Package 'ReliabilityTheory'

August 29, 2016

Title Tools for Structural Reliability Analysis

Description A variety of tools useful for performing structural reliability analysis, such as with structure function and system signatures. Plans to expand more widely.

URL http://www.louisaslett.com/

Version 0.1.5

Maintainer Louis Aslett <aslett@stats.ox.ac.uk>

Author Louis Aslett <aslett@stats.ox.ac.uk>

Depends igraph (>= 1.0.1)

Imports sfsmisc, combinat, FRACTION, mcmc, PhaseType (>= 0.1.3), actuar, HI

Suggests testthat

License GPL-2 | GPL-3

ByteCompile yes

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2015-10-14 13:54:57

R topics documented:

`ReliabilityTheory-package`

Structural Reliability Theory Toolbox

Description

A collection of tools for working structural reliability problems, such as catalogues of system signatures and Bayesian inferential functions.

Details

Package:	ReliabilityTheory
Type:	Package
Version:	0.1.5
Date:	2015-10-13
License:	GPL-2 | GPL-3
LazyLoad:	yes

Author(s)

Louis J. M. Aslett, <aslett@stats.ox.ac.uk> (http://www.louisaslett.com)

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

Samaniego, F. J. (2007), *System Signatures and Their Applications in Engineering Reliability*, Springer.

cn02 *Catalogue of Coherent Networks of Order 2*

Description

This data sets provides a catalogue of the network graph, signature and minimal cut-sets of all coherent networks of node order 2.

Usage

```
data(cn02)
```

Format

A list object, one item for each such network. Each item is itself a list, with the elements `$graph`, `$cutsets` and `$signature`.

Source

Derived in the thesis Aslett (2012).

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

cn03 *Catalogue of Coherent Networks of Order 3*

Description

This data sets provides a catalogue of the network graph, signature and minimal cut-sets of all coherent networks of node order 3.

Usage

```
data(cn03)
```

Format

A list object, one item for each such network. Each item is itself a list, with the elements `$graph`, `$cutsets` and `$signature`.

Source

Derived in the thesis Aslett (2012).

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

`computeSystemSignature` *Compute the signature of a system*

Description

The system signature (Samaniego, 2007) is an alternative to the structure function as a starting point for a structural reliability analysis. This automatically computes the signature of the specified system or network. Here, system implies components are unreliable whereas network implies links are unreliable.

Usage

```
computeSystemSignature(graph, cutsets=NULL, frac=FALSE)
computeNetworkSignature(graph, cutsets=NULL, frac=FALSE)
```

Arguments

`graph`	an `igraph` object representing the system or network whose signature is to be computed. There should be two terminal 'dummy' nodes to represent either end of the structure which must be labelled "s" and "t". They are assumed perfectly reliable. See details and examples.
`cutsets`	if the cut-sets of the system or network are already known they may be passed in as a list of numeric vectors. This can save time because cut-set computation is the slowest part of the algorithm. Leaving as NULL causes the function to find the cut sets itself.
`frac`	if TRUE then the function prints out signature elements as fractions rather than returning a decimal signature vector.

Details

The signature of a system is the probability vector $\mathbf{s} = (s_1, \ldots, s_n)$ with elements:

$$s_i = P(T = T_{i:n})$$

where T is the failure time of the system and $T_{i:n}$ is the ith order statistic of the n component failure times. Likewise the network signature is the same but where components are reliable and it is links which fail. See Samaniego (2007) for details.

The system or network is specified by means of an `igraph` object, whereby each end of the system is denoted by nodes names "s" and "t" which are taken to be perfectly reliable. It is easy to construct the appropriate graph representation using the function `graph.formula`.

Value

computeSystemSignature returns a numeric probability vector which is the system/network signature.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Samaniego, F. J. (2007), *System Signatures and Their Applications in Engineering Reliability*, Springer.

