Package ‘GRENITS’

April 25, 2017

Type Package

Title Gene Regulatory Network Inference Using Time Series

Version 1.28.0

Date 2012-03-15

Author Edward Morrissey

Maintainer Edward Morrissey <edward.morrissey@gmail.com>

Description The package offers four network inference statistical models using Dynamic Bayesian Networks and Gibbs Variable Selection: a linear interaction model, two linear interaction models with added experimental noise (Gaussian and Student distributed) for the case where replicates are available and a non-linear interaction model.

License GPL (>= 2)

LazyLoad yes

Depends R (>= 2.12.0), Rcpp (>= 0.8.6), RcppArmadillo (>= 0.2.8), ggplot2 (>= 0.9.0)

Imports graphics, grDevices, reshape2, stats, utils

Suggests network

LinkingTo Rcpp, RcppArmadillo

biocViews NetworkInference, GeneRegulation, TimeCourse, GraphAndNetwork, GeneExpression, Network, Bayesian

NeedsCompilation yes

R topics documented:

analyse.output .................................................. 2
Athaliana_ODE ...................................................... 3
Athaliana_ODE_4NoiseReps ...................................... 4
LinearNet .......................................................... 4
mcmc.defaultParams_gauss ...................................... 6
mcmc.defaultParams_Linear ..................................... 7
mcmc.defaultParams_nonLinear .................................. 8
mcmc.defaultParams_student ................................... 9
NonLinearNet ..................................................... 11
plotPriors ....................................................... 12
**Description**

Analyse output from network inference functions. Basic convergence and analysis plots.

**Usage**

```r
analyse.output(output.folder, timeSeries = NULL)
```

**Arguments**

- `output.folder`: Name of folder (including path) where chains are kept
- `timeSeries`: Only used by NonLinearModel analysis. Data matrix containing gene expression time series. Where genes will be placed in rows and time points in columns.

**Details**

Read first two chains run and plot some basic convergence plots (ConvergencePlots.pdf), analysis plots (AnalysisPlots.pdf), as well as inferred network probabilities in two formats (NetworkProbability_List.txt and NetworkProbability_Matrix.txt).

**Value**

The output of the analysis will be four files (five if nonLinearNet). The contents of the two plot files change depending on the network inference function used.

- **ConvergencePlots.pdf**: Basic convergence plots. The posterior means of each variable are compared.
- **AnalysisPlots.pdf**: Heatmap plot of network link probabilities as well as marginal network uncertainty plot. A plot of the number of links predicted by the model for a given probability threshold. For ReplicatesNet_student, the posterior distribution of the degrees of freedom are also plotted. For NonLinearNet, the posterior of the smoothness parameter is plotted.
- **NetworkProbability_List.txt**: Posterior probabilities for each network connection in list format, including posterior interaction strength for linear models. Can be imported with network analysis software such as cytoscape.
- **NetworkProbability_Matrix.txt**: Posterior probabilities for each network connection in matrix format.
- **ProbNumParents.txt**: Posterior probabilities for number of regulators for each gene.
- **InferredFunctionPlots.pdf**: (Only for nonLinearNet) Posterior distribution of predicted functions. Data values are plotted as circles.
References


See Also

NonLinearNet, LinearNet, ReplicatesNet_student ,ReplicatesNet_gauss.

Examples

# Load A. thaliana circadian clock ODE generated data
data(Athaliana_ODE)
# Folder where raw runs will be kept and analysed
output.folder <- paste(tempdir(), "/Example_LinearNet",sep="")
# Run network inference, place raw results in output.folder
LinearNet(output.folder, Athaliana_ODE)
# Analyse raw results, place analysis plots and files in output.folder
analyse.output(output.folder)

Athaliana_ODE	Gene expression time series generated with ODE model

Description

This data set was generated using a circadian clock regulatory network ODE model. The data was simulated using COPASI, subsampled to produce hourly data and after this the log was calculated.

Usage

data(Athaliana_ODE)

Format

A matrix containing genes in rows and (1h) time points in columns.

References


Athaliana_ODE_4NoiseReps

Gene expression time series generated with ODE model with added noise

Description

This data set was generated using a circadian clock regulatory network ODE model after which noise was added. The data was simulated using COPASI, subsampled to produce hourly data and after this the log was calculated. Further to this, four replicates were produced by adding student noise.

