Package 'multdyn'

August 24, 2017

Version 1.6

Date 2017-08-24

Title Multiregression Dynamic Models

Author Simon Schwab <schw4b@gmail.com>, Ruth Harbord <r.harbord@warwick.ac.uk>, Lilia Costa <liliacosta@ufba.br>, Thomas Nichols <t.e.nichols@warwick.ac.uk>

Maintainer Simon Schwab <schw4b@gmail.com>

Depends R (>= 3.2.0)

Imports Rcpp (>= 0.11.0), data.table (>= 1.10.0), reshape2 (>= 1.4.2), ggplot2 (>= 2.2.1)

LinkingTo Rcpp, RcppArmadillo

Suggests testthat

Description Multiregression Dynamic Models for directed dynamic functional brain network analysis.

License GPL-3

URL https://github.com/schw4b/multdyn

BugReports https://github.com/schw4b/multdyn/issues

RoxygenNote 6.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2017-08-24 12:42:46 UTC

R topics documented:

`binom.nettest` *Performes a binomial test with FDR correction for network edge occurrence.*

Description

Performes a binomial test with FDR correction for network edge occurrence.

Usage

```
binom.nettest(adj, alter = "two.sided", fdr = 0.05)
```

Arguments

`adj`	adjacency matrix, nodes x nodes x subj, or nodes x nodes x runs x subj.
`alter`	type of binomial test, "two.sided" (default), "less", or "greater"
`fdr`	false discovery rate (FDR) control, default is 0.05.

Value

store list with results.

`center` *Mean centers timeseries in a 2D array timeseries x nodes, i.e. each timeseries of each node has mean of zero.*

Description

Mean centers timeseries in a 2D array timeseries x nodes, i.e. each timeseries of each node has mean of zero.

Usage

```
center(X)
```

Arguments

`X`	2D array with dimensions timeseries x nodes.

Value

M 2D array.

`corTs` *Correlation of time series.*

Description

Correlation of time series.

Usage

```
corTs(ts)
```

Arguments

`ts`	a 3D time series time series x nodes x subjects.

Value

M correlation matrix.

`dlm.lpl` *Calculate the log predictive likelihood for a specified set of parents and a fixed delta.*

Description

Calculate the log predictive likelihood for a specified set of parents and a fixed delta.

Usage

```
dlm.lpl(Yt, Ft, delta, priors = priors.spec())
```

Arguments

`Yt`	the vector of observed time series, length `T`.
`Ft`	the matrix of covariates, dim = number of thetas (`p`) x number of time points (`T`), usually a row of 1s to represent an intercept and the time series of the parent nodes.
`delta`	discount factor (scalar).
`priors`	list with prior hyperparameters.

Value

`mt`	the vector or matrix of the posterior mean (location parameter), dim = `p x T`.
`Ct`	and `CSt` the posterior scale matrix `C_{t}` is `C_{t} = C*_{t} x S_{t}`, with dim = `p x p x T`, where `S_{t}` is a point estimate for the observation variance `phi^{-1}`
`Rt`	and `RSt` the prior scale matrix `R_{t}` is `R_{t} = R*_{t} x S_{t-1}`, with dim = `p x p x T`, where `S_{t-1}` is a point estimate for the observation variance `phi^{-1}` at the previous time point.
`nt`	and `dt` the vectors of the updated hyperparameters for the precision `phi` with length `T`.
`S`	the vector of the point estimate for the observation variance `phi^{-1}` with length `T`.
`ft`	the vector of the one-step forecast location parameter with length `T`.
`Qt`	the vector of the one-step forecast scale parameter with length `T`.
`ets`	the vector of the standardised forecast residuals with length `T`, defined as `(Y_{t} - f_{t}) / sqrt (Q_{t})`.
`lpl`	the vector of the Log Predictive Likelihood with length `T`.

References

West, M. & Harrison, J., 1997. Bayesian Forecasting and Dynamic Models. Springer New York.

