Package 'matrixdist'

February 26, 2021

Type Package

Title Statistics for Matrix Distributions

Version 1.0.2

Date 2021-02-26

Maintainer Martin Bladt <martinbladt@gmail.com>

Description Tools for homogeneous and in-homogeneous phase-type distributions.
Methods for functional evaluation, simulation and estimation using the
expectation-maximization (EM) algorithm are provided.
The methods of this package are based on the following references.
Asmussen, S., Nerman, O., & Olsson, M. (1996) <https://www.jstor.org/stable/4616418>,
Olsson, M. (1996) <https://www.jstor.org/stable/4616419>.
Albrecher, H., & Bladt, M. (2019) <doi:10.1017/jpr.2019.60>
Albrecher, H., Bladt, M., & Yslas, J. (2020) <doi:10.1111/sjos.12505>
Bladt, M., & Yslas, J. (2020) <arXiv:2011.03219>.

Depends R (>= 3.1.0)

License GPL-3

Imports Rcpp, methods

LinkingTo Rcpp

SystemRequirements C++11

Encoding UTF-8

RoxygenNote 7.1.1

NeedsCompilation yes

Author Martin Bladt [aut, cre],
Jorge Yslas [aut]

Repository CRAN

Date/Publication 2021-02-26 09:50:08 UTC

R topics documented:

`matrixdist-package` *Statistics for Matrix Distributions*

Description

This package is concerned with homogeneous and inhomogeneous phase–type distributions. Methods for functional evaluation, simulation and estimation using the EM algorithm are provided.

Author(s)

Martin Bladt and Jorge Yslas.

Maintainer: Martin Bladt <martinbladt@gmail.com>

References

Asmussen, S., Nerman, O., & Olsson, M. (1996). Fitting phase-type distributions via the EM algorithm. Scandinavian Journal of Statistics, 419-441.

Olsson, M. (1996). Estimation of phase-type distributions from censored data. Scandinavian journal of statistics, 443-460.

Albrecher, H., & Bladt, M. (2019). Inhomogeneous phase-type distributions and heavy tails. Journal of Applied Probability, 56(4), 1044-1064.

Albrecher, H., Bladt, M., & Yslas, J. (2020). Fitting inhomogeneous Phase-Type distributions to data: The univariate and the multivariate case. Scandinavian Journal of Statistics.

Bladt, M., & Yslas, J. (2020). Inhomogeneous Markov Survival Regression Models. arXiv:2011.03219.

`+,ph,ph-method` *Sum Method for phase type distributions*

Description

Sum Method for phase type distributions

Usage

```
## S4 method for signature 'ph,ph'
e1 + e2
```

Arguments

`e1`	an object of class ph.
`e2`	an object of class ph.

Value

An object of class ph.

Examples

```
ph1 <- ph(structure = "general", dimension = 3)
ph2 <- ph(structure = "gcoxian", dimension = 5)
ph_sum <- ph1 + ph2
ph_sum
```

a_rungekutta *Runge Kutta for the calculation of the a vectors in a EM step*

Description

Can be used for the loglikelihood

Usage

```
a_rungekutta(avector, dt, h, S)
```

Arguments

avector	the a vector
dt	increment
h	step-length
S	sub-intensity

cdf *New Generic for the Distribution of Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
cdf(x, ...)
```

Arguments

x	an object of the model class.
...	further parameters to be passed on

Value

CDF from the matrix distribution.

cdf,iph-method *Distribution Method for inhomogeneous phase type distributions*

Description

Distribution Method for inhomogeneous phase type distributions

Usage

```
## S4 method for signature 'iph'
cdf(x, q = seq(0, quan(x, 0.95)$quantile, length.out = 10), lower.tail = TRUE)
```

Arguments

x	an object of class iph.
q	a vector of locations.
lower.tail	logical parameter specifying whether lower tail (cdf) or upper tail is computed.

Value

A list containing the locations and corresponding CDF evaluations.

Examples