Usage

data(Athaliana_ODE_4NoiseReps)

Format

A matrix containing genes in rows and (1h) time points in columns. The four replicates are appended (columns)

References


LinearNet

Dynamic Bayesian Network Inference Using Linear Interactions

Description

Run Bayesian inference of linear interaction network. The function generates MCMC chains that can later be analysed.

Usage

LinearNet( resultsFolder, timeSeries, ParamVec = NULL, chains = 2, user.seeds = NULL, Regulators = NULL, fixMe = NULL)
LinearNet

Arguments

resultsFolder  Name of output folder. The folder will be created and the output of the run will be placed there.
timeSeries  Data matrix containing gene expression time series. Where genes will be placed in rows and time points in columns. Gene names may be included as row names.
ParamVec  A parameter vector created using "mcmc.defaultParams_Linear". If none is given, default parameters will be used. The vector contains parameters associated to the priors as well as MCMC run length. (See mcmc.defaultParams_Linear)
chains  Number of MCMC chains to run.
user.seeds  An optional vector with seeds to use for MCMC chains.
Regulators  An optional vector with the indices of which genes are regulators. If provided, all non-regulator genes will not be allowed to regulate.
fixMe  An optional matrix of size genes x genes, where columns represent regulators and rows regulated genes. The matrix informs the model of network connections known to be present/absent. For each position use either 0 (no regulation, fix off), 1 (known regulatory interaction, fix on) or NaN (no information, do not fix).

Value

For each chain run, a folder (chain1, chain2, ...) will be created and the output of the MCMC run will be placed there. The files will be B_mcmc (the coefficients of the linear regression), Gamma_mcmc (the indicator variables of Gibbs variable selection), Lambda_mcmc (the precision of each regression), Mu_mcmc (the intercept of each regression) and Rho_mcmc (the network connectivity parameter).

References


See Also

mcmc.defaultParams_Linear, analyse.output.

Examples

# Load A. thaliana circadian clock ODE generated data
data(Athaliana_ODE)
# Folder where raw runs will be kept and analysed
output.folder <- paste(tempdir(), "/Example_LinearNet",sep="")
# Run network inference, place raw results in output.folder
LinearNet(output.folder, Athaliana_ODE)
# Analyse raw results, place analysis plots and files in output.folder
analyse.output(output.folder)
mcmc.defaultParams_gauss

Default Parameters for Linear Model with Gaussian distributed replicates

Description
Create parameter vector with default parameters for ReplicatesNet_gauss function

Usage
mcmc.defaultParams_gauss()

Details
Use this function to generate a template parameter vector to use non-default parameters for the ReplicatesNet_gauss model.

Value
Returns a single vector with the following elements (in this order):

(1) samples Number of MCMC iterations to run
(2) burn.in Number of initial iterations to discard as burn in
(3) thin Subsampling frequency
(4) c Shape parameter 1 for Beta(c,d) prior on rho (connectivity parameter)
(5) d Shape parameter 2 for Beta(c,d) prior on rho (connectivity parameter)
(6) sigma.s Standard deviation parameter for N(0,sigma.s) prior on B (Regression coefficients)
(7) a Shape parameter for Gamma(a,b) prior on lambda (Regression precision)
(8) b Rate parameter for Gamma(a,b) prior on lambda (Regression precision)
(9) a_exp Shape parameter for Gamma(a_exp,b_exp) prior on tau (Replicates precision)
(10) b_exp Rate parameter for Gamma(a_exp,b_exp) prior on tau (Replicates precision)
(11) sigma.mu Standard deviation parameter for N(0,sigma.mu) prior on mu (Regression intercept)
(12) fix.y.iter Number of iterations for which sampled data Y is fixed

References

See Also
plotPriors, ReplicatesNet_gauss.
Examples

```r
# Get default parameters
linearNet_Gauss.params <- mcmc.defaultParams_gauss()

# Change run length

# Change prior regression precision
linearNet_Gauss.params[7] <- 0.001
linearNet_Gauss.params[8] <- 0.001

# Plot to visualise changes
plotPriors(linearNet_Gauss.params)

## Use to run ReplicatesNet_gauss ...
```

---

### mcmc.defaultParams_Linear

**Default Parameters for Linear Model**

**Description**

Create parameter vector with default parameters for LinearNet function

**Usage**

```r
mcmc.defaultParams_Linear()
```

**Details**

Use this function to generate a template parameter vector to use non-default parameters for the LinearNet model.

