Package ‘epinet’

May 8, 2026

Version 2.1.11

Date 2023-11-29

Title Epidemic/Network-Related Tools

Imports network

LazyLoad yes

LazyData yes

Description A collection of epidemic/network-related tools. Simulates transmission of diseases through contact networks. Performs Bayesian inference on network and epidemic parameters, given epidemic data.

License GPL-2

NeedsCompilation yes

Author Chris Groendyke [aut, cre],
David Welch [aut],
David Hunter [ctb]

Maintainer Chris Groendyke <cgroendyke@gmail.com>

Repository CRAN

Date/Publication 2023-11-29 19:40:02 UTC

Contents

BuildX 2
epi2newick 3
epinet 4
ess 8
Hagelloch 9
MCMCcontrol 10
plot.epidemic 12
plot.epinet 14
print.epidemic 16
print.epinet 17
priorcontrol 19
SEIR.simulator 21

SimulateDyadicLinearERGM 23
summary.epidemic 24
summary.epinet 25
write.epinet 27

Index **29**

`BuildX` *Build a dyadic covariate matrix (X)*

Description

Build a dyadic covariate matrix (X) from a given nodal covariate matrix.

Usage

```
BuildX(nodecov, unaryCol = NULL, unaryFunc = NULL,
binaryCol = NULL, binaryFunc = NULL, includeIntercept = TRUE)
```

Arguments

`nodecov`	an N x k matrix where N is the number of nodes, column 1 is the node id and columns 2:k are covariate values for the node
`unaryCol`	a vector of column indices
`unaryFunc`	a vector of the same length as unaryCol of method names for comparing dyads. Possible method names are "match" and "absdiff"
`binaryCol`	a list of 2 element vectors of column indices
`binaryFunc`	a vector of the same length as binaryCol of method names for comparing dyads. Possible method names are "euclidean" and "manhattan"
`includeIntercept`	logical. If TRUE, includes a column of all ones. Defaults to TRUE

Value

A dyadic covariate matrix with $\binom{N}{2}$ rows, columns 1 and 2 are node ids, column 3 is all ones (if requested) and then one column for each given element of unaryCol and binaryCol.

Assigns colnames depending on type of unaryFunc and binaryFunc and colnames of nodecov.

Author(s)

David Welch <david.welch@auckland.ac.nz>, Chris Groendyke <cgroendyke@gmail.com>

See Also

`SimulateDyadicLinearERGM` for simulating a contact network based on a dyadic covairate matrix, and `epinet` for performing inference on the network and epidemic model parameters.

Examples

```
# make some nodal covariates
set.seed(3)
mycov = data.frame(id = 1:5, xpos = rnorm(5), ypos = rnorm(5),
house = c(1, 1, 2, 2, 2), gender = c(0, 0, 0, 1, 1))
# make matrix
dyadCov = BuildX(mycov, unaryCol = c(4, 5), unaryFunc = c("match", "match"),
binaryCol = list(c(2, 3)), binaryFunc = "euclidean")
```

`epi2newick` *Prints a transmission tree in Newick format.*

Description

Prints a simulated or inferred transmission tree in Newick format.

Usage

```
epi2newick(epi)

epi2newickmcmc(mcmcoutput, index = dim(mcmcoutput$transtree)[2])
```

Arguments

`epi`	a simulated epidemic, in the form of the output produced by `SEIR.simulator`.
`mcmcoutput`	output from `epinet`.
`index`	a number indicating which of the MCMC samples to plot. Defaults to the final sample in the chain.

Details

Converts the epinet epidemic format into a transmssion tree represented as a Newick string which is the standard tree format used in phylogenetics. There are many packages available to analyse Newick format trees such as the ape package, IcyTree and FigTree.

Value

A character string representing the epidemic transmission tree in Newick format. Note that this string contains control characters that can be removed by using `cat`

Author(s)

David Welch <david.welch@auckland.ac.nz>, Chris Groendyke <cgroendyke@gmail.com>

References

Rambaut A. 2014. FigTree v1.4. http://tree.bio.ed.ac.uk/software/figtree/. Vaughan T. 2015. IcyTree https://icytree.org.

See Also

epinet for generating posterior samples of the parameters, print.epinet and summary.epinet for printing basic summary information about an epinet object, write.epinet for writing parameter and transmission tree posterior samples to file, and plot.epinet for plotting the posterior samples of the transmission tree.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist="gamma")
cat(epi2newick(exampleepidemic))

## Not run:
# Build covariates
set.seed(1)
N <- 50
mycov <- data.frame(id = 1:N, xpos = runif(N), ypos = runif(N))
dyadCov <- BuildX(mycov,binaryCol = list(c(2, 3)),binaryFunc = c("euclidean"))
# Build network
eta <- c(0, -7)
net <- SimulateDyadicLinearERGM(N = N,dyadiccovmat = dyadCov,eta = eta)
# Simulate epidemic
epi <- SEIR.simulator(M=net,N=N,beta=1,ki=3,thetai=7,ke=3,latencydist="gamma")
# Run MCMC routine on simulated epidemic
mcmcinput <- MCMCcontrol(nsamp = 1000000, thinning = 100, etapropsd = c(1, 1))
priors <- priorcontrol(bprior = c(0, 4), tiprior = c(1, 15), teprior = c(1, 15),
etaprior = c(0, 10, 0, 10), kiprior = c(1, 7), keprior = c(1, 7), priordists = "uniform")
out <- epinet(~ xpos.ypos.L2Dist, epidata = epi, dyadiccovmat = dyadCov,
mcmcinput = mcmcinput, priors = priors)
cat(epi2newickmcmc(out))
## End(Not run)
```

---

epinet — *Uses epidemic data to perform Bayesian inference on a contact network*

Description

Performs Bayesian inference on parameters for an SEIR epidemic model and a random graph model, given recovery (and perhaps also exposure/infective) times for each individual infected during the course of an epidemic.

Usage