```
obj <- iph(ph(structure = "general"), gfun = "weibull", gfun_pars = 2)
cdf(obj, c(1, 2, 3))
```

cdf,ph-method *Distribution Method for phase type distributions*

Description

Distribution Method for phase type distributions

Usage

```
## S4 method for signature 'ph'
cdf(x, q = seq(0, quan(x, 0.95)$quantile, length.out = 10), lower.tail = TRUE)
```

Arguments

x	an object of class ph.
q	a vector of locations.
lower.tail	logical parameter specifying whether lower tail (cdf) or upper tail is computed.

Value

A list containing the locations and corresponding CDF evaluations.

Examples

```
obj <- ph(structure = "general")
cdf(obj, c(1, 2, 3))
```

`clone_matrix` *Clone a matrix*

Description

Clone a matrix

Usage

```
clone_matrix(m)
```

Arguments

`m`	a matrix

`clone_vector` *Clone a vector*

Description

Clone a vector

Usage

```
clone_vector(v)
```

Arguments

`v`	a vector

coef,iph-method *Coef Method for iph Class*

Description

Coef Method for iph Class

Usage

```
## S4 method for signature 'iph'
coef(object)
```

Arguments

object	an object of class iph.

Value

parameters of iph model.

Examples

```
obj <- iph(ph(structure = "general", dimension = 2), gfun = "lognormal", gfun_pars = 2)
coef(obj)
```

coef,ph-method *Coef Method for ph Class*

Description

Coef Method for ph Class

Usage

```
## S4 method for signature 'ph'
coef(object)
```

Arguments

object	an object of class ph.

Value

Parameters of ph model.

Examples

```
obj <- ph(structure = "general")
coef(obj)
```

cumulateMatrix *Cumulate matrix*

Description

Creates a new matrix with entries the cumulated rows of `A`

Usage

```
cumulateMatrix(A)
```

Arguments

`A`	A matrix

Value

The cumulated matrix

cumulateVector *Cumulate vector*

Description

Creates a new vector with entries the cumulated entries of `A`

Usage

```
cumulateVector(A)
```

Arguments

`A`	A vector

Value

The cumulated vector

default_step_length *Default size of the steps in the RK*

Description

Computes the default step length for a matrix `S` to be employed in the RK method

Usage

```
default_step_length(S)
```

Arguments

`S`	sub-intensity matrix

Value

The step length for `S`

dens *New Generic for the Density of Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
dens(x, ...)
```

Arguments

`x`	an object of the model class.
`...`	further parameters to be passed on

Value

Density from the matrix distribution.

dens,iph-method *Density Method for inhomogeneous phase type distributions*

Description

Density Method for inhomogeneous phase type distributions

Usage

```
## S4 method for signature 'iph'
dens(x, y = seq(0, quan(x, 0.95)$quantile, length.out = 10))
```

Arguments

x	an object of class iph.
y	a vector of locations.

Value

A list containing the locations and corresponding density evaluations.

Examples

```
obj <- iph(ph(structure = "general"), gfun = "weibull", gfun_pars = 2)
dens(obj, c(1, 2, 3))
```

dens,ph-method *Density Method for phase type distributions*

Description

Density Method for phase type distributions

Usage

```
## S4 method for signature 'ph'
dens(x, y = seq(0, quan(x, 0.95)$quantile, length.out = 10))
```

Arguments

x	an object of class ph.
y	a vector of locations.

Value

A list containing the locations and corresponding density evaluations.

