Package 'BBNI'

November 3, 2026

Title Bayesian Inference of Boolean Genetic Networks

Version 0.2.3

Description Implements a fully Bayesian Markov chain Monte Carlo (MCMC) approach for inferring the topology and Boolean logic transition functions of gene regulatory networks from noisy, binary time-series expression data. Network structure and Boolean rules are sampled jointly from their posterior distribution, providing principled uncertainty quantification rather than a single point estimate. Method described in Han et al. (2014) <doi:10.1371/journal.pone.0115806>.

Depends R (>= 4.1.0)

License BSD_3_clause + file LICENSE

URL https://anson-li8.github.io/BBNI/, https://github.com/anson-li8/BBNI

BugReports https://github.com/anson-li8/BBNI/issues

Encoding UTF-8

Language en-US

Imports stats, igraph, graphics, utils

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

LazyData true

Config/roxygen2/version 8.1.1

NeedsCompilation no

Author Anson Li [aut, cre], Shengtong Han [aut]

Maintainer Anson Li <liyuanrui618@gmail.com>

Repository CRAN

Date/Publication 2026-11-03 05:10:02 UTC

Contents

bbni_class . . . 2
GenerateNetwork . . . 3
GenerateSample . . . 3
plot.bbni . . . 5
plot_bbni . . . 6
plot_network . . . 7
plot_trace . . . 8
print.bbni . . . 9
run_bbni . . . 9
summary.bbni . . . 12
yeast_data . . . 13

Index **14**

`bbni_class` *BBNI MCMC Output Class*

Description

`run_bbni()` returns a list of class `bbni` containing the sampled path of the Metropolis-within-Gibbs chain. Classed objects have `print()`, `summary()`, and `plot()` methods; all list components remain directly accessible via `$`.

Details

Components:

- `networks`: list of MCMC sampled transition-function matrices.
- `log_posterior`: numeric vector of collapsed log-posterior values.
- `post_edge_prob`: matrix of marginal posterior edge probabilities; entry `[i, j]` is the probability of the directed edge `j -> i`.
- `burn_in`: the burn-in ratio used for posterior summarization.

Run metadata (`num.node`, `SampleSize`, `num_update`, `timeseries`) is stored as attributes.

See Also

`run_bbni()`, `print.bbni()`, `summary.bbni()`, `plot.bbni()`

GenerateNetwork *Generate Initial Network Topology*

Description

Randomly generates a valid directed acyclic graph (DAG) topology T and assigns a corresponding Boolean transition function F to each node. The algorithm samples parent set configurations, keeping the constraint that the maximum in-degree for any node is 2, and further ensures the resulting structure does not contain directed cyclic loops.

Usage

```
GenerateNetwork(num.node)
```

Arguments

num.node	An integer representing the total number of genes/nodes in the network.

Value

A square transition function matrix combining the initial DAG topology with randomly assigned Boolean logic functions. Elements with a value of 0 indicate no directed edge, while positive integers indicate the presence of an edge and specify the defining Boolean function type (codes 1-12: 1-10 for two-input functions, 11-12 for one-input identity/negation functions).

Examples

```
# Generate a true network topology and Boolean rules for 5 nodes
set.seed(123)
true_network <- GenerateNetwork(num.node = 5)

# Graph the network with built-in exported function plot_network
plot_network(true_network)
```

GenerateSample *Simulate Boolean Network Observation Dataset*

Description

Simulates a synthetic binary observation dataset (G) based on a given directed acyclic graph topology with Boolean rules. It starts by running independent Bernoulli trials on root nodes. The remaining non-root nodes are calculated based on their assigned Boolean logic functions and parent states. A pre-generated binary noise matrix is applied via a bitwise XOR operation to occasionally flip the Boolean outputs, injecting natural biological noise expected by the model.

