Package 'diffusion'

June 29, 2006

Title Diffusion processes

Version 0.3.1

Author Barry Rowlingson <b.rowlingson@lancs.ac.uk>

Description Create diffusion processes, simulate, plot etc.

Maintainer Barry Rowlingson <b.rowlingson@lancs.ac.uk>

License GPL

R topics documented:

Afunction *Get A() for an EA1-ready diffusion process*

Description

Returns the A function for an EA1-ready diffusion process.

Usage

```
Afunction(process)
```

Arguments

process	An EA1-ready diffusion process

Value

The A function for the process.

Author(s)

Barry Rowlingson

EA1functions<- *Add the functions needed for EA1 to a process*

Description

To simulate using EA1, you need to add some extra functions to a diffusion process.

Usage

```
EA1functions<-(x, value)
```

Arguments

x	A diffusion process object
value	a list with components A, K, Kbounds and alphabounds.

Details

The components of the list are:

A The integrated drift from 0 to x

K (alpha'(x) + alpha(x)^2)$/2 vector of length 2 of limits on alpha (the drift)$

Kbounds vector of length 2 of limits on K

alphabounds

Author(s)

Barry Rowlingson

Examples

```
ap <- diffusionProcess(atan(Xt))
EA1functions(ap) <- list(A=function(x){x*atan(x)-0.5*(log(1+x^2))},K=function(x){0.5*(at
```

---

alphaBounds *Get the bounds on alpha for an EA1-ready diffusion process*

Description

Get the bounds on alpha for an EA1-ready diffusion process

Usage

```
alphaBounds(process)
```

Arguments

process	An EA1-ready diffusion process

Value

A vector of length 2 of the bounds on alpha.

Author(s)

Barry Rowlingson

---

ars *Adaptive Rejection Sampler for EA1 processes*

Description

Samples from density proportional to exp(-(x)$^2/(2) + H(x))where H'(x) is bounded by bounds$

Usage

```
ars(n = 1, H, bounds, plot = FALSE, targetacceptance = 0.8)
```

Arguments

n	Number of samples to return
H	H Function
bounds	Bounds on H'
plot	Plot things as we go
targetacceptance	Threshold for adapting

Value

A list:

`samples`	The sampled values
`trials`	Other stuff
`f`	Other stuff
`fbound`	Other stuff
`H`	Other stuff
`U`	Other stuff
`cp`	Envelope control points

Author(s)

Duncan Murdoch

See Also

rH

Examples

```
ars(10,function(x){1-cos(x)},c(-1,1))
```

`bridge` *Simulate a Brownian bridge*

Description

Simulate a Brownian bridge

Usage

```
bridge(times, x, y, tau)
```

Arguments

`times`	Return bridge at these time points
`x`	value at t=0
`y`	value at t=tau
`tau`	end time of bridge

Details

This function generates a Brownian bridge from (0,x) to (tau,y).

Value

A vector of the length of times with the values of the bridge.

Author(s)

Barry Rowlingson, after Alex Beskos (C version)

Examples

```
bridge(0:10,0,10,10)
```

---

diffusionProcess *Create a diffusion process object*

Description

This creates an object that represents a diffusion process.

Usage

```
diffusionProcess(mu)
```

Arguments

mu	The expression in Xt for the mean, where Xt will be the current value of the process.

Details

TBD

Value

A diffusion process object

Author(s)

Barry Rowlingson

Examples

```
dp1 = diffusionProcess(sin(Xt-2))
```

partialProp *EA1 realisation core routine*

Description

Generates a realisation of a diffusion process using EA1

Usage

```
partialProp(time, y, process)
```

Arguments

time	End point in time for the realisation
y	Start value of the process
process	EA1-ready diffusion process object

Details

Computes an EA1 realisation from (0,y) to (time,?). method: sample an endpoint from the endpoint distribution, create a brownian bridge, generate some poisson process, accept if no points of the process are under the bridge

Value

A list with various components:

ending	The end point
skel	The skeleton
endCount	Number of samples needed for the endpoint. Broken with DM's sampler
propCount	Number of proposals needed before acceptance at the Poisson point stage

Author(s)

Barry Rowlingson, from C code by Alex Beskos

phiDiffu *Get the phi function for an EA1 process*

Description

Gets the phi function for an EA1 process

Usage

```
phiDiffu(dp)
```

Arguments

dp	An EA1-ready diffusion process

Value

A function, phi, for the process. Its a simple function of K.