Examples

```
obj <- ph(structure = "general")
dens(obj, c(1, 2, 3))
```

`derivativeMatrixweibull` *Derivative of matrix Weibull*

Description

Can be used to increase performance

Usage

```
derivativeMatrixweibull(h, obs, weight, rcens, rcweight, alpha, S, beta)
```

Arguments

`h`	step-length
`obs`	the observations
`weight`	weight of the observations
`rcens`	censored observations
`rcweight`	weight of the censored observations
`alpha`	initial probabilities
`S`	sub-intensity
`beta`	parameter of transformation

`diagonal_vector` *Creates a matrix with the given vector in the diagonal*

Description

Creates a matrix with the given vector in the diagonal

Usage

```
diagonal_vector(vec)
```

Arguments

`vec`	a vector

embeddedMC — *Embedded Markov chain of a sub-intensity matrix*

Description

Returns the transition probabilities of the embedded Markov chain determined the sub-intensity matrix

Usage

```
embeddedMC(S)
```

Arguments

S	A sub-intensity matrix

Value

The embedded Markov chain

EMstep_RK — *EM step using Runge Kutta*

Description

Computes one step of the EM algorithm by using a Runge-Kutta method of 4th order

Usage

```
EMstep_RK(h, alpha, S, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probalities
S	sub-intensity
obs	the observations
weight	the weights for the observations
rcens	censored observations
rcweight	the weights for the censored observations

`fit` *New Generic for Estimating Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
fit(x, y, ...)
```

Arguments

`x`	an object of the model class.
`y`	a vector of data.
`...`	further parameters to be passed on

Value

An object of the fitted model class.

`fit,ph-method` *Fit Method for ph Class*

Description

Fit Method for ph Class

Usage

```
## S4 method for signature 'ph'
fit(
  x,
  y,
  weight = numeric(0),
  rcen = numeric(0),
  rcenweight = numeric(0),
  stepsEM = 1000,
  rkstep = NA,
  maxit = 100,
  reltol = 1e-08,
  every = 100
)
```

Arguments

x	an object of class ph.
y	vector or data.
weight	vector of weights.
rcen	vector of right-censored observations
rcenweight	vector of weights for right-censored observations.
stepsEM	number of EM steps to be performed.
rkstep	Runge-Kutta step size (optional)
maxit	maximum number of iterations when optimizing g function.
reltol	relative tolerance when optimizing g function.
every	number of iterations between likelihood display updates.

Value

An object of class ph.

Examples

```
obj <- iph(ph(structure = "general", dimension = 2), gfun = "weibull", gfun_pars = 2)
data <- sim(obj, n = 100)
fit(obj, data, stepsEM = 1000, every = 200)
```

---

haz *New Generic for the Hazard rate of Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
haz(x, ...)
```

Arguments

x	an object of the model class.
...	further parameters to be passed on

Value

Hazard rate from the matrix distribution.

haz,ph-method *Hazard rate Method for phase type distributions*

Description

Hazard rate Method for phase type distributions

Usage

```
## S4 method for signature 'ph'
haz(x, y = seq(0, quan(x, 0.95)$quantile, length.out = 10))
```

Arguments

x	an object of class ph.
y	a vector of locations.

Value

A list containing the locations and corresponding hazard rate evaluations.

Examples

```
obj <- ph(structure = "general")
haz(obj, c(1, 2, 3))
```

initialState *Initial state of Markov jump process*

Description

Given the accumulated values of the initial probabilities Pi and a uniform value u, it returns the initial state of a Markov jump process

Usage

```
initialState(cumulatedPi, u)
```

Arguments

cumulatedPi	A vector
u	A random value in (0,1)

Value

The initial state of the Markov jump process

iph *Constructor Function for inhomogeneous phase type distributions*

Description

Constructor Function for inhomogeneous phase type distributions

Usage

```
iph(
  ph = NULL,
  gfun = NULL,
  gfun_pars = NULL,
  alpha = NULL,
  S = NULL,
  structure = NULL,
  dimension = 3,
  scale = 1
)
```

Arguments

ph	An object of class ph.
gfun	inhomogeneity transform
gfun_pars	the parameters of the inhomogeneity function
alpha	a probability vector.
S	a sub-intensity matrix.
structure	a valid ph structure
dimension	the dimension of the ph structure (if provided)
scale	scale

Value

An object of class iph.

Examples