```
epinet(formula, epidata, dyadiccovmat, mcmcinput = MCMCcontrol(),
priors = priorcontrol(), verbose = TRUE)
```

Arguments

formula	an object of class formula giving a symbolic description of the model to be fit.
epidata	input data consisting of exposure, infection, and recovery times.
dyadiccovmat	matrix of dyadic covariates (X). Can be constructed using BuildX.
mcmcinput	list of control options for MCMC algorithm. Can be constructed using MCMCcontrol.
priors	list of prior distributions and parameters. Can be constructed using priorcontrol.
verbose	boolean variable specifying whether progress and information messages are displayed during the course of the MCMC routine. Defaults to TRUE.

Details

Uses exposed, infective, and removal times from the infected nodes of an epidemic in order to perform inference on the parameters of the network and epidemic models.

The formula will consist of variables (column names) found in the dyadiccovmat parameter. By default, the model will include an intercept term.

epidata is an N row by 5 column array giving the identity, likely parent, and exposed, infective, and removal times for each of the N individuals in the population, as well as the values of any nodal covariates. Column 1 gives the ID (an integer) of the node, and column 2 gives the identity of the probable parent of the node (if known). Columns 3, 4, and 5 give the exposed, infective, and removal times. Individuals who were not infected during the course of the epidemic should have NA coded in columns 3, 4, and 5; the records for these individuals should appear AFTER those corresponding to the individuals that were infected during the epidemic. Note that if the times are not internally consistent, an error message will be generated and no inference will be performed. It is necessary to include data for exposure and infective times, even if these values are not known (in this case, set the respective entries to NA).

Any data rows corresponding to individuals not infected during the course of the epidemic, if present, must occur at the end of the array, after all rows for infected individuals. These rows must have removal times of NA.

dyadiccovmat is an $\binom{N}{2}$ row by $(k+2)$ column matrix containing the dyadic covariates for the population, where N is the number of individuals in the population and k is the number of dyadic covariates used in the model. The matrix contains one row for each dyad (pair of nodes). Columns 1 and 2 give the ID of the two nodes comprising the dyad, and the remaining k columns give the covariate values.

Uses an algorithm similar to that described in Groendyke and Welch (2018), Groendyke et al. (2010), and Britton and O'Neill (2002).

Value

accept	vector containing the number of times a proposed new value was accepted for the parameters (P, eta, G, beta, thetai, ki, thetae, ke).

`propose`	vector containing the number of times a new value was proposed for the parameters (P, eta, G, beta, thetai, ki, thetae, ke).
`llkd`	vector containing the log-likelihood at each iteration of the MCMC algorithm.
`beta`	vector containing the sample for parameter beta.
`thetai`	vector containing the sample for parameter thetai.
`thetae`	vector containing the sample for parameter thetae.
`ki`	vector containing the sample for parameter ki.
`ke`	vector containing the sample for parameter ke.
`eta`	2-dimensional array containing the samples for the eta parameters. The ith column contains the sample for the ith eta parameter.
`initexp`	vector containing the sample for parameter kappa (identity of initial exposed). Will only vary when both the exposure and infection times are assumed unknown.
`initexptime`	vector containing the sample for parameter E_κ (initial exposure time). Will only vary when the exposure times are assumed unknown.
`exptimes`	if exposure times are inferred and corresponding posterior samples are returned, this is two-dimensional array containing the inferred exposure times (exptimes[i,] contains the sample of exposure times for node i). Otherwise, this will be NULL.
`inftimes`	if infection times are inferred and corresponding posterior samples are returned, this is two-dimensional array containing the inferred infection times (inftimes[i,] contains the sample of infection times for node i). Otherwise, this will be NULL.
`rectimes`	vector containing the original recovery times.
`nodeid`	vector containing the node IDs for the individuals in the population.
`transtree`	A two-dimensional array containing the sample for inferred transmission tree. transtree[i,] contains the sample of parent nodes for node i. A parent node of -999 for i designates that i is the initial exposed node. If the transmission tree is not inferred and returned, this will be NULL.
`call`	the matched call.
`formula`	the formula used in the inference routine.
`mcmcinfo`	input settings for the MCMC chain

Author(s)

Chris Groendyke <cgroendyke@gmail.com>, David Welch <david.welch@auckland.ac.nz>

References

Groendyke, C. and Welch, D. 2018. epinet: An R Package to Analyze Epidemics Spread across Contact Networks, *Journal of Statistical Software*, **83-11**.

Groendyke, C., Welch, D. and Hunter, D. 2012. A Network-based Analysis of the 1861 Hagelloch Measles Data, *Biometrics*, **68-3**.

Groendyke, C., Welch, D. and Hunter, D. 2010. Bayesian inference for contact networks given epidemic data, *Scandinavian Journal of Statistics*, **38-3**.

Britton, T. and O'Neill, P.D. 2002. Bayesian inference for stochastic epidemics in populations with random social structure, *Scandinavian Journal of Statistics*, **29-3**.

See Also

BuildX for building a dyadic covariate matrix, MCMCcontrol for specifying control parameters for the MCMC algorithm, priorcontrol for specifying prior distributions and their hyperparameters, epi2newick and write.epinet for writing the output of the algorithm to file, and plot.epinet for plotting the posterior samples of the transmission tree.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N,nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat=dcm, eta=-1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist="gamma")
# Set inputs for MCMC algorithm
mcmcinput <- MCMCcontrol(nsamp = 5000, thinning = 10, etapropsd = 0.2)
priorcontrol <- priorcontrol(bprior = c(0, 1), tiprior = c(1, 3), teprior = c(1, 3),
    etaprior = c(0, 10), kiprior = c(2, 8), keprior = c(2, 8), priordists = "uniform")
# Run MCMC algorithm on this epidemic
# Note: Not enough data or iterations for any real inference
examplemcmc <- epinet( ~ 1, exampleepidemic, dcm, mcmcinput, priorcontrol)

## Not run:
# Note: This may take a few minutes to run.
set.seed(1)
N <- 50
mycov <- data.frame(id = 1:N, xpos = runif(N), ypos = runif(N))
dyadCov <- BuildX(mycov, binaryCol = list(c(2, 3)),binaryFunc = c("euclidean"))
# Build network
eta <- c(0,-7)
net <- SimulateDyadicLinearERGM(N = N, dyadiccovmat = dyadCov, eta = eta)
# Simulate epidemic
epi <- SEIR.simulator(M = net, N = N, beta = 1, ki = 3, thetai = 7, ke = 3, latencydist = "gamma")
# Run MCMC routine on simulated epidemic
mcmcinput <- MCMCcontrol(nsamp = 1000000, thinning = 100, etapropsd = c(1, 1))
priors <- priorcontrol(bprior = c(0, 4), tiprior = c(1, 15), teprior = c(1, 15),
etaprior = c(0, 10, 0, 10), kiprior = c(1, 7), keprior = c(1, 7), priordists = "uniform")
out <- epinet(~ xpos.ypos.L2Dist, epidata = epi, dyadiccovmat = dyadCov,
mcmcinput = mcmcinput, priors = priors)

## End(Not run)
```

ess *Calculate the Effective Sample Size*

Description

Calculate the Effective Sample Size for a marginal posterior sample obtained via MCMC

