

Package ‘emhawkes’

October 13, 2022

Title Exponential Multivariate Hawkes Model

Version 0.9.5

Maintainer Kyungsub Lee <kyungsub@gmail.com>

Description Simulate and fitting exponential multivariate Hawkes model.

This package simulates a multivariate Hawkes model, introduced by Hawkes (1971) <doi:10.2307/2334319>, with an exponential kernel and fits the parameters from the data.

Models with the constant parameters, as well as complex dependent structures, can also be simulated and estimated.

The estimation is based on the maximum likelihood method, introduced by introduced by Ozaki (1979) <doi:10.1007/BF02480272>, with 'maxLik' package.

Depends R (>= 3.4.0)

License GPL (>= 2)

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Imports maxLik, methods

Collate 'harrival.R' 'hspec.R' 'hmoment.R' 'hllf.R' 'hfit.R'
'utilities.R' 'hgfit.R' 'hreal.R' 'hsim.R' 'script.R'

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Kyungsub Lee [aut, cre]

Repository CRAN

Date/Publication 2021-02-16 06:40:02 UTC

R topics documented:

hfit	2
hreal	5

hsim	6
hspec-class	7
infer_lambda	8
logLik,hspec-method	9
residual_process	10

Index	12
--------------	-----------

hfit *Perform Maximum Likelihood Estimation*

Description

Generic function hfit. A method for estimating the parameters of the exponential Hawkes model. The reason for being constructed as the S4 method is as follows. First, to represent the structure of the model as an hspec object. There are numerous variations on the multivariate marked Hawkes model. Second, to convey the starting point of numerical optimization. The parameter values assigned to the hspec slots become initial values. This function uses `maxLik` for the optimizer.

Usage

```
hfit(
  object,
  inter_arrival = NULL,
  type = NULL,
  mark = NULL,
  N = NULL,
  Nc = NULL,
  lambda0 = NULL,
  N0 = NULL,
  mylogLik = NULL,
  reduced = TRUE,
  grad = NULL,
  hess = NULL,
  constraint = NULL,
  method = "BFGS",
  verbose = FALSE,
  ...
)

## S4 method for signature 'hspec'
hfit(
  object,
  inter_arrival = NULL,
  type = NULL,
  mark = NULL,
  N = NULL,
  Nc = NULL,
```

```

lambda0 = NULL,
N0 = NULL,
mylogLik = NULL,
reduced = TRUE,
grad = NULL,
hess = NULL,
constraint = NULL,
method = "BFGS",
verbose = FALSE,
...
)

```

Arguments

object	hspec-class . This object includes the parameter values
inter_arrival	inter-arrival times of events. Includes inter-arrival for events that occur in all dimensions. Start with zero.
type	a vector of dimensions. Distinguished by numbers, 1, 2, 3, and so on. Start with zero.
mark	a vector of mark (jump) sizes. Start with zero.
N	a matrix of counting processes
Nc	a matrix of cumulated counting processes
lambda0	the initial values of lambda component. Must have the same dimensional matrix (n by n) with hspec.
N0	the initial values of N.
mylogLik	user defined log likelihood function. mylogLik function should have 'object' argument, consistent with hspec.
reduced	When TRUE, reduced estimation performed.
grad	gradient matrix for the likelihood function. For more information, see maxLik .
hess	Hessian matrix for the likelihood function. For more information, see maxLik .
constraint	constraint matrix. For more information, see maxLik .
method	method for optimization. For more information, see maxLik .
verbose	If TRUE, print the progress of the estimation.
...	other parameters for optimization. For more information, see maxLik .