```
iph(ph(structure = "coxian", dimension = 4), gfun = "pareto", gfun_pars = 3)
```

iph-class *Inhomogeneous Phase Type distributions*

Description

Class of objects for inhomogeneous phase type distributions

Value

Class object

Slots

`name` name of the phase type distribution.

`gfun` a list comprising of the parameters.

`scale` scale.

LInf_norm *L-oo norm of a matrix*

Description

Computes the L-oo norm of a matrix `A`, which is defined as: L-oo A = max (1 <= I <= M) sum (1 <= J <= N) abs (A(I,J)).

Usage

```
LInf_norm(A)
```

Arguments

`A`	A matrix

logLik,ph-method *logLik Method for ph Class*

Description

logLik Method for ph Class

Usage

```
## S4 method for signature 'ph'
logLik(object)
```

Arguments

object	an object of class ph.

Value

An object of class logLik.

Examples

```
obj <- iph(ph(structure = "general", dimension = 2), gfun = "weibull", gfun_pars = 2)
data <- sim(obj, n = 100)
fitted_ph <- fit(obj, data, stepsEM = 10)
logLik(fitted_ph)
```

logLikelihoodMgev_RK *Loglikelihood of matrix GEV using RK*

Description

Loglikelihood for a sample

Usage

```
logLikelihoodMgev_RK(h, alpha, S, beta, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probabilities
S	sub-intensity
beta	parameter of transformation
obs	the observations

weight	weight of the observations
rcens	censored observations
rcweight	weight of the censored observations

---

logLikelihoodMgompertz_RK

Loglikelihood of matrix Gompertz using RK

Description

Loglikelihood for a sample

Usage

```
logLikelihoodMgompertz_RK(h, alpha, S, beta, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probabilities
S	sub-intensity
beta	parameter of transformation
obs	the observations
weight	weight of the observations
rcens	censored observations
rcweight	weight of the censored observations

---

logLikelihoodMloglogistic_RK

Loglikelihood of matrix Log-Logistic using RK

Description

Loglikelihood for a sample

Usage

```
logLikelihoodMloglogistic_RK(h, alpha, S, beta, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probabilities
S	sub-intensity
beta	parameter of transformation
obs	the observations
weight	weight of the observations
rcens	censored observations
rcweight	weight of the censored observations

---

logLikelihoodMlognormal_RK

Loglikelihood of matrix LogNormal using RK

Description

Loglikelihood for a sample

Usage

```
logLikelihoodMlognormal_RK(h, alpha, S, beta, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probabilities
S	sub-intensity
beta	parameter of transformation
obs	the observations
weight	weight of the observations
rcens	censored observations
rcweight	weight of the censored observations

logLikelihoodMpareto_RK

Loglikelihood of matrix Pareto using RK

Description

Loglikelihood for a sample

Usage

```
logLikelihoodMpareto_RK(h, alpha, S, beta, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probabilities
S	sub-intensity
beta	parameter of transformation
obs	the observations
weight	weight of the observations
rcens	censored observations
rcweight	weight of the censored observations

logLikelihoodMweibull_RK

Loglikelihood of matrix Weibull using RK

Description

Loglikelihood for a sample

Usage

```
logLikelihoodMweibull_RK(h, alpha, S, beta, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probabilities
S	sub-intensity
beta	parameter of transformation
obs	the observations
weight	weight of the observations
rcens	censored observations
rcweight	weight of the censored observations

logLikelihoodPH_RK *Loglikelihood using RK*

Description

Loglikelihood for a sample

Usage

```
logLikelihoodPH_RK(h, alpha, S, obs, weight, rcens, rcweight)
```

Arguments

h	step-length
alpha	initial probabilities
S	sub-intensity
obs	the observations
weight	weight of the observations
rcens	censored observations
rcweight	weight of the censored observations

matrixMax *Maximum entry in a matrix*

Description

Find the maximum entry

Usage