Usage

```
ess(x, ignoreBurnin = FALSE, burninProportion = 0.1)
```

Arguments

x	a numeric vector of length N assumed to be samples from a Markov chain
ignoreBurnin	logical indictating whether or not the first burninProportion of vector x should be ignored
burninProportion	if ignoreBurnin == TRUE, the first burninProportion*length(x) samples are removed from x before the ess is calculated

Details

Calculates the effective sample size of x based on an estimate of the lag autocorrelation function. Details of the method are in Section 11.5 of Bayesian Data Analysis, Third Edition, 2013, Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, Donald B. Rubin.

Value

Returns the estimated effective sample size for the last (1-burninProportion) samples in x.

Author(s)

David Welch <david.welch@auckland.ac.nz>, Chris Groendyke <cgroendyke@gmail.com>

References

Gelman, A., Carlin, J.B., Stern, H.S., Dunson, D.B., Vehtari, A., Rubin, D.B., 2013 *Bayesian Data Analysis*, Third Edition, (Section 11.5), Boca Raton, Florida: CRC Press.

Examples

```
set.seed(8)
x <- runif(1000)
# expect ESS of close to 900 as samples are iid
ess(x, ignoreBurnin = TRUE)
# no burnin to ignore so ess is actually close to 1000
ess(x, ignoreBurnin = FALSE)
```

```
# ESS is a rough measure at best
ess(1:1000,ignoreBurnin = FALSE)
```

`Hagelloch` *Hagelloch measles data.*

Description

Epidemic data derived from a measles outbreak in the town of Hagelloch, Germany in 1861. 188 individuals were infected over the course of the epidemic. (One individual was removed from this dataset.)

Consists of two files: HagellochTimes and HagellochDyadCov. These two files contain the data necessary to analyze the Hagelloch measles data.

HagellochTimes contains 187 rows (one for each individual included) and 5 columns: NodeID (a numerical index ranging from 1 to 187); Putative Parent (the ID of the individual considered most likely to have been responsible for infecting this person, as determined by Osterle); Exposure time, Infectious time, and Removal time (the time index in days at which the individual entered the Exposed, Infectious, and Removed states, respectively). Note that Exposure times are not known for this data set. See references below for details regarding the determination of Infectious and Removal times.

HagellochDyadCov is a matrix of dyadic covariates corresponding to the individuals in the Hagelloch data set. Contains one row for each dyad (pair of individuals) in the population. The first two columns are the Node IDs for the two individuals in the dyad. The third column is a column of all 1 values, used as a baseline or intercept term. Columns 4, 5, and 6 are indicator variables for whether the two individuals in the dyad are in the same household, are both in classroom 1, or both in classroom 2, respectively. Column 7 is the household distance between the two individuals in the dyad, measured in units of 2.5m, Columns 8 and 9 are indicator variables based on whether both individuals in the dyad are male or female, respectively. Column 10 is the age difference (in years) between the two individuals in the dyad.

Usage

```
data(Hagelloch)
```

Format

See above.

Source

Thanks to Peter Neal for providing this data set. This data was originally collected by Pfeilsticker:

Pfeilsticker, A. (1863). Beitrage zur Pathologie der Masern mit besonderer Berucksichtigung der statistischen Verhaltnisse, M.D. thesis, Eberhard-Karls Universitat, Tubingen.

and later modified by Osterle:

Oesterle, H. (1992). Statistiche Reanalyse einer Masernepidemie 1861 in Hagelloch, M.D. Thesis, Eberhard-Karls Universitat, Tubingen.

References

Neal, P. and Roberts, G. (2004). Statistical inference and model selection for the 1861 Hagelloch measles epidemic. *Biostatistics* **5** (2), 249.

`MCMCcontrol` *Set control parameters for epinet MCMC algorithm*

Description

Sets parameter values that control the MCMC algorithm used by epinet to produce posterior samples.

Usage

```
MCMCcontrol(nsamp, thinning, extrathinning = FALSE, burnin = 0,
seed = floor(runif(1, 0, 2^30)), etapropsd)
```

Arguments

`nsamp`	number of iterations to run MCMC algorithm.
`thinning`	thinning interval.
`extrathinning`	set to FALSE unless we want to return inferred values of the exposure / infective times and the transmission tree, in which case it is an integer specifying the extra thinning interval. Defaults to FALSE.
`burnin`	number of burn-in iterations to the run the MCMC algorithm. Defaults to 0.
`seed`	seed for random number generation. Defaults to a random value.
`etapropsd`	standard deviation of proposal distributions for eta parameters.

Details

Auxiliary function that can be used to set parameter values that control the MCMC algorithm used by epinet to produce posterior samples. This function is only used in conjunction with the `epinet` function.

nsamp is the number of samples that will be produced for each of the model parameters.

thinning is the thinning interval, e.g., to return every 10th sample, use thinning = 10.

If exposure and / or infective times are being inferred and we wish to return the inferred values of these times (along with the inferred transmission tree), set extrathinning equal to an integer specifying the extra thinning interval for these values. Because returning values for a large number of nodes can be very space-intensive, an extra thinning interval can be given as a multiple of the thinning interval for the other parameters. For example, using thinning = 10 and extrathinning = 20 will return the values of the inferred exposure and infective times and transmission tree every 200 iterations, and the values of the other parameters every 10 iterations. If these inferred values are not desired, set this variable to FALSE.

burnin controls the number of burn-in iterations to be run by the MCMC algorithm before samples begin to become recorded.

etapropsd is a vector of length k, where k is the number of eta (network) parameters in the model, including the intercept. These are tuning parameters for the MCMC algorithm.