See Also

computeSystemSurvivalSignature

Examples

```
# Find the signature of two component series system (which is just s=(1, 0))
computeSystemSignature(graph.formula(s -- 1 -- 2 -- t))

# Find the signature of two component parallel system (which is just s=(0, 1))
computeSystemSignature(graph.formula(s -- 1:2 -- t))

# Find the signature of the five component 'bridge' system (which
# is s=(0, 0.2, 0.6, 0.2, 0))
computeSystemSignature(graph.formula(s -- 1 -- 2 -- t, s -- 3 -- 4 -- t, 1:2 -- 5 -- 3:4))
```

---

computeSystemSurvivalSignature

Compute the survival signature of a system

Description

The system survival signature (Coolen and Coolen-Maturi, 2012) is a generalisation of the signature to systems with multiple component types. This function automatically computes the survival signature of the specified system. Here, system implies components (as opposed to links) are unreliable.

Usage

```
computeSystemSurvivalSignature(graph, cutsets=NULL, frac=FALSE)
```

Arguments

graph	an igraph object representing the system whose survival signature is to be computed. There should be two terminal 'dummy' nodes to represent either end of the structure which must be labelled "s" and "t" (assumed perfectly reliable) as well as a vertex attribute named compType identifying each component type. See details and examples.
cutsets	if the cut-sets of the system or network are already known they may be passed in as a list of numeric vectors. This can save time because cut-set computation is the slowest part of the algorithm. Leaving as NULL causes the function to find the cut sets itself.
frac	if TRUE then the function prints out survival signature probabilities as fractions rather than decimals.

Details

The survival signature of a system with K types of component is the functional $\Phi(l_1, \ldots, l_K)$ giving the probability that the system works given exactly l_k of the components of type k are working. See Coolen and Coolen-Maturi (2012) for details. Thus, the survival signature can be represented by a table with $K + 1$ columns, the first K being the number of each type of component which is working and the final column being the probability the system works.

The system or network is specified by means of an igraph object, whereby each end of the system is denoted by nodes names "s" and "t" which are taken to be perfectly reliable. It is easy to construct the appropriate graph representation using the function graph.formula. Note that each physically distinct component should be separately numbered when constructing this graph object.

Once the topology of the system has been defined, one must attach a vertex attribute named compType to each component to indicate the type of the component. The Examples section below features the full computation of the survival signature for Figure 1 in Coolen and Coolen-Maturi (2012) and Figure 2 in Coolen *et al* (2013) to make this clear.

Value

computeSystemSurvivalSignature returns a data frame with $K+1$ columns. The first K columns represent the function inputs, $l_1, \ldots, l_K$ and the final column is the probability that the system works given the corresponding numbers of each component which are working.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J. M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Coolen, F. P. A. and Coolen-Maturi, T. (2012), Generalizing the signature to systems with multiple types of components, *in* 'Complex Systems and Dependability', Springer, pp. 115-130.

Coolen, F. P. A., Coolen-Maturi, T., Al-nefaiee, A. H. and Aboalkhair, A. M. (2013), 'Recent advances in system reliability using the survival signature', *Proceedings of Advances in Risk and Reliability Technology Symposium, Loughborough*.

See Also

computeSystemSignature

Examples

```
## EXAMPLE 1
## Figure 1 in Coolen and Coolen-Maturi (2012)

# First, define the structure, ensuring that each physically separate component
# is separately numbered
fig1 <- graph.formula(s -- 1 -- 2:3 -- 4 -- 5:6 -- t, 2 -- 5, 3 -- 6)

# Second, create a vertex attribute specifying the type of each of those
# numbered component (leaving s,t with no type)
V(fig1)$compType <- NA # This just creates the attribute compType
V(fig1)$compType[match(c("1","2","5"), V(fig1)$name)] <- "Type 1"
V(fig1)$compType[match(c("3","4","6"), V(fig1)$name)] <- "Type 2"
V(fig1)$compType[match(c("s","t"), V(fig1)$name)] <- NA

# Third, compute the survival signature (getting fractions rather than decimals)
computeSystemSurvivalSignature(fig1, frac=TRUE)



## EXAMPLE 2
## Figure 3 in Coolen et al (2013)

# First, define the structure, ensuring that each physically separate component
# is separately numbered
fig3 <- graph.formula(s -- 1:4 -- 2:5 -- 3:6 -- t, s -- 7:8, 8 -- 9, 7:9 -- t)

# Second, create a vertex attribute specifying the type of each of those
# numbered component (leaving s,t with no type)
V(fig3)$compType <- NA # This just creates the attribute compType
V(fig3)$compType[match(c("1"), V(fig3)$name)] <- "Type 1"
V(fig3)$compType[match(c("2","3","4","7"), V(fig3)$name)] <- "Type 2"
V(fig3)$compType[match(c("5","6","8","9"), V(fig3)$name)] <- "Type 3"
V(fig3)$compType[match(c("s","t"), V(fig3)$name)] <- NA

# Third, compute the survival signature (getting fractions rather than decimals)
computeSystemSurvivalSignature(fig3, frac=TRUE)
```

`expectedSystemLifetimeExp` *Compute the expected lifetime of a given system*

Description

Computes the expected lifetime of a system/network specified by its signature or graph structure when the components have Exponential lifetime distribution with specified rate. Useful for ordering systems/networks by expected lifetime.