`dlmLplCpp` *C++ implementation of the dlm.lpl*

Description

C++ implementation of the dlm.lpl

Usage

```
dlmLplCpp(Yt_, Ft_, delta, m0_, CS0_, n0, d0)
```

Arguments

`Yt_`	the vector of observed time series
`Ft_`	the matrix of covariates
`delta`	discount factor
`m0_`	the value of the prior mean
`CS0_`	controls the scaling of the prior variance
`n0`	prior hypermarameter
`d0`	prior hypermarameter

`exhaustive.search` *A function for an exhaustive search, calculates the optimum value of the discount factor.*

Description

A function for an exhaustive search, calculates the optimum value of the discount factor.

Usage

```
exhaustive.search(Data, node, nbf = 15, delta = seq(0.5, 1, 0.01),
  cpp = TRUE, priors = priors.spec())
```

Arguments

`Data`	Dataset with dimension number of time points T x Number of nodes Nn.
`node`	The node to find parents for.
`nbf`	Log Predictive Likelihood will sum from (and including) this time point.
`delta`	a vector of potential values for the discount factor.
`cpp`	boolean true (default): fast C++ implementation, false: native R code.
`priors`	list with prior hyperparameters.

Value

model.store a matrix with the model, LPL and chosen discount factor for all possible models. runtime an estimate of the run time of the function, using proc.time().

`getAdjacency` *Get adjacency and associated likelihoods (LPL) and disount factros (df) of winning models.*

Description

Get adjacency and associated likelihoods (LPL) and disount factros (df) of winning models.

Usage

```
getAdjacency(winner, nodes)
```

Arguments

`winner,`	2D matrix.
`nodes`	number of nodes.

Value

adj, 2D adjacency matrix.

`getModel` *Get specific parent model from all models.*

Description

Get specific parent model from all models.

Usage

```
getModel(models, parents)
```

Arguments

`models`	a 2D model matrix.
`parents`	a vector with parent nodes.

Value

mod specific parent model.

`getThreshAdj` *Get thresholded adjacency network.*

Description

Get thresholded adjacency network.

Usage

```
getThreshAdj(adj, models, winner, bf = 20)
```

Arguments

`adj`	list with network adjacency from getAdjacency().
`models`	matrix 3D with full model estimates.
`winner`	matrix 2D with winning models.
`bf`	bayes factor for network thresholding.

Value

thr list with thresholded network adjacency.

`getWinner` *Get winner network by maximazing log predictive likelihood (LPL) from a set of models.*

Description

Get winner network by maximazing log predictive likelihood (LPL) from a set of models.

Usage

```
getWinner(models, nodes)
```

Arguments

`models`	2D matrix, or 3D models x node.
`nodes`	number of nodes.

Value

winner array with highest scored model(s).

gplotMat *Plots network as adjacency matrix.*

Description

Plots network as adjacency matrix.

Usage

```
gplotMat(adj, title = NULL, colMapLabel = NULL, hasColMap = NULL,
  lim = c(0, 1), gradient = c("white", "orange", "red"),
  nodeLabels = waiver(), axisTextSize = 12, xAngle = 0,
  titleTextSize = 12)
```

Arguments

adj	2D adjacency matrix.
title	title.
colMapLabel	label for colormap.
hasColMap	FALSE turns off color map, default is NULL (on).
lim	vector with min and max value for color scaling.
gradient	gradient colors.
nodeLabels	node labels.
axisTextSize	text size of the y and x tick labels.
xAngle	orientation of the x tick labels.
titleTextSize	text size of the title.

mdm.group *A group is a list containing restructured data from subejcts for easier group analysis.*

Description

A group is a list containing restructured data from subejcts for easier group analysis.

Usage

```
mdm.group(subj)
```

Arguments

subj	a list of subjects.

Value

group a list.

model.generator *A function to generate all the possible models.*

Description

A function to generate all the possible models.

Usage

```
model.generator(Nn, node)
```

Arguments

Nn	number of nodes; the number of columns of the dataset can be used.
node	The node to find parents for.

Value

output.model = a matrix with dimensions (Nn-1) x number of models, where number of models = 2^(Nn-1).

myts *Network simulation data.*

Description

Simulation 22 5 node net from Smith et al. 2011 (only first subject).

node *Runs exhaustive search on a single node and saves results in txt file.*

Description

Runs exhaustive search on a single node and saves results in txt file.