Value

A list with arguments as components.

Author(s)

Chris Groendyke <cgroendyke@gmail.com>

References

Groendyke, C. and Welch, D. 2018. epinet: An R Package to Analyze Epidemics Spread across Contact Networks, *Journal of Statistical Software*, **83-11**.

See Also

epinet for generating posterior samples of the parameters, and priorcontrol for specifying prior distributions and their hyperparameters.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat=dcm, eta=-1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist="gamma")
# Set inputs for MCMC algorithm
mcmcinput <- MCMCcontrol(nsamp = 5000, thinning = 10, etapropsd = 0.2)
priorcontrol <- priorcontrol(bprior = c(0, 1), tiprior = c(1, 3), teprior = c(1, 3),
    etaprior = c(0, 10), kiprior = c(2, 8), keprior = c(2, 8), priordists = "uniform")
# Run MCMC algorithm on this epidemic
# Note: Not enough data or iterations for any real inference
examplemcmc <- epinet( ~ 1, exampleepidemic, dcm, mcmcinput, priorcontrol)

## Not run:
# Note: This may take a few minutes to run.
set.seed(1)
N <- 50
mycov <- data.frame(id = 1:N, xpos = runif(N), ypos = runif(N))
dyadCov <- BuildX(mycov, binaryCol = list(c(2, 3)),binaryFunc = c("euclidean"))
# Build network
```

```
eta <- c(0,-7)
net <- SimulateDyadicLinearERGM(N = N, dyadiccovmat = dyadCov, eta = eta)
# Simulate epidemic
epi <- SEIR.simulator(M = net, N = N, beta = 1, ki = 3, thetai = 7, ke = 3, latencydist = "gamma")
# Run MCMC routine on simulated epidemic
mcmcinput <- MCMCcontrol(nsamp = 1000000, thinning = 100, etapropsd = c(1, 1))
priors <- priorcontrol(bprior = c(0, 4), tiprior = c(1, 15), teprior = c(1, 15),
etaprior = c(0, 10, 0, 10), kiprior = c(1, 7), keprior = c(1, 7), priordists = "uniform")
out <- epinet(~ xpos.ypos.L2Dist, epidata = epi, dyadiccovmat = dyadCov,
mcmcinput = mcmcinput, priors = priors)

## End(Not run)
```

`plot.epidemic` *Plot the spread of an epidemic*

Description

Plot the spread of an epidemic over a contact network.

Usage

```
## S3 method for class 'epidemic'
plot(x, lwd = 1, leaf.labs = TRUE, leaf.cex = 0.75,
    zero.at.start = FALSE, main = "Transmission Tree", xlab = "Time",
    ylab= "", e.col = "black", i.col = "red", lty.transmission = 3,
    marktransitions = TRUE, label.trans = "|", cex.trans = 0.5, ...)
```

Arguments

`x`	a simulated epidemic, in the form of the output produced by `SEIR.simulator`.
`lwd`	line width for the (horizontal) line segments representing the exposed and infective periods for each individual. Also controls the line width for the (vertical) line segments showing the transmission pathways.
`leaf.labs`	boolean variable controlling whether the leaf labels (Node IDs) are displayed to the right of their infective period. Defaults to TRUE.
`leaf.cex`	Character expansion factor for the leaf labels, if they are displayed. Defaults to 0.75.
`zero.at.start`	boolean variable governing whether the epidemic times are shifted so that the epidemic begins at time zero. Defaults to FALSE.
`main`	main title for plot.
`xlab`	label for x axis on plot. Defaults to “Time”.
`ylab`	label for y axis on plot. Defaults to blank.
`e.col`	color to be used to plot the individuals’ exposed periods on the plot. Defaults to black.

`i.col` color to be used to plot the individuals' infective periods on the plot. Defaults to red.

`lty.transmission` line type used to mark the (vertical) infection pathway on the plot. Defaults to 3 (dotted).

`marktransitions` boolean variable indicating whether tick marks should be placed at the times where the individuals move from the exposed to the infective state. Defaults to TRUE.

`label.trans` character used to mark transition points, if marktransitions is TRUE. Defaults to "|".

`cex.trans` magnification to be used to for transition labels, if marktransitions is TRUE. Defaults to 0.5.

`...` other plotting arguments to be passed through to the call to plot() that draws the axes and titles.

Details

Plots a simulated epidemic, or indicating the path that the infection took during the epidemic (the transmission tree) and the times that each node entered the Exposed, Infective, and Removed states. The default plotting parameter values work well for epidemics up to about 50 - 60 infecteds and the function requires at least 2 infecteds. For a larger number of infecteds, it is recommended to use `pdf` and adjust plotting dimensions.

Only works for full data, i.e., the transmission tree must be fully specified and all times for infected individuals must be known.

Value

returns no value. Strictly invoked for the plotting side effect.

Author(s)

David Welch <david.welch@auckland.ac.nz>, Chris Groendyke <cgroendyke@gmail.com>

References

Groendyke, C. and Welch, D. 2018. epinet: An R Package to Analyze Epidemics Spread across Contact Networks, *Journal of Statistical Software*, **83-11**.

See Also

`SEIR.simulator` for producing simulated epidemics. `plot.epinet` produces similar plots for transmission trees inferred as part of the `epinet` inference routine.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")

# Plot the simulated epidemic
plot(exampleepidemic)
```

`plot.epinet` *Plot the spread of an epidemic*

Description

Plot the spread of an epidemic over a contact network.