Usage

```
GenerateSample(
  trans_matrix,
  SampleSize = 50,
  num.node = nrow(trans_matrix),
  para = rep(0.5, nrow(trans_matrix)),
  error = matrix(0, nrow = nrow(trans_matrix), ncol = SampleSize),
  timeseries = TRUE
)
```

Arguments

`trans_matrix`	A square matrix combining the network topology T and integer-coded Boolean logic functions F assigned to each directed edge.
`SampleSize`	An integer representing the total number of samples or time points to simulate. Defaults to 50 if not specified.
`num.node`	An integer representing the total number of network nodes. Defaults to `nrow(trans_matrix)` if not specified.
`para`	A numeric vector of baseline success probabilities (θ_i) used to generate the expression states of root nodes via independent Bernoulli trials. Defaults to a `rep(0.5, nrow(trans_matrix))` if not specified.
`error`	A pre-generated binary noise matrix applied to occasionally flip Boolean outputs, injecting natural noise. Defaults to a zero matrix of size `nrow(trans_matrix)` x `SampleSize` if not specified (no noise).
`timeseries`	Logical. If TRUE, simulates time-lagged data where child state at time t is determined by parent stated at $t-1$. If FALSE, simulates independent samples where child and parent states are resolved simultaneously. Defaults to TRUE.

Value

A simulated binary gene expression matrix G, where rows represent individual genes/nodes and columns represent samples.

Examples

```
# 1. Generate a 5-node network
set.seed(123)
num_nodes <- 5
sample_size <- 10
true_network <- GenerateNetwork(num.node = num_nodes)

# 2. Set baseline probabilities and simulate zero-noise error matrix
root_probs <- rep(0.5, num_nodes)
error_matrix <- matrix(0, nrow = num_nodes, ncol = sample_size)

# 3. Generate cross-sectional (independent mode) synthetic data
dummy_data <- GenerateSample(
  trans_matrix = true_network,
```

```
  SampleSize = sample_size,
  para = root_probs,
  error = error_matrix,
  timeseries = FALSE
)
print(dummy_data)
```

`plot.bbni` *Plot a BBNI MCMC Object*

Description

Produces a trace plot of the log-posterior over MCMC iterations. This method delegates to `plot_trace()`. For network visualization, use `plot_bbni()`.

Usage

```
## S3 method for class 'bbni'
plot(x, ...)
```

Arguments

`x`	A `bbni` object returned by `run_bbni()`.
`...`	Extra arguments passed to `plot_trace()`.

Value

The plot object (invisibly).

Examples

```
set.seed(123)
net <- GenerateNetwork(5)
data <- GenerateSample(net, 100)
res <- run_bbni(data, num_update = 50)
plot(res)
```

plot_bbni *Plot the Inferred Bayesian Boolean Network*

Description

Visualizes the causal network structure inferred by the BBNI MCMC sampler. This function takes the marginal posterior probability of each directed edge and plots the network using the igraph package. Edges with posterior probabilities below the specified threshold are omitted from the plot.

Usage

```
plot_bbni(
  results,
  threshold = 0.5,
  node_names = NULL,
  true_network = NULL,
  ...
)
```

Arguments

results	The list returned by run_bbni(), containing networks and log_posterior.
threshold	Numeric. The minimum posterior probability required to draw an edge. Defaults to 0.5.
node_names	Character vector. Optional names for the nodes. Defaults to "N1", "N2", etc.
true_network	Optional square matrix representing the true network topology. If provided, edges will be color-coded to indicate true positives (along with displaying wrong function inferences), false positives, and false negatives. Purely for simulation purposes.
...	Additional graphical parameters passed to igraph::plot.igraph().

Value

An invisible igraph object.

Examples

```
# 1. Generate synthetic network and time-series data
set.seed(123)
true_network <- GenerateNetwork(num.node = 5)
dummy_data <- GenerateSample(true_network, SampleSize = 15)

# 2. Run BBNI sampler
prior_para <- matrix(3, nrow = 6, ncol = 2)
prior_para[6, 1] <- 2
prior_para[6, 2] <- 100
```

```
results <- run_bbni(dummy_data, prior_para = prior_para, num_update = 100)

# 3. Plot inferred network
plot_bbni(results, true_network = true_network, threshold = 0.5)
```

---

plot_network *Plot a Single Boolean Network*

Description

Visualizes a single directed network topology. For instance, a synthetic "true" network generated by `GenerateNetwork()`.

Usage

```
plot_network(trans_matrix, node_names = NULL, ...)
```

Arguments

`trans_matrix`	A square matrix representing the network topology and Boolean rules (e.g., the output of `GenerateNetwork()`). Elements greater than 0 indicate a directed edge.
`node_names`	Character vector. Optional names for the nodes. Defaults to "N1", "N2", etc.
`...`	Additional graphical parameters passed to `igraph::plot.igraph()`.

Value

An invisible `igraph` object.