**Value**

Returns a single vector with the following elements (in this order):

1. `samples`  Number of MCMC iterations to run.
2. `burn.in`  Number of initial iterations to discard as burn in.
3. `thin`  Subsampling frequency
4. `c`  Shape parameter 1 for Beta(c,d) prior on rho (connectivity parameter)
5. `d`  Shape parameter 2 for Beta(c,d) prior on rho (connectivity parameter)
6. `sigma.s`  Standard deviation parameter for N(0,sigma.s) prior on B (Regression coefficients)
7. `a`  Shape parameter for Gamma(a,b) prior on lambda (Regression precision)
8. `b`  Rate parameter for Gamma(a,b) prior on lambda (Regression precision)
9. `sigma.mu`  Standard deviation parameter for N(0,sigma.mu) prior on mu (Regression intercept)
mcmc.defaultParams_nonLinear

Default Parameters for non-Linear Model

Description

Create parameter vector with default parameters for NonLinearNet function

Usage

mcmc.defaultParams_nonLinear()

Details

Use this function to generate a template parameter vector to use non-default parameters for the NonLinearNet model.

Value

Returns a single vector with the following elements (in this order): 

(1) samples Number of MCMC iterations to run. 

(2) burn.in Number of initial iterations to discard as burn in.
(3) thin  Subsampling frequency
(4) c    Shape parameter 1 for Beta(c,d) prior on rho (connectivity parameter)
(5) d    Shape parameter 2 for Beta(c,d) prior on rho (connectivity parameter)
(6) trunc Truncation parameter for InvertedPareto prior on tau (smoothness parameter)
(7) tau0 Precision parameter for N(0, tau0^(-0.5)) prior on B (first two coefficients)
(8) M    Number of knots used for each spline function
(9) a    Shape parameter for Gamma(a,b) prior on lambda (Regression precision)
(10) b   Rate parameter for Gamma(a,b) prior on lambda (Regression precision)
(11) sigma.mu  Standard deviation parameter for N(0,sigma.mu) prior on mu (Regression intercept)
(12) a_pareto Pareto parameter for InvertedPareto prior on tau (smoothness parameter)

References

See Also
plotPriors, NonLinearNet.

Examples

# Get default parameters
nonLinearNet.params <- mcmc.defaultParams_nonLinear()

# Change run length
nonLinearNet.params[1] <- 150000

# Change prior on smoothness parameter
nonLinearNet.params[6] <- 30000  # Change truncation
nonLinearNet.params[12] <- 3  # Concentrate more mass close to linear region

# Plot to check changes
plotPriors(nonLinearNet.params)

## Use to run LinearNet ...

mcmc.defaultParams_student

Default Parameters for Linear Model with Student distributed replicates

Description
Create parameter vector with default parameters for ReplicatesNet_student function

Usage
mcmc.defaultParams_student()
Details

Use this function to generate a template parameter vector to use non-default parameters for the ReplicatesNet_student model.

Value

Returns a single vector with the following elements (in this order):

1. samples  Number of MCMC iterations to run
2. burn.in   Number of initial iterations to discard as burn in
3. thin      Subsampling frequency
4. c         Shape parameter 1 for Beta(c,d) prior on rho (connectivity parameter)
5. d         Shape parameter 2 for Beta(c,d) prior on rho (connectivity parameter)
6. sigma.s   Standard deviation parameter for N(0,sigma.s) prior on B (Regression coefficients)
7. a         Shape parameter for Gamma(a,b) prior on lambda (Regression precision)
8. b         Rate parameter for Gamma(a,b) prior on lambda (Regression precision)
9. a_exp     Shape parameter for Gamma(a_exp,b_exp) prior on tau (Replicates precision)
10. b_exp    Rate parameter for Gamma(a_exp,b_exp) prior on tau (Replicates precision)
11. a_deg    Shape parameter for Gamma(a_deg,b_deg) prior on nu (Student degrees of freedom)
12. b_deg    Rate parameter for Gamma(a_deg,b_deg) prior on nu (Student degrees of freedom)
13. sigma.mu Standard deviation parameter for N(0,sigma.mu) prior on mu (Regression intercept)
14. fix.y.iter Number of iterations for which sampled data Y is fixed

References


See Also

plotPriors, ReplicatesNet_student.