Usage

```
## S3 method for class 'epinet'
plot(x, index = dim(x$transtree)[2],
    lwd = 1, leaf.labs = TRUE, leaf.cex = 0.75, zero.at.start = FALSE,
    main = "Transmission Tree", xlab = "Time", ylab= "",
    e.col = "black", i.col = "red", lty.transmission = 3,
    marktransitions = TRUE, label.trans = "|", cex.trans = 0.5, ...)
```

Arguments

`x`	output from `epinet`.
`index`	a number indicating which of the MCMC samples to plot. Defaults to the final sample in the chain.
`lwd`	line width for the (horizontal) line segments representing the exposed and infective periods for each individual. Also controls the line width for the (vertical) line segments showing the transmission pathways.
`leaf.labs`	boolean variable controlling whether the leaf labels (Node IDs) are displayed to the right of their infective period. Defaults to TRUE.
`leaf.cex`	Character expansion factor for the leaf labels, if they are displayed. Defaults to 0.75.
`zero.at.start`	boolean variable governing whether the epidemic times are shifted so that the epidemic begins at time zero. Defaults to FALSE.

`main`	main title for plot.
`xlab`	label for x axis on plot. Defaults to "Time".
`ylab`	label for y axis on plot. Defaults to blank.
`e.col`	color to be used to plot the individuals' exposed periods on the plot. Defaults to black.
`i.col`	color to be used to plot the individuals' infective periods on the plot. Defaults to red.
`lty.transmission`	line type used to mark the (vertical) infection pathway on the plot. Defaults to 3 (dotted).
`marktransitions`	boolean variable indicating whether tick marks should be placed at the times where the individuals move from the exposed to the infective state. Defaults to TRUE.
`label.trans`	character used to mark transition points, if marktransitions is TRUE. Defaults to "|".
`cex.trans`	magnification to be used to for transition labels, if marktransitions is TRUE. Defaults to 0.5.
`...`	other plotting arguments to be passed through to the call to plot() that draws the axes and titles.

Details

Plots the output from the `link{epinet}` function indicating the path that the infection took during the epidemic (the transmission tree) and the times that each node entered the Exposed, Infective, and Removed states. The default plotting parameter values work well for epidemics up to about 50 - 60 infecteds. For a larger number of infecteds, it is recommended to use `pdf` and adjust plotting dimensions.

Value

returns no value. Strictly invoked for the plotting side effect.

Author(s)

David Welch <david.welch@auckland.ac.nz>, Chris Groendyke <cgroendyke@gmail.com>

References

Groendyke, C. and Welch, D. 2018. epinet: An R Package to Analyze Epidemics Spread across Contact Networks, *Journal of Statistical Software*, **83-11**.

See Also

`epinet` for performing inference on the network and epidemic model parameters. The functions `epi2newick` and `write.epinet` offer other options for outputting inferred parameter samples and transmission trees.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")

# Set inputs for MCMC algorithm
mcmcinput <- MCMCcontrol(nsamp = 5000, thinning = 10, extrathinning = 10,
    etapropsd = 0.2)
priorcontrol <- priorcontrol(bprior = c(0, 1), tiprior = c(1, 3), teprior = c(1, 3),
    etaprior = c(0, 10), kiprior = c(2, 8), keprior = c(2, 8), priordists = "uniform")
# Run MCMC algorithm on this epidemic
# Delete Exposure and Infection times; will infer these in the MCMC algorithm
exampleepidemic[, 3:4] <- NA
examplemcmc <- epinet( ~ 1, exampleepidemic, dcm, mcmcinput, priorcontrol,
     verbose = FALSE)

# Plot starting state of epidemic from chain
plot(examplemcmc, index = 1)

# Plot final state of epidemic from chain
plot(examplemcmc)
```

`print.epidemic` *Prints an epidemict object*

Description

Prints an object created by the `SEIR.simulator` simulation routine.

Usage

```
## S3 method for class 'epidemic'
print(x, ...)
```

Arguments

`x`	an object of class epidemic, produced from the `SEIR.simulator` simulation function.
`...`	other arguments to be passed to the print routine.

Details

Prints the epidemic inference object, including the exposure, infectious, and recovery times of each node in the epidemic.

Value

Strictly invoked for side effect.

Author(s)

Chris Groendyke <cgroendyke@gmail.com>, David Welch <david.welch@auckland.ac.nz>

References

Groendyke, C. and Welch, D. 2018. epinet: An R Package to Analyze Epidemics Spread across Contact Networks, *Journal of Statistical Software*, **83-11**.

See Also

`SEIR.simulator` for simulating an epidemic, `summary.epidemic` for the summary method of an epidemic object, and `plot.epidemic` for plotting a visual display of the epidemic.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")
print(exampleepidemic)
```

`print.epinet` *Print basic information about an epinet object*

Description

Prints some general information about an object created by the `epinet` inference routine.

Usage

```
## S3 method for class 'epinet'
print(x, ...)
```

Arguments

x	an object of class epinet, produced from the epinet inference function.
...	other arguments to be passed to the print routine.

Details

Prints some basic information about an epinet inference object, including the call, network parameters in the model, and number of iterations of the MCMC algorithm.

Value

Strictly invoked for side effect.

Author(s)

Chris Groendyke <cgroendyke@gmail.com>, David Welch <david.welch@auckland.ac.nz>

References

Groendyke, C. and Welch, D. 2018. epinet: An R Package to Analyze Epidemics Spread across Contact Networks, *Journal of Statistical Software*, **83-11**.

See Also

epinet for generating posterior samples of the parameters, and plot.epinet for plotting the posterior samples of the transmission tree.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")
# Set inputs for MCMC algorithm
mcmcinput <- MCMCcontrol(nsamp = 5000, thinning = 10, etapropsd = 0.2)
priorcontrol <- priorcontrol(bprior = c(0, 1), tiprior = c(1, 3), teprior = c(1, 3),
    etaprior = c(0, 10), kiprior = c(2, 8), keprior = c(2, 8), priordists = "uniform")
# Run MCMC algorithm on this epidemic
# Note: Not enough data or iterations for any real inference
examplemcmc <- epinet( ~ 1, exampleepidemic, dcm, mcmcinput, priorcontrol)
print(examplemcmc)