```
matrixMax(A)
```

Arguments

A	a matrix

matrixMaxDiagonal *Maximum entry in the diagonal of a matrix*

Description

Maximum entry in the diagonal of a matrix

Usage

```
matrixMaxDiagonal(A)
```

Arguments

A	a matrix

matrix_exponential *Matrix exponential algorithm*

Description

MATLAB's built-in algorithm - Pade approximation

Usage

```
matrix_exponential(A)
```

Arguments

A	a matrix

matrix_inverse *Inverse of a matrix*

Description

Computes the inverse

Usage

```
matrix_inverse(A)
```

Arguments

A	a matrix

matrix_power *Computes A^n*

Description

Computes A^n

Usage

```
matrix_power(n, A)
```

Arguments

n	integer
A	a matrix

matrix_product *Product of two matrices*

Description

Product of two matrices

Usage

```
matrix_product(a, b)
```

Arguments

a	matrix
b	matrix

Value

Computes c = a * b

matrix_sum *Add matrices*

Description

Computes C = A + B

Usage

```
matrix_sum(A, B)
```

Arguments

A	A matrix
B	A matrix

matrix_VanLoan *Creates the matrix (A1, B1 ; 0, A2)*

Description

Creates the matrix (A1, B1 ; 0, A2)

Usage

```
matrix_VanLoan(A1, A2, B1)
```

Arguments

A1	a matrix
A2	a matrix
B1	a matrix

maximum *New Generic for Maximum of two Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
maximum(x1, x2, ...)
```

Arguments

x1	an object of the model class.
x2	an object of the model class.
...	further parameters to be passed on

Value

A realization from the matrix distribution.

maximum,iph,iph-method *Maximum Method for inhomogeneous phase type distributions*

Description

Maximum Method for inhomogeneous phase type distributions

Usage

```
## S4 method for signature 'iph,iph'
maximum(x1, x2)
```

Arguments

x1	an object of class iph.
x2	an object of class iph.

Value

An object of class iph.

Examples

```
iph1 <- iph(ph(structure = "general", dimension = 3), gfun = "weibull", gfun_pars = 2)
iph2 <- iph(ph(structure = "gcoxian", dimension = 5), gfun = "weibull", gfun_pars = 2)
iph_min <- maximum(iph1, iph2)
iph_min
```

---

maximum,ph,ph-method *Maximum Method for phase type distributions*

Description

Maximum Method for phase type distributions

Usage

```
## S4 method for signature 'ph,ph'
maximum(x1, x2)
```

Arguments

x1	an object of class ph.
x2	an object of class ph.

Value

An object of class ph.

Examples

```
ph1 <- ph(structure = "general", dimension = 3)
ph2 <- ph(structure = "gcoxian", dimension = 5)
ph_min <- minimum(ph1, ph2)
ph_min
```

---

mgevcdf *Matrix GEV cdf*

Description

Computes the cdf (tail) of a matrix GEV distribution with parameters alpha, S and beta at x

Usage

```
mgevcdf(x, alpha, S, mu, sigma, xi, lower_tail = TRUE)
```

Arguments

x	non-negative value
alpha	Initial probabilities
S	sub-intensity matrix
mu	location parameter
sigma	scale parameter
xi	shape parameter
lower_tail	cdf or tail

Value

The cdf (tail) at x

---

mgevden *Matrix GEV density*

Description

Computes the density of a matrix GEV distribution with parameters alpha, S and beta at x Dont allow for atoms in zero

Usage

```
mgevden(x, alpha, S, mu, sigma, xi)
```

Arguments

x	non-negative value
alpha	Initial probabilities
S	sub-intensity matrix
mu	location parameter
sigma	scale parameter
xi	shape parameter