Usage

```
expectedSystemLifetimeExp(g, rate=1)
expectedNetworkLifetimeExp(g, rate=1)
expectedSignatureLifetimeExp(s, rate=1)
```

Arguments

`g`	an `igraph` object representing the system or network whose expected lifetime is to be computed. There should be two terminal 'dummy' nodes to represent either end of the structure which must be labelled "s" and "t". They are assumed perfectly reliable. See details and examples.
`s`	the signature vector of the system/network whose expected lifetime is to be computed.
`rate`	the rate parameter of the Exponential distribution.

Details

The system or network can be specified by means of an `igraph` object, whereby each end of the system is denoted by nodes names "s" and "t" which are taken to be perfectly reliable. It is easy to construct the appropriate graph representation using the function `graph.formula`.

Alternatively, the signature may be provided instead (the other functions simply use the graph object to compute the signature).

Value

All the functions return a single scalar value which is the expected lifetime.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Samaniego, F. J. (2007), *System Signatures and Their Applications in Engineering Reliability*, Springer.

See Also

computeSystemSignature

Examples

```
# Find the expected lifetime of two component series system
expectedSystemLifetimeExp(graph.formula(s -- 1 -- 2 -- t))

# Find the expected lifetime of two component series system using it's signature directly
expectedSignatureLifetimeExp(c(1,0))

# Find the expected lifetime of two component parallel system
expectedSystemLifetimeExp(graph.formula(s -- 1:2 -- t))

# Find the expected lifetime of two component parallel system using it's signature directly
expectedSignatureLifetimeExp(c(0,1))
```

---

maskedInferenceEXCHCustom

Inference for Masked Exchangeable System Lifetimes, Custom Distribution

---

Description

Performs Bayesian inference via a signature based data augmentation MCMC scheme for masked system lifetime data for any custom component lifetime distribution. The underlying assumption is of exchangeability at the system level (iid components within each exchangeable system).

Usage

```
maskedInferenceEXCHCustom(t, signature, cdfComp, pdfComp, rParmGivenData,
                          rCompGivenParm, startCompParm, startHypParm, iter, ...)
```

Arguments

t	a vector of masked system lifetimes.
signature	the signature vector of the system/network for which inference is performed. It may be a list of signatures which results in topological inference on the system design being jointly performed over the collection of signatures provided.
cdfComp	user-defined vectorised cumulative distribution function of component lifetime $F_Y()$ with prototype: `function(y, parametersm, ...)`

`pdfComp` user-defined vectorised probability distribution function of component lifetime $f_Y()$ with prototype: `function(y, parameters, ...)`

`rParmGivenData` user-defined function which should produce random draws from $f_{\Xi \mid Y}$ with prototype:

`function(y, ...)`

This must return the new parameters as a 2 item list: the first item being the hyperprior parameters drawn in the same named vector format and order as `startHypPriorParm`; the second item being a list of component lifetime parameters drawn in the same named vector format and order as `startCompParm`.

`rCompGivenParm` user-defined function which should produce random draws from $f_{Y \mid \Psi}$ with prototype:

`function(parameters, t, censoring, ...)`

where censoring is -1 for left censoring, 0 for exact observations and 1 for right censoring.

`startCompParm` list consisting of a vectors of starting values per system (in the same order as `t`) of named parameters for the component lifetime distribution. The order within each named vector should match the order expected for the `parameters` argument in the user defined functions above.

`startHypParm` vector of starting values of named hyper-parameters for the hyperprior.

`iter` number of MCMC iterations to perform.

`...` additional arguments which are passed through to the user-defined functions above.