Examples

# Get default parameters
linearNet_Student.params <- mcmc.defaultParams_student()

# Change run length
linearNet_Student.params[1] <- 200000

# Change prior regression precision
linearNet_Student.params[7] <- 0.001
linearNet_Student.params[8] <- 0.001
# Plot to visualise changes
plotPriors(linearNet_Student.params)

## Use to run ReplicatesNet_student ...

---

**NonLinearNet**

Dynamic Bayesian Network Inference Using Non-Linear Interactions

**Description**

Run Bayesian inference of non-linear interaction network. Non linear interactions are modelled using Penalised Splines. The function generates MCMC chains that can later be analysed.

**Usage**

```r
NonLinearNet(resultsFolder, timeSeries, ParamVec = NULL, chains = 2, user.seeds = NULL, Regulators = NULL, fixMe = NULL)
```

**Arguments**

- `resultsFolder`: Name of output folder. The folder will be created and the output of the run will be placed there.
- `timeSeries`: Data matrix containing gene expression time series. Where genes will be placed in rows and time points in columns. Gene names may be included as row names.
- `ParamVec`: A parameter vector created using "mcmc.defaultParams_nonLinear". If none is given, default parameters will be used. The vector contains parameters associated to the priors as well as MCMC run length. (See mcmc.defaultParams_nonLinear)
- `chains`: Number of MCMC chains to run.
- `user.seeds`: An optional vector with seeds to use for MCMC chains.
- `Regulators`: An optional vector with the indices of which genes are regulators. If provided, all non-regulator genes will not be allowed to regulate.
- `fixMe`: An optional matrix of size genes x genes, where columns represent regulators and rows regulated genes. The matrix informs the model of network connections known to be present/absent. For each position use either 0 (no regulation, fix off), 1 (known regulatory interaction, fix on) or NaN (no information, do not fix).

**Value**

For each chain run, a folder (chain1, chain2, ...) will be created and the output of the MCMC run will be placed there. The files will be Gamma_mcmc (the indicator variables of Gibbs variable selection), Lambda_mcmc (the precision of each regression), Mu_mcmc (the intercept of each regression), Rho_mcmc (the network connectivity parameter), Tau_mcmc (the "smoothness parameter"), all_f (posterior mean of all functions), all_f_sqr (posterior mean of the square of all functions) and Full_F_sqr (posterior mean of the square of the sum of all functions, for each regression). For the files all_f and all_f_sqr functions are placed in column-wise order. The file is filled by placing all interactions for each regression one after another.
References


See Also

mcmc.defaultParams_nonLinear, analyse.output.

Examples

# Synthetic data
data(Athaliana_ODE)
# Reduced data set to 3 genes 20 TP for faster run
Athaliana_ODE.reduced <- Athaliana_ODE[c(1,3,5),1:20]
# Folder where raw runs will be kept and later analysed
output.folder <- paste(tempdir(), "/ExampleNonLinearNet", sep = "")
# Run network inference and place raw results in output.folder
NonLinearNet(output.folder, Athaliana_ODE.reduced)
# Analyse raw results, place analysis plots and files in output.folder
analyse.output(output.folder, Athaliana_ODE.reduced)

plotPriors

Plot prior using parameter vector

Description

Plot appropriate priors using parameters from vector

Usage

plotPriors(parameter.vec)

Arguments

parameter.vec MCMC parameter vector of the type generated by e.g. mcmc.defaultParams_Linear

Details

This function takes the parameter vector that will be used for network inference function and plots the priors associated with the parameters given.

References


read.chain

See Also

mcmc.defaultParams_gauss, mcmc.defaultParams_Linear, mcmc.defaultParams_nonLinear, mcmc.defaultParams_student.

Examples

# Get default parameters
nonLinearNet.params <- mcmc.defaultParams_nonLinear()

# Change run length
nonLinearNet.params[1] <- 150000

# Change prior on smoothness parameter
nonLinearNet.params[6] <- 30000 # Change truncation
nonLinearNet.params[12] <- 3 # Concentrate more mass close to linear region

# Plot to check changes
plotPriors(nonLinearNet.params)

read.chain

Description

Read MCMC chains for further analysis.