Examples

```
# 1. Generate synthetic 5-node network topology
set.seed(123)
true_network <- GenerateNetwork(num.node = 5)

# 2. Plot the generated network
plot_network(true_network)
```

plot_trace *Plot MCMC Trace for BBNI*

Description

Generates a trace plot of the log-posterior values over iterations of the MCMC to visually assess convergence and stability of the executed Markov chain. Burn-in line is graphed to show when the data started to be utilized for edge-probability calculations.

Usage

```
plot_trace(results, every = 1)
```

Arguments

results	The list returned by run_bbni(), containing networks and log_posterior.
every	An integer specifying the thinning interval (sampling frequency) for plotting. Default is 1, which plots all log-posterior values. Values greater than 1 plot every every-th iteration.

Value

A base R trace plot.

Examples

```
# 1. Generate synthetic network and time-series data
set.seed(123)
true_network <- GenerateNetwork(num.node = 5)
dummy_data <- GenerateSample(true_network, SampleSize = 15)

# 2. Run BBNI sampler
prior_para <- matrix(3, nrow = 6, ncol = 2)
prior_para[6, 1] <- 2
prior_para[6, 2] <- 100

results <- run_bbni(dummy_data, prior_para = prior_para, num_update = 100)

# 3. Visualize MCMC results
plot_trace(results)
```

print.bbni *Print a BBNI MCMC Object*

Description

The default display method for bbni objects. It provides a one-line overview instead of a statistical summary; use summary() for posterior statistics and plot() for trace plots.

Usage

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

Arguments

x	A bbni object returned by run_bbni().
...	Additional arguments (ignored).

Value

x, invisibly.

Examples

```
set.seed(123)
net <- GenerateNetwork(5)
data <- GenerateSample(net, 100)
res <- run_bbni(data, num_update = 50)
print(res)
```

run_bbni *Execute Metropolis-within-Gibbs MCMC Sampler for Boolean Networks*

Description

Executes a Metropolis-within-Gibbs Markov chain Monte Carlo (MCMC) algorithm to sample from the joint posterior distribution of Directed Acyclic Graph (DAG) topologies (T) and Boolean logic transition functions (F). The algorithm iterates through individual network nodes and proposes parent set mutations (edge additions, removals, or swaps) paired with transition function reassignments to one of 14 candidate Boolean rules (stored as codes 1-12 in the transition matrix). Proposed states transitions are strictly verified to follow the DAG constraint and evaluated with a Metropolis-Hastings acceptance threshold using log-posterior values.

Usage

```
run_bbni(
  GeneData,
  num.node = nrow(GeneData),
  SampleSize = ncol(GeneData),
  prior_para = NULL,
  num_update = 4000,
  penalty = 0.1,
  prop.ratio = 0.1,
  verbose = FALSE,
  timeseries = TRUE,
  burn_in = 0.7
)
```

Arguments

GeneData	A binary empirical observation matrix (G), where rows represent individual network nodes (genes) and columns represent independent samples or sequential time points.
num.node	An integer representing the total number of network nodes. Defaults to `nrow(GeneData)` if not specified.)
SampleSize	An integer representing the total number of time points or independent samples in the dataset. Defaults to `ncol(GeneData)` if not specified.
prior_para	A matrix (with dimensions `(num.node + 1) x 2`) of Beta prior hyperparameters α and β for root node probabilities and the global noise parameter e. Defaults to a flat prior if not specified.
num_update	An integer representing the total number of MCMC iterations to perform. Defaults to 4000 if not specified.
penalty	Structural-prior hyperparameter in $(0, 1]$. A value of 1 corresponds to a uniform prior over valid network topologies; values below 1 apply an edge-count penalty that favors sparser networks. Defaults to 0.1 if not specified.
prop.ratio	A numeric value between 0 and 1 giving the final probability of using the empirical proposal distribution after the first 10% of outer iterations. During the first 10% of outer iterations, the empirical proposal is used with probability 0.9. Defaults to 0.1.
verbose	Logical. If TRUE, prints verbose MCMC iteration progress to the console. Default is FALSE.
timeseries	Logical. If TRUE, the algorithm assumes a time-series dataset. If FALSE, the algorithm assumes independent samples. Default is TRUE.
burn_in	A numeric value between 0 and 1 representing the proportion of initial MCMC samples to discard as burn-in. Defaults to 0.7 if not specified.