Details

This is a low level implementation of the signature based data augmented MCMC scheme described in Aslett (2012) for exchangeable systems. This function need only be used if the component lifetime distribution of interest has not already been implemented within this package.

The arguments of the function are the prerequisites described in Algorithm 6.2 of Aslett (2012). The interested user is advised to inspect the source code of this package at the file `MaskedLifetimeInference_Exponential.R` for an example of its usage, which may be seen in the function `maskedInferenceEXCHExponential` defined there, together with the associated user-definied functions above it.

Value

If a single signature vector is provided above, then a data frame of MCMC samples with columns named the same as the `startParm` argument is returned.

If a list of signature vectors is provided above, then a list is returned containing three items:

`topology` A vector of posterior samples from the discrete marginal posterior distribution of topologies provided in the signature list.

`parameters` A list of data frames of MCMC samples with columns named the same as the `startCompParm` argument.

`hyperparameters`

A data frame of MCMC samples with columns named the same as the `startHypPriorParm` argument.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

See Also

computeSystemSignature

Examples

```
# Please inspect the source of this package, file MaskedLifetimeInference_Exponential.R
# for example usage (see details section)
```

---

maskedInferenceEXCHExponential

Inference for Masked Exchangeable System Lifetimes, Exponential Components

---

Description

Performs Bayesian inference via a signature based data augmentation MCMC scheme for masked system lifetime data for Exponentially distributed component lifetimes. The underlying assumption is exchangeability at the system level.

Usage

```
maskedInferenceEXCHExponential(t, signature, iter, priorMu_Mu, priorSigma_Mu,
                               priorMu_Sigma, priorSigma_Sigma)
```

Arguments

t	a vector of masked system lifetimes.
signature	the signature vector of the system/network for which inference is performed. It may be a list of signatures which results in topological inference on the system design being jointly performed over the collection of signatures provided.
iter	number of MCMC iterations to perform.

`priorMu_Mu, priorSigma_Mu`
: μ and σ parameters for Log-Normal hyperprior on the mean of the exchangeable Gamma population distribution for the rate.

`priorMu_Sigma, priorSigma_Sigma`
: μ and σ parameters for Log-Normal hyperprior on the variance of the exchangeable Gamma population distribution for the rate.

Details

This is a full implementation of the signature based data augmented MCMC scheme described in Aslett (2012) for exchangeable systems with Exponential component lifetimes.

Thus, components are taken to have Exponential lifetimes where the rate of the components in any given system is a realisation from an exchangeable population distribution. However, only the failure time of the system is observed, not those of the components or indeed which components were failed at the system failure time. By specifying a Log-Normal hyper-prior on the exchangeable Gamma rate parameter population distribution, this function then produces MCMC samples from the posterior of the rate parameter and hyperparameters.

The model is as follows:

$$Y \mid \lambda \sim \text{Exponential}(\lambda)$$

$$\lambda \mid \nu, \zeta \sim \text{Gamma}(\text{shape} = \nu, \text{scale} = \zeta)$$

$$\mu \sim \text{Log-Normal}(\mu_\nu, \sigma_\nu)$$

$$\sigma^2 \sim \text{Log-Normal}(\mu_\zeta, \sigma_\zeta)$$

Additionally, if one does not know the system design, then it is possible to pass a list of many system signatures in the `signature` argument, in which case the topology of the system is jointly inferred with the parameters.

Value

If a single signature vector is provided above, then a list is returned containing two items:

`parameters`
: A list of data frames, each with a single column of MCMC samples from the posterior samples from the exhcangeable rate parameter for the given system.

`hyperparameters`
: A data frame with a column of posterior MCMC samples for each of the Log-Normal hyperprior parameters.

If a list of signature vectors is provided above, then a list is returned containing three items – the two items above, plus:

`topology`
: A vector of posterior samples from the discrete marginal posterior distribution of topologies provided in the signature list.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