Value

The density at x

mgompertzcdf *Matrix Gompertz cdf*

Description

Computes the cdf (tail) of a matrix Gompertz distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mgompertzcdf(x, alpha, S, beta, lower_tail = TRUE)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	shape parameter
`lower_tail`	cdf or tail

Value

The cdf (tail) at `x`

mgompertzden *Matrix Gompertz density*

Description

Computes the density of a matrix Gompertz distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mgompertzden(x, alpha, S, beta)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	parameter

Value

The density at `x`

minimum *New Generic for Minimum of two Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
minimum(x1, x2, ...)
```

Arguments

x1	an object of the model class.
x2	an object of the model class.
...	further parameters to be passed on

Value

A realization from the matrix distribution.

minimum,iph,iph-method *Minimum Method for inhomogeneous phase type distributions*

Description

Minimum Method for inhomogeneous phase type distributions

Usage

```
## S4 method for signature 'iph,iph'
minimum(x1, x2)
```

Arguments

x1	an object of class iph.
x2	an object of class iph.

Value

An object of class iph.

Examples

```
iph1 <- iph(ph(structure = "general", dimension = 3), gfun = "weibull", gfun_pars = 2)
iph2 <- iph(ph(structure = "gcoxian", dimension = 5), gfun = "weibull", gfun_pars = 2)
iph_min <- minimum(iph1, iph2)
iph_min
```

`minimum,ph,ph-method` *Minimum Method for phase type distributions*

Description

Minimum Method for phase type distributions

Usage

```
## S4 method for signature 'ph,ph'
minimum(x1, x2)
```

Arguments

`x1`	an object of class ph.
`x2`	an object of class ph.

Value

An object of class ph.

Examples

```
ph1 <- ph(structure = "general", dimension = 3)
ph2 <- ph(structure = "gcoxian", dimension = 5)
ph_min <- minimum(ph1, ph2)
ph_min
```

`mloglogisticcdf` *Matrix Log-Logistic cdf*

Description

Computes the cdf (tail) of a matrix Log-Logistic distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mloglogisticcdf(x, alpha, S, beta, lower_tail = TRUE)
```

Arguments

x	non-negative value
alpha	Initial probabilities
S	sub-intensity matrix
beta	shape parameter
lower_tail	cdf or tail

Value

The cdf (tail) at x

---

mloglogisticden *Matrix Log-Logistic density*

Description

Computes the density of a matrix Log-Logistic distribution with parameters alpha, S and beta at x

Usage

```
mloglogisticden(x, alpha, S, beta)
```

Arguments

x	non-negative value
alpha	Initial probabilities
S	sub-intensity matrix
beta	scale parameter

Value

The density at x

mlognormalcdf *Matrix LogNormal cdf*

Description

Computes the cdf (tail) of a matrix LogNormal distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mlognormalcdf(x, alpha, S, beta, lower_tail = TRUE)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	shape parameter
`lower_tail`	cdf or tail

Value

The cdf (tail) at `x`

mlognormalden *Matrix LogNormal density*

Description

Computes the density of a matrix LogNormal distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mlognormalden(x, alpha, S, beta)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	shape parameter

Value

The density at `x`

moment *New Generic for Moment of Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
moment(x, ...)
```

Arguments

x	an object of the model class.
...	further parameters to be passed on

Value

A realization from the matrix distribution.

moment,ph-method *Moment Method for phase type distributions*

Description

Moment Method for phase type distributions

Usage

```
## S4 method for signature 'ph'
moment(x, k = 1)
```

Arguments

x	an object of class ph.
k	a positive integer (moment order).

Value

The raw moment of the ph (or undelying ph) object.

Examples