See Also

[hspec-class](#), [hsim](#), [hspec-method](#)

Examples

```

# example 1
mu <- c(0.1, 0.1)
alpha <- matrix(c(0.2, 0.1, 0.1, 0.2), nrow=2, byrow=TRUE)

```

```

beta <- matrix(c(0.9, 0.9, 0.9, 0.9), nrow=2, byrow=TRUE)
h <- new("hspec", mu=mu, alpha=alpha, beta=beta)
res <- hsim(h, size=100)
summary(hfit(h, res$inter_arrival, res$type))

# example 2

mu <- matrix(c(0.08, 0.08, 0.05, 0.05), nrow = 4)
alpha <- function(param = c(alpha11 = 0, alpha12 = 0.4, alpha33 = 0.5, alpha34 = 0.3)){
  matrix(c(param["alpha11"], param["alpha12"], 0, 0,
           param["alpha12"], param["alpha11"], 0, 0,
           0, 0, param["alpha33"], param["alpha34"],
           0, 0, param["alpha34"], param["alpha33"]), nrow = 4, byrow = TRUE)
}
beta <- matrix(c(rep(0.6, 8), rep(1.2, 8)), nrow = 4, byrow = TRUE)

impact <- function(param = c(alpha1n=0, alpha1w=0.2, alpha2n=0.001, alpha2w=0.1),
                       n=n, N=N, ...){

  Psi <- matrix(c(0, 0, param['alpha1w'], param['alpha1n'],
                 0, 0, param['alpha1n'], param['alpha1w'],
                 param['alpha2w'], param['alpha2n'], 0, 0,
                 param['alpha2n'], param['alpha2w'], 0, 0), nrow=4, byrow=TRUE)

  ind <- N[, "N1"][n] - N[, "N2"][n] > N[, "N3"][n] - N[, "N4"][n] + 0.5

  km <- matrix(c(!ind, !ind, !ind, !ind,
                ind, ind, ind, ind,
                ind, ind, ind, ind,
                !ind, !ind, !ind, !ind), nrow = 4, byrow = TRUE)

  km * Psi
}
h <- new("hspec",
        mu = mu, alpha = alpha, beta = beta, impact = impact)
hr <- hsim(h, size=100)
plot(hr$arrival, hr$N[, 'N1'] - hr$N[, 'N2'], type='s')
lines(hr$N[, 'N3'] - hr$N[, 'N4'], type='s', col='red')
fit <- hfit(h, hr$inter_arrival, hr$type)
summary(fit)

# example 3

mu <- c(0.15, 0.15)
alpha <- matrix(c(0.75, 0.6, 0.6, 0.75), nrow=2, byrow=TRUE)
beta <- matrix(c(2.6, 2.6, 2.6, 2.6), nrow=2, byrow=TRUE)
rmark <- function(param = c(p=0.65), ...){
  rgeom(1, p=param[1]) + 1
}
impact <- function(param = c(eta1=0.2), alpha, n, mark, ...){
  ma <- matrix(rep(mark[n]-1, 4), nrow = 2)

```

```

    alpha * ma * matrix( rep(param["eta1"], 4), nrow=2)
  }
h1 <- new("hspec", mu=mu, alpha=alpha, beta=beta,
         rmark = rmark,
         impact=impact)
res <- hsim(h1, size=100, lambda0 = matrix(rep(0.1,4), nrow=2))

fit <- hfit(h1,
           inter_arrival = res$inter_arrival,
           type = res$type,
           mark = res$mark,
           lambda0 = matrix(rep(0.1,4), nrow=2))
summary(fit)

# For more information, please see vignettes.

```

hreal

Generics for hreal

Description

Generic functions list for hreal:

- Print the realization of the Hawkes model.
- Print the summary of the Hawkes process realization.
- Matrix like output of the realization of Hawkes model.

Usage

```

## S3 method for class 'hreal'
print(x, n = 20, ...)

## S3 method for class 'hreal'
summary(object, n = 20, ...)

## S3 method for class 'hreal'
as.matrix(x, ...)

```

Arguments

x	S3-object of hreal.
n	number of rows to display.
...	further arguments passed to or from other methods.
object	S3-object of hreal.

 hsim

Simulate a multivariate Hawkes process. Generic function hsim.

Description

Simulate a multivariate Hawkes process. Generic function hsim.

The method simulate multivariate Hawkes processes. The object `hspec-class` contains the parameter values such as `mu`, `alpha`, `beta`. The mark (jump) structure may or may not be included. It returns an object of class `hreal` which contains `inter_arrival`, `arrival`, `type`, `mark`, `N`, `Nc`, `lambda`, `lambda_component`, `rambda`, `rambda_component`.