## Not run:
# Note: This may take a few minutes to run.
```

```
set.seed(1)
N <- 50
mycov <- data.frame(id = 1:N, xpos = runif(N), ypos = runif(N))
dyadCov <- BuildX(mycov, binaryCol = list(c(2, 3)),binaryFunc = c("euclidean"))
# Build network
eta <- c(0,-7)
net <- SimulateDyadicLinearERGM(N = N, dyadiccovmat = dyadCov, eta = eta)
# Simulate epidemic
epi <- SEIR.simulator(M = net, N = N, beta = 1, ki = 3, thetai = 7, ke = 3, latencydist = "gamma")
# Run MCMC routine on simulated epidemic
mcmcinput <- MCMCcontrol(nsamp = 1000000, thinning = 100, etapropsd = c(1, 1))
priors <- priorcontrol(bprior = c(0, 4), tiprior = c(1, 15), teprior = c(1, 15),
etaprior = c(0, 10, 0, 10), kiprior = c(1, 7), keprior = c(1, 7), priordists = "uniform")
out <- epinet(~ xpos.ypos.L2Dist, epidata = epi, dyadiccovmat = dyadCov,
mcmcinput = mcmcinput, priors = priors)
print(out)

## End(Not run)
```

`priorcontrol` *Set prior distributions and hyperparameters for epinet MCMC algorithm*

Description

Sets the prior distributions and corresponding hyperparameters to be used in the epinet MCMC algorithm.

Usage

```
priorcontrol(bprior, tiprior, teprior, etaprior, kiprior, keprior,
priordists = "gamma", betapriordist = priordists, thetaipriordist = priordists,
thetaepriordist = priordists, etapriordist = "normal", kipriordist = priordists,
kepriordist = priordists, parentprobmult = 1)
```

Arguments

`bprior`	parameters for beta prior.
`tiprior`	parameters for thetai prior.
`teprior`	parameters for thetae prior.
`etaprior`	parameters for eta priors.
`kiprior`	parameters for ki prior.
`keprior`	parameters for ke prior.
`priordists`	can be “uniform” or “gamma”.
`betapriordist`	can be “uniform” or “gamma”.

`thetaipriordist`
: can be "uniform" or "gamma".

`thetaepriordist`
: can be "uniform" or "gamma".

`etapriordist`
: prior distribution for the network parameters.

`kipriordist`
: can be "uniform" or "gamma".

`kepriordist`
: can be "uniform" or "gamma".

`parentprobmult`
: multiplier for prior probability placed on suspected parent node. Default is a uniform prior assumption.

Details

Auxiliary function that can be used to set prior distributions and parameter values that control the MCMC algorithm used by epinet to produce posterior samples. This function is only used in conjunction with the `epinet` function.

The type of prior distribution (default is gamma / inverse gamma) can be specified for all epidemic parameters (i.e., all parameters except the eta network parameters) using priordists or for each parameter individually. Either uniform or gamma / inverse gamma priors can be chosen. (The two theta parameters use inverse gamma prior distributions, while the other epidemic parameters use gamma priors.)

The parameters of the epidemic parameter prior distributions are given as vectors of (two) hyper-parameters. If the uniform prior is being used for a parameter, then the hyper-parameters are the lower and upper limits of the distribution. If the gamma distribution is being used with parameters c and d, then the prior mean is $c \cdot d$ and the prior variance is $c \cdot d^2$. If the inverse gamma distribution is being used with parameters c and d, then the prior mean is $\frac{d}{c-1}$ and the prior variance is $\frac{d^2}{(c-1)^2 \cdot (c-2)}$.

For the network parameters (the eta parameters), the only prior assumption currently implemented is a set of independent normal distributions.

etaprior contains the hyper-parameters for the prior distributions of the eta parameters. This is a vector of $2k$ values, giving the mean and standard deviation of each distribution (i.e., the first two entries are the mean and standard deviation of the prior distribution for the first eta parameter, the next two entries are the mean and standard deviation of the prior distribution for the second eta parameter, etc.)

The default prior distribution for the parent of each node is uniform on all of the other nodes. To specify a non-uniform distribution, use column 2 of epidata and set parentpriormult to an integer multiplier greater than 1.

Value

A list with arguments as components.

Author(s)

Chris Groendyke <cgroendyke@gmail.com>

References

Groendyke, C. and Welch, D. 2018. epinet: An R Package to Analyze Epidemics Spread across Contact Networks, *Journal of Statistical Software*, **83-11**.

See Also

epinet for generating posterior samples of the parameters and MCMCcontrol for specifying control parameters for the MCMC algorithm.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")
# Set inputs for MCMC algorithm
mcmcinput <- MCMCcontrol(nsamp = 5000, thinning = 10, etapropsd = 0.2)
priorcontrol <- priorcontrol(bprior = c(0, 1), tiprior = c(1, 3), teprior = c(1, 3),
    etaprior = c(0, 10), kiprior = c(2, 8), keprior = c(2, 8), priordists = "uniform")
# Run MCMC algorithm on this epidemic
# Note: Not enough data or iterations for any real inference
examplemcmc <- epinet( ~ 1, exampleepidemic, dcm, mcmcinput, priorcontrol)
```

SEIR.simulator *Simulate an epidemic on a contact network*

Description

Simulate the spread of an epidemic across an (undirected) contact network.

Usage

```
SEIR.simulator(M, N, beta, ki, thetai, ke = ki, thetae = thetai,
    latencydist = "fixed", latencyperiod = 0)
```

Arguments

M	an undirected network, given in edgelist matrix format
N	the number of nodes in the contact network.
beta	the transmission rate of the virus across an edge of the network.
ki	the shape parameter for the removal process for the epidemic.
thetai	the scale parameter for the removal process for the epidemic.
ke	the shape parameter for the removal process for the epidemic.
thetae	the scale parameter for the removal process for the epidemic.

latencydist	type of latency period; can be "fixed" or "gamma".
latencyperiod	length of latency period, if using latencydist == "fixed". Ignored if latencydist == "gamma". Set to 0 to get an SIR epidemic.