Usage

read.chain(output.folder, chainNumber)

Arguments

  output.folder  Name of folder (including path) where chains are kept
  chainNumber    Which of the chains will be read

Details

Read chains produced by NonLinearNet, LinearNet, ReplicatesNet_student and ReplicatesNet_gauss for further analysis.

Value

Returns a list of vectors/matrices with the value of the variables at each MCMC iteration.

References

ReplicatesNet_gauss

Dynamic Bayesian Network Inference Using Linear Interactions and Gaussian Experimental Noise

Description

Run Bayesian inference of linear interaction network on data with replicates. The replicates are assumed to follow a Gaussian distribution. The function generates MCMC chains that can later be analysed.

Usage

```
ReplicatesNet_gauss(resultsFolder, timeSeries, numReps,
ParamVec = NULL, chains = 2, user.seeds = NULL,
Regulators = NULL, fixMe = NULL)
```

Arguments

- **resultsFolder**: Name of output folder. The folder will be created and the output of the run will be placed there.
- **timeSeries**: Data matrix containing gene expression time series. Where genes will be placed in rows and time points in columns. Each times series must be placed one after another, so that the first n columns correspond to time series replicate one, the next n columns to time series replicate two, etc. Gene names may be included as row names.
- **numReps**: Number of replicate time series included in timeSeries matrix.
- **ParamVec**: A parameter vector created using "mcmc.defaultParams_gauss". If none is given, default parameters will be used. The vector contains parameters associated to the priors as well as MCMC run length. (See mcmc.defaultParams_gauss)
- **chains**: Number of MCMC chains to run.

Examples

```
# Run inference using one chain
# Load A. thaliana circadian clock ODE generated data
data(Athaliana_ODE)
# Folder where raw runs will be kept and analysed
output.folder <- paste(tempdir(), "/Example_LinearNet", sep="")
# Run network inference, place raw results in output.folder
# Run just one chain for example purpose
LinearNet(output.folder, Athaliana_ODE, chains = 1)

# Read chain
chain1 <- read.chain(output.folder, 1)
```
ReplicatesNet_gauss

user.seeds An optional vector with seeds to use for MCMC chains.
Regulators An optional vector with the indices of which genes are regulators. If provided, all non-regulator genes will not be allowed to regulate.
fixMe An optional matrix of size genes x genes, where columns represent regulators and rows regulated genes. The matrix informs the model of network connections known to be present/absent. For each position use either 0 (no regulation, fix off), 1 (known regulatory interaction, fix on) or NaN (no information, do not fix).

Details

The order in which the replicates are placed do not affect the output. In other words swapping timepoint 2 replicate 1 and timepoint 2 replicate 2 makes no difference. For the cases where a measurement is not available, an NaN may be used.

Value

For each chain run, a folder (chain1, chain2, ...) will be created and the output of the MCMC run will be placed there. The files will be B_mcmc (the coefficients of the linear regression), Gamma_mcmc (the indicator variables of Gibbs variable selection), Lambda_mcmc (the precision of each regression), Mu_mcmc (the intercept of each regression), Rho_mcmc (the network connectivity parameter), DataMean_standarised (a times series with the mean of the replicates), Lambda_exp_mcmc (the precision of the replicate noise) and Y_mean (the mean posterior value of the inferred "true mRNA").

References


See Also

mcmc.defaultParams_gauss, analyse.output.

Examples

# Synthetic data
data(Athaliana_ODE_4NoiseReps)
# Folder where raw runs will be kept and later analysed
output.folder <- paste(tempdir(), "/Example_ReplicatesNet_Gauss",sep="")
# Run network inference and place raw results in output.folder
ReplicatesNet_gauss(output.folder, Athaliana_ODE_4NoiseReps, numReps = 4)
# Analyse raw results, place analysis plots and files in output.folder
analyse.output(output.folder)
ReplicatesNet_student  Dynamic Bayesian Network Inference Using Linear Interactions and Student Experimental Noise

Description

Run Bayesian inference of linear interaction network on data with replicates. The replicates are assumed to follow a Student distribution. The function generates MCMC chains that can later be analysed.