Usage

```
hsim(object, size = 100, lambda0 = NULL, N0 = NULL)
```

```
## S4 method for signature 'hspec'
hsim(object, size = 100, lambda0 = NULL, N0 = NULL)
```

Arguments

<code>object</code>	<code>hspec-class</code> . This object includes the parameter values.
<code>size</code>	the number of observations.
<code>lambda0</code>	the starting values of lambda component. numeric or matrix.
<code>N0</code>	the starting values of N

Value

`hreal` S3-object, summary of the realization of the Haweks model

Examples

```
mu <- c(0.1, 0.1)
alpha <- matrix(c(0.2, 0.1, 0.1, 0.2), nrow=2, byrow=TRUE)
beta <- matrix(c(0.9, 0.9, 0.9, 0.9), nrow=2, byrow=TRUE)
h <- new("hspec", mu=mu, alpha=alpha, beta=beta)
res <- hsim(h, size=100)
```

hspec-class

An S4 class to represent an exponential marked Hawkes model

Description

This class represents a specification of a marked Hawkes model with exponential kernel. The intensity of the ground process is defined by:

$$\lambda(t) = \mu + \int (\alpha + \Psi) * \exp(-\beta(t-u)) dN(u).$$

For more details, please see the vignettes.

Details

μ is base intensity. This is generally a constant vector but can be extended to stochastic processes. Currently, piecewise constant μ is also possible. μ is left continuous.

α is a constant matrix which represents impacts on intensities after events. It is represented by slot `mu`.

Ψ is for non-constant parts of the impact. It may depend on any information generated by N , λ , k and so on. It is represented by slot `impact`.

β is a constant matrix for exponential decay rates. It is represented by slot `beta`.

k is mark and represented by slot `rmark`.

`mu`, `alpha` and `beta` are required slots for every exponential Hawkes model. `rmark` and `impact` are additional slots.

Slots

`mu` numeric value or matrix or function, if numeric, automatically converted to matrix

`alpha` numeric value or matrix or function, if numeric, automatically converted to matrix, exciting term

`beta` numeric value or matrix or function, if numeric,, automatically converted to matrix, exponential decay

`dimens` dimension of the model

`rmark` a function that generates mark for counting process, for simulation

`dmark` a density function for mark, for estimation

`impact` a function that describe the after impact of mark to lambda

`type_col_map` used for multiple kernel

Examples

```
MU <- matrix(c(0.2), nrow = 2)
ALPHA <- matrix(c(0.75, 0.92, 0.92, 0.75), nrow = 2, byrow=TRUE)
BETA <- matrix(c(2.25, 2.25, 2.25, 2.25), nrow = 2, byrow=TRUE)
mhspec2 <- new("hspec", mu=MU, alpha=ALPHA, beta=BETA)
```

infer_lambda

*Infer lambda process with given Hawkes model and realized path***Description**

This method compute the inferred lambda process and returns it as hreal form. If we have realized path of Hawkes process and its parameter value, then we can compute the inferred lambda processes. Similarly with other method such as hfit, the input arguments are inter_arrival, type, mark, or equivalently, N and Nc.

Usage

```
infer_lambda(
  object,
  inter_arrival = NULL,
  type = NULL,
  mark = NULL,
  N = NULL,
  Nc = NULL,
  lambda0 = NULL,
  N0 = NULL
)

## S4 method for signature 'hspec'
infer_lambda(
  object,
  inter_arrival = NULL,
  type = NULL,
  mark = NULL,
  N = NULL,
  Nc = NULL,
  lambda0 = NULL,
  N0 = NULL
)
```

Arguments

object	hspec-class . This object includes the parameter values.
inter_arrival	inter-arrival times of events. Includes inter-arrival for events that occur in all dimensions. Start with zero.
type	a vector of dimensions. Distinguished by numbers, 1, 2, 3, and so on. Start with zero.
mark	a vector of mark (jump) sizes. Start with zero.
N	Hawkes process. if not provided, then generate using inter_arrival and type.
Nc	mark accumulated Hawkes process. if not provided, then generate using inter_arrival, type and mark.

lambda0	the initial values of lambda component. Must have the same dimensional matrix (n by n) with hspec.
N0	the initial values of N.