Details

Takes as input an undirected network, given in edgelist matrix format, which is the same format returned by SimulateDyadicLinearERGM. Randomly chooses an initial infective individual. The infection spreads randomly across edges in the network according to exponential infective periods with mean $\frac{1}{beta}$. An infective individual remains in the exposed state for a either a fixed period of time given by latencyperiod or a time described by a gamma RV with parameters ke and thetae (mean = $ke \cdot thetae$, var = $ke \cdot thetae^2$). After this exposed period, an infected person moves to the Infected state, at which point they can infect susceptible individuals. The infective individuals are removed after an infective period whose length is governed by a gamma RV with parameters ki and thetai (mean = $ki \cdot thetai$, var = $ki \cdot thetai^2$). Once an individual is removed, they cannot be re-infected and cannot infect others.

Value

matrix consisting of one row for each individual in the population. Each row contains (in columns 1 - 5, respectively): the node infected, the infecting node, the time of infection, the time of transition from exposed to infective, and the time of removal. The times are shifted so that the first removal occurs at time 0. The rows corresponding to the susceptible members of the population (i.e., the individuals that were not infected during the course of the epidemic) are placed after those for the infected individuals.

Author(s)

Chris Groendyke <cgroendyke@gmail.com>, David Welch <david.welch@auckland.ac.nz>,

See Also

SimulateDyadicLinearERGM for simulating an Erdos-Renyi contact network, epinet for performing inference on the network and epidemic model parameters, and plot.epidemic and epi2newick for plotting functions.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N,nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")
```

SimulateDyadicLinearERGM

Simulates an ERGM network using given covariate values

Description

Simulates a random ERGM network using a given matrix of covariate values and a corresponding vector of parameter values.

Usage

```
SimulateDyadicLinearERGM(N, dyadiccovmat, eta)
```

Arguments

N	number of individuals in the population.
dyadiccovmat	matrix of dyadic covariates.
eta	vector of parameters.

Details

dyadiccovmat is an $\binom{N}{2}$ by $(k+2)$ matrix containing the dyadic covariates for the population, where N is the number of individuals in the population and k is the number of dyadic covariates used in the model. The matrix contains one row for each dyad (pair of nodes). Columns 1 and 2 give the ID of the two nodes comprising the dyad, and the remaining k columns give the covariate values; eta is the vector of parameters corresponding to the covariates.

For this class of dyadic independence network, the probability of an edge between individuals i and j is $p_{\{i,j\}}$, where

$$\log\left(\frac{p_{\{i,j\}}}{1-p_{\{i,j\}}}\right) = \sum_k \eta_k X_{\{i,j\},k}$$

More information about this type of model can be found in Groendyke et al. (2012).

Value

a network in edgelist matrix format

Author(s)

David Welch <david.welch@auckland.ac.nz>, Chris Groendyke <cgroendyke@gmail.com>

References

Groendyke, C., Welch, D. and Hunter, D. 2012. A Network-based Analysis of the 1861 Hagelloch Measles Data, *Biometrics*, **68-3**.

See Also

SEIR.simulator for simulating an SEIR epidemic over a network.

Examples

```
# Construct a network of 30 individuals
set.seed(3)
N <- 30
# Build dyadic covariate matrix
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)

# Another example
set.seed(1)
N <- 50
mycov <- data.frame(id = 1:N, xpos = runif(N), ypos = runif(N))
dyadCov <- BuildX(mycov, binaryCol = list(c(2, 3)),binaryFunc = c("euclidean"))
# Build network
eta <- c(0,-7)
net <- SimulateDyadicLinearERGM(N = N, dyadiccovmat = dyadCov, eta = eta)
```

---

summary.epidemic *Summarize simulated epidemic*

Description

Prints a summary of an epidemic simulated by the SEIR.simulator simulation routine.

Usage

```
## S3 method for class 'epidemic'
summary(object, ...)
```

Arguments

object	an object of class epidemic, produced from the SEIR.simulator function.
...	other arguments to be passed to the summary routine.

Details

Prints a summary of the simulated epidemic, including the number of individuals infected over the course of the epidemic, the number remaining susceptible throughout the epidemic, the total size of the population, and length of the epidemic.

Value

Strictly invoked for side effect.

Author(s)

Chris Groendyke <cgroendyke@gmail.com>, David Welch <david.welch@auckland.ac.nz>

See Also

SEIR.simulator for simulating an epidemic, and plot.epidemic for plotting the simulated epidemic.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")
summary(exampleepidemic)
```

---

summary.epinet *Summarize posterior samples from epinet object*

Description

Prints summaries of posterior samples generated by the epinet inference routine.

Usage

```
## S3 method for class 'epinet'
summary(object, ...)
```

Arguments

object	an object of class epinet, produced from the epinet inference function.
...	other arguments to be passed to the summary routine.

Details

Prints summaries of the epidemic and network parameters of an epinet inference object. Epidemic parameters are beta, thetae, ke, thetai, and ki. Network parameters are specified in the model formula, and may include an intercept term.

Value

Strictly invoked for side effect.

Author(s)

Chris Groendyke <cgroendyke@gmail.com>, David Welch <david.welch@auckland.ac.nz>

See Also

epinet for generating posterior samples of the parameters, and plot.epinet for plotting the posterior samples of the transmission tree.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")
# Set inputs for MCMC algorithm
mcmcinput <- MCMCcontrol(nsamp = 5000, thinning = 10, etapropsd = 0.2)
priorcontrol <- priorcontrol(bprior = c(0, 1), tiprior = c(1, 3), teprior = c(1, 3),
    etaprior = c(0, 10), kiprior = c(2, 8), keprior = c(2, 8), priordists = "uniform")
# Run MCMC algorithm on this epidemic
# Note: Not enough data or iterations for any real inference
examplemcmc <- epinet( ~ 1, exampleepidemic, dcm, mcmcinput, priorcontrol)
summary(examplemcmc)

## Not run:
# Note: This may take a few minutes to run.
set.seed(1)
N <- 50
mycov <- data.frame(id = 1:N, xpos = runif(N), ypos = runif(N))
dyadCov <- BuildX(mycov, binaryCol = list(c(2, 3)),binaryFunc = c("euclidean"))
# Build network
eta <- c(0,-7)
net <- SimulateDyadicLinearERGM(N = N, dyadiccovmat = dyadCov, eta = eta)
# Simulate epidemic
epi <- SEIR.simulator(M = net, N = N, beta = 1, ki = 3, thetai = 7, ke = 3, latencydist = "gamma")
```