See Also

computeSystemSignature

Examples

```
# Some masked system lifetime data for an exchangeable collection of systems with
# Exponential component lifetime, rate drawn from the population distribution
# Gamma(shape=9, scale=0.5)
t <- c(0.2265, 0.0795, 0.1178, 0.2463, 0.1053, 0.0982, 0.0349, 0.0363,
0.1546, 0.1357, 0.1239, 0.0354, 0.0124, 0.1003, 0.0827, 0.2446,
0.1214, 0.1272, 0.5438, 0.2738, 0.0378, 0.2293, 0.1706, 0.0146,
0.1506, 0.3665, 0.046, 0.1196, 0.2724, 0.2593, 0.0438, 0.1493,
0.0697, 0.1774, 0.1157, 0.0996, 0.2815, 0.1411, 0.0921, 0.2088,
0.1164, 0.149, 0.048, 0.1019, 0.2349, 0.2211, 0.0475, 0.0721,
0.0371, 0.611, 0.1959, 0.0666, 0.0956, 0.1416, 0.2126, 0.0104,
0.088, 0.0159, 0.078, 0.1747, 0.1921, 0.3558, 0.4956, 0.0436,
0.2292, 0.1159, 0.1201, 0.1299, 0.043, 0.0584, 0.0347, 0.2084,
0.1334, 0.1491, 0.0384, 0.0589, 0.2962, 0.1023, 0.0506, 0.0501,
0.1859, 0.0714, 0.1424, 0.0027, 0.2812, 0.0318, 0.4147, 0.1088,
0.2894, 0.0734, 0.1405, 0.0367, 0.0323, 0.517, 0.1034, 0.026,
0.0485, 0.0512, 0.0116, 0.1629)

# Load the signatures of order 4 simply connected coherent systems -- the data
# above correspond to simulations from system number 3
data(sccs04)

# Perform inference on the rate parameter:
## Not run: samps <- maskedInferenceEXCHExponential(t, sccs04[[3]]$signature,
2000, priorMu_Mu=1, priorSigma_Mu=0.5, priorMu_Sigma=1, priorSigma_Sigma=0.7)
## End(Not run)

# Or perform inference on rate parameter and topology jointly, taking as candidate
# set all possible simply connected coherent systems of order 4:
## Not run: samps <- maskedInferenceEXCHExponential(t, sccs04, 2000, priorMu_Mu=1,
priorSigma_Mu=0.5, priorMu_Sigma=1, priorSigma_Sigma=0.7)
## End(Not run)
```

`maskedInferenceIIDCustom` *Inference for Masked iid System Lifetimes, Custom Distribution*

Description

Performs Bayesian inference via a signature based data augmentation MCMC scheme for masked system lifetime data for any custom component lifetime distribution. The underlying assumption is iid components and iid systems.

Usage

```
maskedInferenceIIDCustom(t, signature, cdfComp, pdfComp, rParmGivenData,
                         rCompGivenParm, startParm, iter, ...)
```

Arguments

`t`	a vector of masked system lifetimes.
`signature`	the signature vector of the system/network for which inference is performed. It may be a list of signatures which results in topological inference on the system design being jointly performed over the collection of signatures provided.
`cdfComp`	user-defined vectorised cumulative distribution function of component lifetime $F_Y()$ with prototype: `function(y, parametersm, ...)`
`pdfComp`	user-defined vectorised probability distribution function of component lifetime $f_Y()$ with prototype: `function(y, parameters, ...)`
`rParmGivenData`	user-defined function which should produce random draws from $f_{\Psi \mid Y}$ with prototype: `function(y, ...)` This must return the parameters in the same order named vector format as used for `startParm`.
`rCompGivenParm`	user-defined function which should produce random draws from $f_{Y \mid \Psi}$ with prototype: `function(parameters, t, censoring, ...)` where censoring is -1 for left censoring, 0 for exact observations and 1 for right censoring.
`startParm`	vector of starting values of named parameters in the correct order for the `parameters` argument in the user defined functions above.
`iter`	number of MCMC iterations to perform.
`...`	additional arguments which are passed through to the user-defined functions above.

Details

This is a low level implementation of the signature based data augmented MCMC scheme described in Aslett (2012) for iid systems. This function need only be used if the component lifetime distribution of interest has not already been implemented within this package.

The arguments of the function are the prerequisites described in Algorithm 6.2 of Aslett (2012). The interested user is advised to inspect the source code of this package at the file `MaskedLifetimeInference_Exponential.R` for an example of its usage, which may be seen in the function `maskedInferenceIIDExponential` defined there, together with the associated user-definied functions above it.

Value

If a single signature vector is provided above, then a data frame of MCMC samples with columns named the same as the `startParm` argument is returned.