```
set.seed(123)
ph1 <- ph(structure = "general", dimension = 3)
moment(ph1, 2)
```

mparetocdf *Matrix Pareto cdf*

Description

Computes the cdf (tail) of a matrix Pareto distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mparetocdf(x, alpha, S, beta, lower_tail = TRUE)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	shape parameter
`lower_tail`	cdf or tail

Value

The cdf (tail) at `x`

mparetoden *Matrix Pareto density*

Description

Computes the density of a matrix Pareto distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mparetoden(x, alpha, S, beta)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	scale parameter

Value

The density at `x`

mweibullcdf *Matrix Weibull cdf*

Description

Computes the cdf (tail) of a matrix Weibull distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mweibullcdf(x, alpha, S, beta, lower_tail)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	shape parameter
`lower_tail`	cdf or tail

Value

The cdf (tail) at `x`

mweibullden *Matrix Weibull density*

Description

Computes the density of a matrix Weibull distribution with parameters `alpha`, `S` and `beta` at `x`

Usage

```
mweibullden(x, alpha, S, beta)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`beta`	shape parameter

Value

The density at `x`

newState *New state in a Markov jump process*

Description

Given a transition matrix Q, a uniform value u, and a previous state k, it returns the new state of a Markov jump process

Usage

```
newState(previousState, cumulatedEmbeddedMC, u)
```

Arguments

previousState	Previous state of the Markov jump process
cumulatedEmbeddedMC	A transition matrix
u	A random value in (0,1)

Value

The next state of the Markov jump process

ph *Constructor Function for phase type distributions*

Description

Constructor Function for phase type distributions

Usage

```
ph(alpha = NULL, S = NULL, structure = NULL, dimension = 3)
```

Arguments

alpha	a probability vector.
S	a sub-intensity matrix.
structure	a valid ph structure ("general", "coxian", "hyperexponential", "gcoxian", "gerlang").
dimension	the dimension of the ph structure (if structure is provided).

Value

An object of class ph.

Examples

```
ph(structure = "gcoxian", dim = 5)
ph(alpha = c(.5, .5), S = matrix(c(-1, .5, .5, -1), 2, 2))
```

---

ph-class *Phase Type distributions*

Description

Class of objects for phase type distributions

Value

Class object

Slots

`name` name of the phase type distribution.

`pars` a list comprising of the parameters.

`fit` a list containing estimation information.

---

phcdf *Phase-type cdf or tail*

Description

Computes the cdf of phase-type distribution with parameters `alpha` and `S` at `x`

Usage

```
phcdf(x, alpha, S, lower_tail = TRUE)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix
`lower_tail`	cdf or tail

Value

The cdf (tail) at `x`

phdensity *Phase-type density*

Description

Computes the density of phase-type distribution with parameters `alpha` and `S` at `x`

Usage

```
phdensity(x, alpha, S)
```

Arguments

`x`	non-negative value
`alpha`	Initial probabilities
`S`	sub-intensity matrix

Value

The density at `x`

quan *New Generic for the Quantile of Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
quan(x, ...)
```

Arguments

`x`	an object of the model class.
`...`	further parameters to be passed on

Value

Quantile from the matrix distribution.

quan,ph-method *Quantile Method for phase type distributions*

Description

Quantile Method for phase type distributions

Usage

```
## S4 method for signature 'ph'
quan(x, p = seq(0, 1, length.out = 10))
```

Arguments

x	an object of class ph.
p	a vector of probabilities.

Value

A list containing the probabilities and corresponding quantile evaluations.

Examples

```
obj <- ph(structure = "general")
quan(obj, c(0.5, 0.9, 0.99))
```

random_structure *Random structure of a phase-type*

Description

Generates random parameters alpha and S of a phase-type distribution of dimension p with chosen structure

Usage

```
random_structure(p, structure = "general", scale_factor = 1)
```

Arguments

p	Dimension of the phase-type
structure	Type of structure: "general", "hyperexponential", "gerlang", "coxian" or "gcoxian"
scale_factor	A factor that multiplies the sub-intensity matrix

Value

Random parameters alpha and S of a phase-type

reversTransformData *Applies the inverse of the GEV but giving back the vector in reverse order*