Details

Posterior edge probabilities (`post_edge_prob`) are computed from one thinned sample per outer iteration (a full Gibbs sweep) after discarding `burn_in`, as designed in the original method paper.

Value

A list containing the full trajectory of the MCMC chain. Specifically, `networks` (a list of sampled transition function matrices) and `log_posterior` (a numeric vector of log-posterior scores for each iteration). These represent samples drawn from the marginal posterior distribution $P(T, F|G)$ used for Bayesian model averaging. Additionally, the `post_edge_prob` (matrix of marginal posterior edge probabilities) and `burn_in` ratio are returned in the list. The result is a classed `bbni` object, so `print()`, `summary()`, and `plot()` methods are available (see `?bbni_class`).

Examples

```
# 1. Define network parameters
set.seed(235)
num_nodes <- 8
sample_size <- 50

# 2. Generate true network and simulate data
true_network <- GenerateNetwork(num.node = num_nodes)

# Set up Beta priors for root-node probabilities and the noise rate
prior_para <- matrix(3, nrow = num_nodes + 1, ncol = 2)
prior_para[num_nodes + 1, 1] <- 2
prior_para[num_nodes + 1, 2] <- 100

# Simulate parameters
para <- numeric(num_nodes + 1)
for (i in 1:(num_nodes + 1)) {
  para[i] <- stats::rbeta(1, prior_para[i, 1], prior_para[i, 2])
}
para[num_nodes + 1] <- 0.1 # Fixed noise rate for simulation

error_matrix <- matrix(stats::rbinom(num_nodes * sample_size, 1, para[num_nodes + 1]),
  nrow = num_nodes, ncol = sample_size
)

dummy_data <- GenerateSample(
  trans_matrix = true_network,
  SampleSize = sample_size,
  para = para,
  error = error_matrix
)

# 3. Run the MCMC sampler (silently)
mcmc_results <- run_bbni(
  GeneData = dummy_data,
  prior_para = prior_para,
  num_update = 80, # Scaled down for example speed
  prop.ratio = 0.1
)

# 4. Visualize results
plot_bbni(mcmc_results, true_network = true_network, threshold = 0.5)
plot_trace(mcmc_results)
```

summary.bbni *Posterior Summary of a BBNI MCMC Object*

Description

Prints a statistical summary of the MCMC run and returns a structured list of posterior quantities for further usage in downstream code.

Usage

```
## S3 method for class 'bbni'
summary(object, threshold = 0.5, n_top = 5, ...)
```

Arguments

object	A bbni object returned by run_bbni().
threshold	Numeric. Posterior probability threshold for counting strong edges.
n_top	Integer. Number of highest-posterior edges to display and include in top_edges.
...	Additional arguments (ignored).

Value

A list with components num_nodes, num_update, burn_in, final_log_posterior, n_strong_edges, and top_edges (data frame with columns child, parent, and posterior).

Examples

```
set.seed(123)
net <- GenerateNetwork(5)
data <- GenerateSample(net, 100)
res <- run_bbni(data, num_update = 50)
summary(res, threshold = 0.6, n_top = 3)
```

yeast_data *Yeast Cell-Cycle Gene Expression Data*

Description

A binary empirical dataset with the gene expression states of 14 yeast cell-cycle genes across 385 complete pooled experimental conditions. The data were filtered to the 14 genes used by Han et al. (2014), columns with missing values were removed, and each gene was binarized by its row mean.

Usage

```
data(yeast_data)
```

Format

A numeric matrix with 14 rows (representing individual genes/nodes) and 385 columns (representing independent samples or sequential time points).

Source

Han, S., Wong, R. K. W., Lee, T. C. M., Shen, L., Li, S.-Y. R., & Fan, X. (2014). A Full Bayesian Approach for Boolean Genetic Network Inference. *PLOS ONE*, 9(12), e115806. doi:10.1371/journal.pone.0115806

Index

∗ **datasets**
yeast_data, 13

bbni_class, 2

GenerateNetwork, 3
GenerateSample, 3

plot.bbni, 5
plot.bbni(), *2*
plot_bbni, 6
plot_bbni(), *5*
plot_network, 7
plot_trace, 8
plot_trace(), *5*
print.bbni, 9
print.bbni(), *2*

run_bbni, 9
run_bbni(), *2*

summary.bbni, 12
summary.bbni(), *2*

yeast_data, 13