If a list of signature vectors is provided above, then a list is returned containing two items:

`topology`	A vector of posterior samples from the discrete marginal posterior distribution of topologies provided in the signature list.
`parameters`	A data frame of MCMC samples with columns named the same as the `startParm` argument.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

See Also

`computeSystemSignature`

Examples

```
# Please inspect the source of this package, file MaskedLifetimeInference_Exponential.R
# for example usage (see details section)
```

`maskedInferenceIIDExponential`

Inference for Masked iid System Lifetimes, Exponential Components

---

Description

Performs Bayesian inference via a signature based data augmentation MCMC scheme for masked system lifetime data for Exponentially distributed component lifetimes. The underlying assumption is iid components and iid systems.

Usage

```
maskedInferenceIIDExponential(t, signature, iter, priorShape, priorScale)
```

Arguments

`t`	a vector of masked system lifetimes.
`signature`	the signature vector of the system/network for which inference is performed. It may be a list of signatures which results in topological inference on the system design being jointly performed over the collection of signatures provided.
`iter`	number of MCMC iterations to perform.
`priorShape`	the shape parameter of the Gamma prior of the Exponential rate.
`priorScale`	the scale parameter of the Gamma prior of the Exponential rate.

Details

This is a full implementation of the signature based data augmented MCMC scheme described in Aslett (2012) for iid systems with Exponential component lifetimes.

Thus, components are taken to have Exponential lifetimes and be arranged into some system. However, only the failure time of the system is observed, not those of the components or indeed which components were failed at the system failure time. By specifying a Gamma prior distribution on the component lifetime Exponential rate parameter via `priorShape` and `priorScale`, this function then produces MCMC samples from the posterior of the rate parameter.

Additionally, if one does not know the system design, then it is possible to pass a list of many system signatures in the `signature` argument, in which case the topology of the system is jointly inferred with the parameters.

Value

If a single signature vector is provided above, then a data frame with a single column of MCMC samples from the posterior of the rate parameter are returned.

If a list of signature vectors is provided above, then a list is returned containing two items:

`topology`	A vector of posterior samples from the discrete marginal posterior distribution of topologies provided in the signature list.
`parameters`	A data frame with a single column of MCMC samples from the posterior of the rate parameter.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

See Also

computeSystemSignature

Examples

```
# Some masked system lifetime data for a system with Exponential component
# lifetime, rate=3.14
t <- c(0.2696, 0.3613, 0.0256, 0.1287, 0.2305, 0.1565, 0.2484, 0.7482,
0.1748, 0.1805, 0.1985, 0.0799, 0.2843, 0.2392, 0.2151, 0.1177,
0.1278, 0.4189, 0.4374, 0.0931, 0.2846, 0.0357, 0.1809, 0.2077,
0.5211, 0.4935, 0.1464, 0.0297, 0.5429, 0.1294, 0.7089, 0.5534,
0.1183, 0.2628, 0.0481, 0.0518, 0.0533, 0.3595, 0.0767, 0.2606,
0.1005, 0.227, 0.01, 0.0947, 0.1248, 0.2288, 0.1422, 0.233, 0.1428,
0.2043)

# Load the signatures of order 4 simply connected coherent systems -- the data
# above correspond to simulations from system number 3
data(sccs04)

# Perform inference on the rate parameter:
## Not run: samps <- maskedInferenceIIDExponential(t, sccs04[[3]]$signature, 2000,
priorShape=9, priorScale=0.5)
## End(Not run)

# Or perform inference on rate parameter and topology jointly, taking as candidate
# set all possible simply connected coherent systems of order 4:
## Not run: samps <- maskedInferenceIIDExponential(t, sccs04, 2000, priorShape=9,
priorScale=0.5)
## End(Not run)
```

---

sccs02 *Catalogue of Simply Connected Coherent Systems of Order 2*

Description

This data sets provides a catalogue of the network graph, signature and minimal cut-sets of all simply connected coherent systems of order 2.

Usage

```
data(sccs02)
```

Format

A list object, one item for each such systems. Each item is itself a list, with the elements `$graph`, `$cutsets` and `$signature`.

Source

Derived in the thesis Aslett (2012).

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

`sccs03` *Catalogue of Simply Connected Coherent Systems of Order 3*

Description

This data sets provides a catalogue of the network graph, signature and minimal cut-sets of all simply connected coherent systems of order 3.