```
# Run MCMC routine on simulated epidemic
mcmcinput <- MCMCcontrol(nsamp = 1000000, thinning = 100, etapropsd = c(1, 1))
priors <- priorcontrol(bprior = c(0, 4), tiprior = c(1, 15), teprior = c(1, 15),
etaprior = c(0, 10, 0, 10), kiprior = c(1, 7), keprior = c(1, 7), priordists = "uniform")
out <- epinet(~ xpos.ypos.L2Dist, epidata = epi, dyadiccovmat = dyadCov,
mcmcinput = mcmcinput, priors = priors)
summary(out)

## End(Not run)
```

`write.epinet` *Writes posterior samples from an epinet object to an output file*

Description

Outputs posterior samples of an object created by the `epinet` inference routine; creates two output files.

Usage

```
write.epinet(out, filename)
```

Arguments

`out`	an object of class epinet, produced from the `epinet` inference function.
`filename`	the name of the output file.

Details

Writes two output files corresponding to the output produced from the `epinet` inference function. The first is a .log file, containing the posterior samples from the epidemic parameters in tab delimited form. [This .log file can be read by Tracer, which calculates summary statistics and diagnostics, and displays trace plots, histograms, etc.] The second file (which is only written if the transmission trees are returned from the inference routine), is a .trees file, containing the inferred transmission trees, output in Newick format.

Value

Strictly invoked for side effect.

Author(s)

David Welch <david.welch@auckland.ac.nz>, Chris Groendyke <cgroendyke@gmail.com>

References

Rambaut A., Suchard M., Xie D., Drummond A.J. 2014. Tracer v1.6. http://beast.community/tracer.html.

See Also

epinet for generating posterior samples of the parameters, print.epinet and summary.epinet for printing basic summary information about an epinet object, epi2newickmcmc for printing an inferred transmission tree to the screen in Newick format, and plot.epinet for plotting the posterior samples of the transmission tree.

Examples

```
# Simulate an epidemic through a network of 30
set.seed(3)
N <- 30
# Build dyadic covariate matrix (X)
# Have a single covariate for overall edge density; this is the Erdos-Renyi model
nodecov <- matrix(1:N, nrow = N)
dcm <- BuildX(nodecov)
# Simulate network and then simulate epidemic over network
examplenet <- SimulateDyadicLinearERGM(N, dyadiccovmat = dcm, eta = -1.8)
exampleepidemic <- SEIR.simulator(examplenet, N = 30,
    beta = 0.3, ki = 2, thetai = 5, latencydist = "gamma")
# Set inputs for MCMC algorithm
mcmcinput <- MCMCcontrol(nsamp = 5000, thinning = 10, etapropsd = 0.2)
priorcontrol <- priorcontrol(bprior = c(0, 1), tiprior = c(1, 3), teprior = c(1, 3),
    etaprior = c(0, 10), kiprior = c(2, 8), keprior = c(2, 8), priordists = "uniform")
# Run MCMC algorithm on this epidemic
# Note: Not enough data or iterations for any real inference
examplemcmc <- epinet( ~ 1, exampleepidemic, dcm, mcmcinput, priorcontrol)
## Not run: write.epinet(examplemcmc, "examplemcmc")

## Not run:
# Note: This may take a few minutes to run.
set.seed(1)
N <- 50
mycov <- data.frame(id = 1:N, xpos = runif(N), ypos = runif(N))
dyadCov <- BuildX(mycov, binaryCol = list(c(2, 3)),binaryFunc = c("euclidean"))
# Build network
eta <- c(0,-7)
net <- SimulateDyadicLinearERGM(N = N, dyadiccovmat = dyadCov, eta = eta)
# Simulate epidemic
epi <- SEIR.simulator(M = net, N = N, beta = 1, ki = 3, thetai = 7, ke = 3, latencydist = "gamma")
# Run MCMC routine on simulated epidemic
mcmcinput <- MCMCcontrol(nsamp = 1000000, thinning = 100, etapropsd = c(1, 1))
priors <- priorcontrol(bprior = c(0, 4), tiprior = c(1, 15), teprior = c(1, 15),
etaprior = c(0, 10, 0, 10), kiprior = c(1, 7), keprior = c(1, 7), priordists = "uniform")
out <- epinet(~ xpos.ypos.L2Dist, epidata = epi, dyadiccovmat = dyadCov,
mcmcinput = mcmcinput, priors = priors)
write.epinet(out, "SampleInferenceOutput")

## End(Not run)
```

Index

∗ **datasets**
Hagelloch, 9
∗ **graphs**
BuildX, 2
epi2newick, 3
epinet, 4
MCMCcontrol, 10
plot.epidemic, 12
plot.epinet, 14
print.epidemic, 16
print.epinet, 17
priorcontrol, 19
SEIR.simulator, 21
SimulateDyadicLinearERGM, 23
summary.epidemic, 24
summary.epinet, 25
write.epinet, 27

BuildX, 2, *5*, *7*

cat, *3*

epi2newick, 3, *7*, *15*, *22*
epi2newickmcmc, *28*
epi2newickmcmc *(*epi2newick*)*, 3
epibayesmcmc *(*epinet*)*, 4
epinet, *2–4*, 4, *10, 11*, *13–15*, *17, 18*, *20–22*, *25–28*
ess, 8

formula, *5*

Hagelloch, 9
HagellochDyadCov *(*Hagelloch*)*, 9
HagellochTimes *(*Hagelloch*)*, 9

MCMCcontrol, *5*, *7*, 10, *21*

pdf, *13*, *15*
plot.epidemic, 12, *17*, *22*, *25*
plot.epinet, *4*, *7*, *13*, 14, *18*, *26*, *28*
print.epidemic, 16
print.epinet, *4*, 17, *28*
priorcontrol, *5*, *7*, *11*, 19

SEIR.simulator, *3*, *12, 13*, *16, 17*, 21, *24, 25*
SimulateDyadicLinearERGM, *2*, *22*, 23
summary.epidemic, *17*, 24
summary.epinet, *4*, 25, *28*

write.epinet, *4*, *7*, *15*, 27