Author(s)

Barry Rowlingson

---

phiMax *Get the max phi for an EA1-ready diffusion process*

Description

Gets the max of the phi function.

Usage

```
phiMax(dp)
```

Arguments

dp	an EA1-ready diffusion process

Details

phiMax is just the range of the bounds of K

Value

a scalar value.

Author(s)

Barry Rowlingson

---

plot.diffusionProcess *Plot a diffusion process*

Description

Plot one realisation of a diffusion process on a default basis.

Usage

```
plot.diffusionProcess(dp, X0 = 0, dTime = tDiff(c(0, 1), eps = 1/100), random)
```

Arguments

`dp`	A diffusion process object
`X0`	The initial value of the process.
`dTime`	The basis for the simulation of the process
`random`	The random part of the realisation.

Details

Does a pretty plot of a diffusion process.

Value

None

Author(s)

Barry Rowlingson

Examples

```
plot(diffusionProcess(sin(Xt)))
```

`plot.realDiffusion` *Plot a realisation of a diffusion process.*

Description

Draw a pretty plot of a realisation of a diffusion process.

Usage

```
plot.realDiffusion(obj, ...)
```

Arguments

`obj`	a realisation of a diffusion process
`...`	Other arguments, to be passed to plot eventually. Not yet implemented.

Details

TBD

Value

None

Author(s)

Barry Rowlingson

Examples

```
dp1=diffusionProcess(sin(Xt))
rdp1=rEuler(dp1,0)
plot(rdp1)
```

---

poissonPoints *Simulate poisson points for EA1*

Description

Simulate poisson points for EA1.

Usage

```
poissonPoints(size, tlim, bound)
```

Arguments

size	Number of points
tlim	Time range (vector of length 2)
bound	Upper bound (lower bound is zero)

Value

A matrix of two columns of X and Y coords

Author(s)

Barry Rowlingson

Examples

```
poissonPoints(4,c(0,1),5)
```

---

print.diffusionProcess *Print method for a diffusion process*

Description

Print a diffusion process object. It shows the mean component.

Usage

```
print.diffusionProcess(x, ...)
```

Arguments

x	A diffusion process object.
...	Other arguments, currently unused.

Value

None

Author(s)

Barry Rowlingson

Examples

```
dp1=diffusionProcess(sin(Xt=1))
print.diffusionProcess(dp1)
```

---

rEA1 *Simulate a diffusion process using EA1*

Description

Simulate a diffusion process using the EA1 method.

Usage

```
rEA1(process, x0 = 0, dTime = tDiff(c(0, 1), eps = 1/10))
```

Arguments

process	A diffusion process object
x0	Start value of the process
dTime	Time points for simulation

Value

A data frame with components:

time	Time value
X	Corresponding value of the simulated process

The data frame has values corresponding to the points of interest in the dTime argument. Lots of other information is stored in attributes of the data frame.

Author(s)

Barry Rowlingson, after Alex Beskos

See Also

diffusionProcess, tDiff

Examples

```
ap <- diffusionProcess(atan(Xt))
EA1functions(ap) <- list(A=function(x){x*atan(x)-0.5*(log(1+x^2))},K=function(x){0.5*(ata
rEA1(ap)
```

rEuler — *Create a realisation of a diffusion process.*

Description

Given a diffusion process, and an optional time basis, simulate one realisation of the process.