Usage

```
data(sccs03)
```

Format

A list object, one item for each such systems. Each item is itself a list, with the elements `$graph`, `$cutsets` and `$signature`.

Source

Derived in the thesis Aslett (2012).

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

sccs04 *Catalogue of Simply Connected Coherent Systems of Order 4*

Description

This data sets provides a catalogue of the network graph, signature and minimal cut-sets of all simply connected coherent systems of order 4.

Usage

```
data(sccs04)
```

Format

A list object, one item for each such systems. Each item is itself a list, with the elements `$graph`, `$cutsets` and `$signature`.

Source

Derived in the thesis Aslett (2012).

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

sccs05 *Catalogue of Simply Connected Coherent Systems of Order 5*

Description

This data sets provides a catalogue of the network graph, signature and minimal cut-sets of all simply connected coherent systems of order 5.

Usage

```
data(sccs05)
```

Format

A list object, one item for each such systems. Each item is itself a list, with the elements `$graph`, `$cutsets` and `$signature`.

Source

Derived in the thesis Aslett (2012).

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

`simulateSystem` *Simulate Masked Lifetime Data for a System*

Description

This function enables easy simulation of iid masked lifetime observations from a system or network.

Usage

```
simulateSystem(system, n, rdens, ...)
```

Arguments

`system`	may be: an `igraph` object representing the system; the collection of cutsets of the system; or the system signature.
`n`	how many simulations to produce.
`rdens`	a user defined function which generates random realisations of the component lifetimes.
`...`	parameters passed to the user defined function `rdens`.

Details

When the system or network is specified by means of an `igraph` object, each end of the system must be denoted by nodes named "s" and "t" which are taken to be perfectly reliable. It is easy to construct the appropriate graph representation using the function `graph.formula`.

This function then generates iid realisations of masked lifetimes.

Value

a numeric vector of length `n` containing the masked lifetime data.

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

Examples

```
# Simulate 20 masked lifetimes of a two component series system with Exponential(2)
# component lifetimes
# Using igraph object ...
simulateSystem(graph.formula(s -- 1 -- 2 -- t), 20, rexp, rate=2)

# ... and using signature
simulateSystem(c(1,0), 20, rexp, rate=2)
```

`systemGraphToGenerator` *Construct a Continuous-time Markov Chain Generator*

Description

This function enables easy construction of an absorbing continuous-time Markov chain generator matrix representation for a system when components are treated as having Exponential failure and repair times.

Usage

```
systemGraphToGenerator(g, failRate, repairRate)
```

Arguments

g	an igraph object representing the system or network whose generator matrix representation is to be computed. There should be two terminal 'dummy' nodes to represent either end of the structure which must be labelled "s" and "t". They are assumed perfectly reliable. See details and examples.
failRate	the rate parameter of the Exponentially distributed lifetime distribution of the components.
repairRate	the rate parameter of the Exponentially distributed repair time distribution of the components.

Details

When the system or network is specified by means of an igraph object, each end of the system must be denoted by nodes named "s" and "t" which are taken to be perfectly reliable. It is easy to construct the appropriate graph representation using the function graph.formula.

This function then creates the generator matrix for an absorbing continuous-time Markov chain representation of such a system where components are repairable. All system states which in which the system is inoperative are collapsed into the absorbing state.

The returned values are in the format required by the phtMCMC2.

Full details are in Aslett (2012).

Value

A list is returned with both a numeric generator matrix (in `$G` with the failure rate, `failRate`, and repair rate, `repairRate`) and a symbolic matrix (in `$structure$G`), along with a matrix of the constant multiples of generator entries (in `$structure$C`).

Note

Please feel free to email <aslett@stats.ox.ac.uk> with any queries or if you encounter errors when running this function.

Author(s)

Louis J.M. Aslett <aslett@stats.ox.ac.uk> (http://www.louisaslett.com/)

References

Aslett, L. J. M. (2012), *MCMC for Inference on Phase-type and Masked System Lifetime Models*, PhD Thesis, Trinity College Dublin.

Examples

```
# Get the generator representing a repairable 5 component 'bridge' system with
# failure rate 1 and repair rate 365.
data(sccs05)
G <- systemGraphToGenerator(sccs05[[18]]$graph, 1, 365)
```

Index