Usage

```
node(X, n, id = NULL, nbf = 15, delta = seq(0.5, 1, 0.01), cpp = TRUE,
  priors = priors.spec(), path = getwd())
```

Arguments

X	array with dimensions timeseries x nodes.
n	node number.
id	subject ID. If set, results are saved to a txt file.
nbf	Log Predictive Likelihood will sum from (and including) this time point.
delta	a vector of potential values for the discount factor.#'
cpp	boolean true (default): fast C++ implementation, false: native R code.
priors	list with prior hyperparameters.
path	a path where results are written.

Value

store list with results.

---

patel *Patel.*

Description

Patel.

Usage

```
patel(X, lower = 0.1, upper = 0.9, bin = 0.75, TK = 0, TT = 0)
```

Arguments

X	time x node 2D matrix.
lower	percentile cuttoff.
upper	percentile cuttoff for 0-1 scaling.
bin	threshold for conversion to binary values.
TK	significance threshold for connection strength kappa.
TT	significance threshold for direction tau.

Value

PT list with strengths kappa, direction tau, and net structure.

patel.group — *A group is a list containing restructured data from subejcts for easier group analysis.*

Description

A group is a list containing restructured data from subejcts for easier group analysis.

Usage

```
patel.group(subj)
```

Arguments

subj	a list of subjects.

Value

group a list.

perf — *Performance of estimates, such as sensitivity, specificity, and more.*

Description

Performance of estimates, such as sensitivity, specificity, and more.

Usage

```
perf(x, xtrue)
```

Arguments

x	estimated binary network matrix.
xtrue,	true binary network matrix.

Value

perf vector.

`perm.test` *Permutation test for Patel's kappa. Creates a distribution of values kappa under the null hypothesis.*

Description

Permutation test for Patel's kappa. Creates a distribution of values kappa under the null hypothesis.

Usage

```
perm.test(X, alpha = 0.05)
```

Arguments

`X`	time x node x subjects 3D matrix.
`alpha`	sign. level

Value

stat lower and upper significance thresholds.

`priors.spec` *Specify the priors. Without inputs, defaults will be used.*

Description

Specify the priors. Without inputs, defaults will be used.

Usage

```
priors.spec(m0 = 0, CS0 = 3, n0 = 0.001, d0 = 0.001)
```

Arguments

`m0`	the value of the prior mean at time `t=0`, scalar (assumed to be the same for all nodes). The default is zero.
`CS0`	controls the scaling of the prior variance matrix `C*_{0}` at time `t=0`. The default is 3, giving a non-informative prior for `C*_{0}, 3 x (p x p)` identity matrix. `p` is the number of thetas.
`n0`	prior hyperparameter of precision `phi ~ G(n_{0}/2; d_{0}/2)`. The default is a non-informative prior, with `n0 = d0 = 0.001`. `n0` has to be higher than 0.
`d0`	prior hyperparameter of precision `phi ~ G(n_{0}/2; d_{0}/2)`. The default is a non-informative prior, with `n0 = d0 = 0.001`.

Details

At time t=0, (theta_{0} | D_{0}, phi) ~ N(m_{0},C*_{0} x phi^{-1}), where D_{0} denotes the set of initial information.

Value

priors a list with the prior hyperparameters. Relevant to dlm.lpl, exhaustive.search, node, subject.

References

West, M. & Harrison, J., 1997. Bayesian Forecasting and Dynamic Models. Springer New York.

---

read.subject *Reads single subject's network from txt files.*

Description

Reads single subject's network from txt files.

Usage

```
read.subject(path, id, nodes, bf = 20)
```

Arguments

path	path.
id	identifier to select all subjects' nodes, e.g. pattern containing subject ID and session number.
nodes	number of nodes.
bf	bayes factor for network thresholding.

Value

store list with results.

`reshapeTs` *Reshapes a 2D concatenated time series into 3D according to no. of subjects and volumes.*

Description

Reshapes a 2D concatenated time series into 3D according to no. of subjects and volumes.

Usage

```
reshapeTs(ts, N, V)
```

Arguments

`ts`	a 2D time series volumes x nodes.
`N`	No. of subjects.
`V`	No. of volumes.

Value

M 3D matrix, time series x nodes x subjects.

`rmdiag` *Removes diagnoal from matrix with NAs.*

Description

Removes diagnoal from matrix with NAs.