Value

hreal S3-object, the Haweks model with inferred intensity, lambda

Examples

```
mu <- c(0.1, 0.1)
alpha <- matrix(c(0.2, 0.1, 0.1, 0.2), nrow=2, byrow=TRUE)
beta <- matrix(c(0.9, 0.9, 0.9, 0.9), nrow=2, byrow=TRUE)
h <- new("hspec", mu=mu, alpha=alpha, beta=beta)
res <- hsim(h, size=100)
res2 <- infer_lambda(h, res$inter_arrival, res$type)
```

logLik,hspec-method *Compute the loglikelihood function*

Description

The loglikelihood of the ground process of the Hawkes model. (The estimation for jump distribution is not provided.)

Usage

```
## S4 method for signature 'hspec'
logLik(
  object,
  inter_arrival,
  type = NULL,
  mark = NULL,
  N = NULL,
  Nc = NULL,
  N0 = NULL,
  lambda0 = NULL
)
```

Arguments

object	hspec-class . The parameter values in the object are used to compute the log-likelihood.
inter_arrival	a vector of realized inter-arrival times of events. Includes inter-arrival for events that occur in all dimensions. Start with zero.
type	a vector of realized dimensions. Distinguished by numbers, 1, 2, 3, and so on. Start with zero.

mark	a vector of realized mark (jump) sizes. Start with zero.
N	a matrix of counting processes
Nc	a matrix of cumulated counting processes
N0	the initial value of N
lambda0	the initial values of lambda component. Must have the same dimensional matrix with hspec.

See Also

[hspec-class](#), [hfit](#), [hspec-method](#)

residual_process	<i>Compute residual process</i>
------------------	---------------------------------

Description

Using random time change, this function compute the residual process, which is the inter-arrival time of a standard Poisson process. Therefore, the return values should follow the exponential distribution with rate 1, if model and rambda are correctly specified.

Usage

```
residual_process(
  component,
  type,
  inter_arrival,
  rambda_component,
  mu,
  beta,
  dims = NULL,
  mark = NULL,
  N = NULL,
  Nc = NULL,
  lambda0 = NULL,
  N0 = NULL
)
```

Arguments

component	the component of type to get the residual process
type	a vector of types. Distinguished by numbers, 1, 2, 3, and so on. Start with zero.
inter_arrival	inter-arrival times of events. Includes inter-arrival for events that occur in all dimensions. Start with zero.
rambda_component	right continuous version of lambda process

mu	numeric value or matrix or function, if numeric, automatically converted to matrix
beta	numeric value or matrix or function, if numeric, automatically converted to matrix, exponential decay
dimens	dimension of the model. if omitted, set to be the length of mu.
mark	a vector of realized mark (jump) sizes. Start with zero.
N	a matrix of counting processes
Nc	a matrix of cumulated counting processes
lambda0	the initial values of lambda component. Must have the same dimensional matrix with hspec.
N0	the initial value of N

Examples

```
mu <- c(0.1, 0.1)
alpha <- matrix(c(0.2, 0.1, 0.1, 0.2), nrow=2, byrow=TRUE)
beta <- matrix(c(0.9, 0.9, 0.9, 0.9), nrow=2, byrow=TRUE)
h <- new("hspec", mu=mu, alpha=alpha, beta=beta)
res <- hsim(h, size=1000)
rp <- residual_process(1, res$type, res$inter_arrival, res$lambda_component, mu, beta)
p <- ppoints(100)
q <- quantile(rp,p=p)
plot(qexp(p), q, xlab="Theoretical Quantiles",ylab="Sample Quantiles")
qqline(q, distribution=qexp,col="blue", lty=2)
```

Index

`as.matrix.hreal` (`hreal`), 5

`hfit`, 2

`hfit`, `hspec-method` (`hfit`), 2

`hreal`, 5

`hsim`, 6

`hsim`, `hspec-method` (`hsim`), 6

`hspec-class`, 7

`infer_lambda`, 8

`infer_lambda`, `hspec-method`
(`infer_lambda`), 8

`logLik`, `hspec-method`, 9

`maxLik`, 2, 3

`print.hreal` (`hreal`), 5

`residual_process`, 10

`summary.hreal` (`hreal`), 5