Usage

```r
ReplicatesNet_student( resultsFolder, timeSeries, numReps, ParamVec = NULL, chains = 2, user.seeds = NULL, Regulators = NULL, fixMe = NULL)
```

Arguments

- `resultsFolder` Name of output folder. The folder will be created and the output of the run will be placed there.
- `timeSeries` Data matrix containing gene expression time series. Where genes will be placed in rows and time points in columns. Each times series must be placed one after another, so that the first n columns correspond to time series replicate one, the next n columns to time series replicate two, etc. Gene names may be included as row names.
- `numReps` Number of replicate time series included in timeSeries matrix.
- `ParamVec` A parameter vector created using "mcmc.defaultParams_student". If none is given, default parameters will be used. The vector contains parameters associated to the priors as well as MCMC run length. (See mcmc.defaultParams_student)
- `chains` Number of MCMC chains to run.
- `user.seeds` An optional vector with seeds to use for MCMC chains.
- `Regulators` An optional vector with the indices of which genes are regulators. If provided, all non-regulator genes will not be allowed to regulate.
- `fixMe` An optional matrix of size genes x genes, where columns represent regulators and rows regulated genes. The matrix informs the model of network connections known to be present/absent. For each position use either 0 (no regulation, fix off), 1 (known regulatory interaction, fix on) or NaN (no information, do not fix).

Details

The order in which the replicates are placed do not affect the output. In other words swapping timepoint 2 replicate 1 and timepoint 2 replicate 2 makes no difference. For the cases where a measurement is not available, an NaN may be used.
For each chain run, a folder (chain1, chain2, ...) will be created and the output of the MCMC run will be placed there. The files will be B_mcmc (the coefficients of the linear regression), Gamma_mcmc (the indicator variables of Gibbs variable selection), Lambda_mcmc (the precision of each regression), Mu_mcmc (the intercept of each regression), Rho_mcmc (the network connectivity parameter), DataMean_standarised (a times series with the mean of the replicates), Lambda_exp_mcmc (the precision of the replicate noise), Y_mean (the mean posterior value of the inferred “true mRNA”), DegFreedom_mcmc (degrees of freedom of the student distributed replicate noise) and acceptanceRatio (acceptance ratio of the Metropolis-Hastings degrees of freedom update).

References


See Also

mcmc.defaultParams_student, analyse.output.

Examples

# Synthetic data
data(Athaliana_ODE_4NoiseReps)
# Folder where raw runs will be kept and later analysed
output.folder <- paste(tempdir(), "/Example_ReplicatesNet_Student",sep="")
# Run network inference and place raw results in output.folder
ReplicatesNet_student(output.folder, Athaliana_ODE_4NoiseReps, numReps = 4)
# Analyse raw results, place analysis plots and files in output.folder
analyse.output(output.folder)
Index

*Topic **AnalyseOutput**
  analyse.output, 2

*Topic **LinearNet**
  LinearNet, 4
  mcmc.defaultParams_Linear, 7

*Topic **NonLinearNet**
  mcmc.defaultParams_nonLinear, 8
  NonLinearNet, 11

*Topic **PlotPriors**
  plotPriors, 12

*Topic **ReadChains**
  read.chain, 13

*Topic **ReplicatesNet_gauss**
  mcmc.defaultParams_gauss, 6

*Topic **ReplicatesNet_student**
  mcmc.defaultParams_student, 9

*Topic **ReplicatesNet**
  ReplicatesNet_gauss, 14
  ReplicatesNet_student, 16

*Topic **datasets**
  Athaliana_ODE, 3
  Athaliana_ODE_4NoiseReps, 4

  analyse.output, 2, 5, 12, 15, 17
  Athaliana_ODE, 3
  Athaliana_ODE_4NoiseReps, 4

  LinearNet, 3, 4, 8, 14

  mcmc.defaultParams_gauss, 6, 13, 15
  mcmc.defaultParams_Linear, 5, 7, 13
  mcmc.defaultParams_nonLinear, 8, 12, 13
  mcmc.defaultParams_student, 9, 13, 17

  NonLinearNet, 3, 9, 11, 14

  plotPriors, 6, 8–10, 12

  read.chain, 13
  ReplicatesNet_gauss, 3, 6, 14, 14
  ReplicatesNet_student, 3, 10, 14, 16