Usage

```
rmdiag(M)
```

Arguments

`M`	Matrix

Value

matrix with diagnoal of NAs.

rmna *Removes NAs from matrix.*

Description

Removes NAs from matrix.

Usage

```
rmna(M)
```

Arguments

M	Matrix

Value

matrix with NAs removed.

scaleTs *Scaling data. Zero centers and scales the nodes (SD=1).*

Description

Scaling data. Zero centers and scales the nodes (SD=1).

Usage

```
scaleTs(X)
```

Arguments

X	time x node 2D matrix, or 3D with subjects as the 3rd dimension.

Value

S centered and scaled matrix.

`stepwise.backward` *Stepise backward non-exhaustive greedy search, calculates the optimum value of the discount factor.*

Description

Stepise backward non-exhaustive greedy search, calculates the optimum value of the discount factor.

Usage

```
stepwise.backward(Data, node, nbf = 15, delta = seq(0.5, 1, 0.01),
  max.break = TRUE, priors = priors.spec())
```

Arguments

`Data`	Dataset with dimension number of time points `T` x number of nodes `Nn`.
`node`	The node to find parents for.
`nbf`	The Log Predictive Likelihood will sum from (and including) this time point.
`delta`	A vector of values for the discount factor.
`max.break`	If `TRUE`, the code will break if adding / removing parents does not improve the LPL. If `FALSE`, the code will continue to the zero parent / all parent model. Default is `TRUE`.
`priors`	List with prior hyperparameters.

Value

model.store The parents, LPL and chosen discount factor for the subset of models scored using this method.

`stepwise.combine` *Stepise combine: combines the stepwise forward and the stepwise backward model.*

Description

Stepise combine: combines the stepwise forward and the stepwise backward model.

Usage

```
stepwise.combine(forward_matrix, backward_matrix)
```

Arguments

forward_matrix	The winning sets of parents using a Forward Selection model search. A matrix with dimension Nn+2 x Nn, rows 1:Nn are the parents (ones and zeros), rows (Nn+1):(Nn+2) are the LPL and discount factor. forward matrix.
backward_matrix	backward_matrixThe winning sets of parents using a Backward Elimination model search. A matrix with dimension Nn+2 x Nn, rows 1:Nn are the parents (ones and zeros), rows (Nn+1):(Nn+2) are the LPL and discount factor.

Value

stepwise_combine_matrix The adjacency network, LPLs and discount factors when the Forward Selection and Backward Elimination model searches are combined.

---

stepwise.forward *Stepise forward non-exhaustive greedy search, calculates the optimum value of the discount factor.*

Description

Stepise forward non-exhaustive greedy search, calculates the optimum value of the discount factor.

Usage

```
stepwise.forward(Data, node, nbf = 15, delta = seq(0.5, 1, 0.01),
  max.break = TRUE, priors = priors.spec())
```

Arguments

Data	Dataset with dimension number of time points T x number of nodes Nn.
node	The node to find parents for.
nbf	The Log Predictive Likelihood will sum from (and including) this time point.
delta	A vector of values for the discount factor.
max.break	If TRUE, the code will break if adding / removing parents does not improve the LPL. If FALSE, the code will continue to the zero parent / all parent model. Default is TRUE.
priors	List with prior hyperparameters.

Value

model.store The parents, LPL and chosen discount factor for the subset of models scored using this method.

subject *Estimate subject's full network: runs exhaustive search on very node.*

Description

Estimate subject's full network: runs exhaustive search on very node.

Usage

```
subject(X, id = NULL, nbf = 15, delta = seq(0.5, 1, 0.01), cpp = TRUE,
  bf = 20, priors = priors.spec(), path = getwd())
```

Arguments

X	array with dimensions timeseries x nodes.
id	subject ID. If set, results are saved to a txt file.
nbf	Log Predictive Likelihood will sum from (and including) this time point.
delta	a vector of potential values for the discount factor.
cpp	boolean true (default): fast C++ implementation, false: native R code.
bf	bayes factor for network thresholding.
priors	list with prior hyperparameters.
path	a path where results are written.

Value

store list with results.

utestdata *Results from v.1.0 for unit tests.*

Description

Some LPL values (n2 parent of n1 Simulation 22) to test against.

Index