Usage

```
rEuler(dProcess, x0=0, dTime = tDiff(c(0, 1), eps = 1/100), random = NULL)
```

Arguments

dProcess	A diffusion process object.
x0	The initial value of the process, at the initial time.
dTime	The temporal basis for the simulation
random	The random component part of the simulation

Value

An object of class 'realDiffusion'. It is also a data frame with components 'time' and 'X' coming from the time points of interest from the dTime argument, by default time=0 and time=1.

Author(s)

Barry Rowlingson

Examples

```
rdp1=rEuler(diffusionProcess(sin(Xt)),0)
print(rdp1)
```

rNearGaussian — *Adaptive rejection sampler for near-Gaussian distributions*

Description

Samples from density proportional to `exp(-(x-mu)^2/(2*sigma^2) + H(x))`.

Usage

```
rNearGaussian(n, H, bounds, mu = 0, sigma = 1, targetacceptance = 0.8, prev = NU
```

Arguments

`n`	number of samples to return
`H`	The H function
`bounds`	bounds on H' (vector of length 2)
`mu`	the mean of the base distribution
`sigma`	the standard deviation of the base distribution
`targetacceptance`	add the point to the adaptive envelope if the acceptance rate is below this value
`prev`	(optionally) the result from a previous with the same `H` and `bounds`

Details

This function constructs a rejection sampler whose density is piecewise Gaussian. The sampler is adaptive, gradually building up a piecewise linear upper bound on `H`.

If the `prev` argument is used, then the envelope from the previous call will be used. The `mu` and `sigma` arguments are not re-used from call to call.

Value

A vector of the sampled values, with attribute `env` containing variables describing the envelope function. The ones that would be likely to be of use to callers of this function are:

`bounds, H, mu, sigma`	Copies of the arguments from the call
`f`	A function returning the target density (unnormalized)
`U`	A function returning the upper envelope on `H`
`fbound`	A function returning the upper envelope on `f`
`trials`	How many attempts were made before a value was accepted for each sample

Other variables are returned for internal use.

Author(s)

Duncan Murdoch

Examples

```
H <- function(x) sin(5 * x)
bounds <- c(-5, 5)

x <- rNearGaussian(10, H, bounds, sigma = 0.5)
par(mfrow=c(2,1))
env <- attr(x, "env")
plot(env$U, from=-3, to=3, n=1000, col="red", ylab="H")
plot(env$H, from=-3, to=3, n=1000, add=TRUE)

plot(env$fbound, from=-3, to=3, n=1000, col="red", ylab="f")
plot(env$f, from=-3, to=3, n=1000, add=TRUE)

rug(x)
env$trials
```

```
x <- rNearGaussian(10, sigma=0.5, prev=x)
env <- attr(x, "env")
env$trials
```

`tDiff` *Compute some basis points for a realisation of a diffusion process.*

Description

Compute some basis points for a realisation of a diffusion process.

Usage

```
tDiff(y, eps = NULL)
```

Arguments

`y`	A vector of values at which the value of diffusion process is wanted.
`eps`	A time-step for the stepping between the points of y

Details

TBD. Talk about how the intervals in y are divided by eps when some diff(y) doesn't divide by eps.

Value

A list with components:

`s`	A vector of all the time points involved.
`X`	A boolean vector of the same length as s, TRUE when the corresponding time point in s is in the original y

Author(s)

Barry Rowlingson

Examples

```
tDiff(c(0,1,2),eps=.21)
```

`title.diffusionProcess` *Create a nice title for a diffusion process plot*

Description

This function makes a nice title string for a diffusion process plot, by creating a greek mu and turning the drift component into proper arithmetic. This is mainly used internally by the plot functions of this library.

Usage

```
title.diffusionProcess(x, method="unknown")
```

Arguments

`x`	A diffusion process object

Value

None

Author(s)

Barry Rowlingson

Examples

```
plot(1:10)
title.diffusionProcess(diffusionProcess(sin(Xt)))
```

Index