Description

Used for EM step

Usage

```
reversTransformData(observations, weights, beta)
```

Arguments

observations	the observations
weights	weithgs of the observations
beta	parameters of the GEV

riph *Random inhomogeneous phase-type*

Description

Generates a sample of size n from an inhomogeneous phase-type distribution with parameters alpha, S and beta

Usage

```
riph(n, dist_type, alpha, S, beta)
```

Arguments

n	Sample size
dist_type	Type of IPH
alpha	Initial probabilities
S	sub-intensity matrix
beta	Parameter of the transformation

Value

The simulated sample

rmatrixgev *Random matrix GEV*

Description

Generates a sample of size n from an inhomogeneous phase-type distribution with parameters alpha, S and beta

Usage

```
rmatrixgev(n, alpha, S, mu, sigma, xi = 0)
```

Arguments

n	Sample size
alpha	Initial probabilities
S	sub-intensity matrix
mu	Location parameter
sigma	Scale parameter
xi	Shape parameter: Default 0 which corresponds to the Gumbel case

Value

The simulated sample

rphasetype *Random phase-type*

Description

Generates a sample of size n from a phase-type distribution with parameters alpha and S

Usage

```
rphasetype(n, alpha, S)
```

Arguments

n	Sample size
alpha	Initial probabilities
S	sub-intensity matrix

Value

The simulated sample

runge_kutta *Runge Kutta for the calculation of the a,b and c vectors in a EM step*

Description

Performce the RK of forth order

Usage

```
runge_kutta(avector, bvector, cmatrix, dt, h, S, t)
```

Arguments

avector	the a vector
bvector	the b vector
cmatrix	the c matrix
dt	the increment
h	step-length
S	sub-intensity
t	exit rates

show,iph-method *Show Method for inhomogeneous phase type distributions*

Description

Show Method for inhomogeneous phase type distributions

Usage

```
## S4 method for signature 'iph'
show(object)
```

Arguments

object	an object of class iph.

show,ph-method *Show Method for phase type distributions*

Description

Show Method for phase type distributions

Usage

```
## S4 method for signature 'ph'
show(object)
```

Arguments

object	an object of class ph.

sim *New Generic for Simulating Matrix Distributions*

Description

Methods are available for objects of class ph

Usage

```
sim(x, ...)
```

Arguments

x	an object of the model class.
...	further parameters to be passed on

Value

A realization from the matrix distribution.

sim,iph-method *Simulation Method for inhomogeneous phase type distributions*

Description

Simulation Method for inhomogeneous phase type distributions

Usage

```
## S4 method for signature 'iph'
sim(x, n = 1000)
```

Arguments

x	an object of class iph.
n	an integer of length of realization.

Value

A realization of independent and identically distributed inhomogeneous phase-type variables.

Examples

```
obj <- iph(ph(structure = "general"), gfun = "lognormal", gfun_pars = 2)
sim(obj, n = 100)
```

sim,ph-method *Simulation Method for phase type distributions*

Description

Simulation Method for phase type distributions

Usage

```
## S4 method for signature 'ph'
sim(x, n = 1000)
```

Arguments

x	an object of class ph.
n	an integer of length of realization.

Value

A realization of independent and identically distributed phase-type variables.

Examples

```
obj <- ph(structure = "general")
sim(obj, n = 100)
```

`solve_linear_system` *Solves a system with multiple right hand sides*

Description

AX=B which can be decomposed as LUX=B and finds X. When B is the identity matrix the solution is the inverse of A

Usage

```
solve_linear_system(A1, B)
```

Arguments

A1	a matrix
B	a matrix

`sumPH` *Computes the initial distribution and sub-intensity of the sum of PH*

Description

Computes the initial distribution and sub-intensity of the sum of PH

Usage

```
sumPH(alpha1, S1, alpha2, S2)
```

Arguments

alpha1	initial distribution
S1	sub-intensity
alpha2	initial distribution
S2	sub-intensity

Index