlist`. Each element of the list is one piece of the piecewise function $s_k(x)$.

faux_uInterval *Helper function to create the z intervals.*

Description

Helper function to create the z intervals.

Usage

```
faux_uInterval(inp_z)
```

Arguments

inp_z	A vector of z values, such as the output from `faux_Zj()`.

Value

A list of numeric vectors, each of length 2. The length of the list is one less than the length of the input `inp_z`. Each element of the list is an interval between 2 consecutive z points.

faux_Ukx *Helper function to get the upper bound linear function u_k(x)*

Description

Helper function to get the upper bound linear function $u_k(x)$

Usage

```
faux_Ukx(inp_xvec, inp_gfun)
```

Arguments

inp_xvec	A vector of x values of all points
inp_gfun	A function user wants to generate samples from. This function is used to calculate $h(x) = \log(g(x))$

Value

A list of functions. The length of the list is equal to the length of the input inp_xvec. Each of the elements of the list is a piece of the piecewise function $u_k(x)$, which forms the upper hull of $h(x)$.

faux_Zj *Helper function to get the intersection of tangents at x_j and x_j + 1*

Description

Helper function to get the intersection of tangents at x_j and x_{j+1}

Usage

```
faux_Zj(inp_xvec, inp_gfun, inp_Dvec)
```

Arguments

inp_xvec	A vector of x values of your points. The vector is ordered in an increasing order.
inp_gfun	A fucntion user wants to generate samples from. This function is used to calculate $h(x) = \log(g(x))$
inp_Dvec	A vector with 2 elements indicating the domain the function g

Value

A numeric vector. The elements of the vector are the intersection points of $s_k(x)$, the upper hull of the function $h(x)$.

Index