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 .................................................. 2
center ............................................................ 3
corTs ............................................................. 3
dlm.lpl ........................................................... 4
dlmLplCpp ......................................................... 5
exhaustive.search ............................................... 5
getAdjacency ..................................................... 6
getModel .......................................................... 6
Performes a binomial test with FDR correction for network edge occurrence.

**Usage**

```r
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

Description
Correlation of time series.

Usage
corTs(ts)

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

Value
M correlation matrix.
Calculate the log predictive likelihood for a specified set of parents and a fixed delta.

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: the posterior scale matrix 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: the prior scale matrix 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: 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
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 hyperparameter
d0 prior hyperparameter

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.
**getAdjacency**

Get adjacency and associated likelihoods (LPL) and discount factors (df) of winning models.

**Description**

Get adjacency and associated likelihoods (LPL) and discount factors (df) of winning models.

**Usage**

```r
getAdjacency(winnerL, nodes)
```

**Arguments**

- `winnerL`: a 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**

```r
getModel(modelsL, 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.

gewinner

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

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

Usage
gewinner(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

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

mdm.group

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

Description

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

Usage

mdm.group(subj)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>subj</td>
<td>a list of subjects.</td>
</tr>
</tbody>
</table>

Value

group a list.
**model.generator**  

A function to generate all the possible models.

---

**Description**

A function to generate all the possible models.

**Usage**

```r
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 \((Nn-1) \times \text{number of models}\), 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**

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

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

Value

store list with results.

Description

Patel.

Usage

\texttt{patel(xL lower = 0NQL upper = 0NYL bin = 0NWUL tk = 0L tt = 0)}

Arguments

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

Value

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

<table>
<thead>
<tr>
<th>patel.group</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A group is a list containing restructured data from subjects for easier group analysis.</td>
<td></td>
</tr>
</tbody>
</table>

**Usage**

patel.group(subj)

**Arguments**

subj 

a list of subjects.

**Value**

group a list.

---

**perf**

<table>
<thead>
<tr>
<th>perf</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance of estimates, such as sensitivity, specificity, and more.</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

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

**Usage**

perf(x, xtrue)

**Arguments**

x 

estimated binary network matrix.

xtrue, 

ture 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*_{t=0} at time t=0. The default is 3, giving a non-informative prior for C*_{t=0}, 3 x (p x p) identity matrix. p is the number of thetas.

n0  
prior hyperparameter of precision \( \phi \sim G(n_0/2; d_0/2) \). The default is a non-informative prior, with \( n0 = d0 = 0.001 \). \( n0 \) has to be higher than 0.

d0  
prior hyperparameter of precision \( \phi \sim G(n_0/2; d_0/2) \). The default is a non-informative prior, with \( n0 = d0 = 0.001 \).
Details

At time $t=0$, $(\theta_0 \mid D_0, \phi) \sim N(m_0\{0\}, C_\phi\{0\} \times \phi^{\{-1\}})$, 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


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.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reshapeTs</td>
<td>Reshapes a 2D concatenated time series into 3D according to no. of subjects and volumes.</td>
</tr>
</tbody>
</table>

**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.

---

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rmdiag</td>
<td>Removes diagonal from matrix with NAs.</td>
</tr>
</tbody>
</table>

**Description**

Removes diagonal from matrix with NAs.

**Usage**

```
rmdiag(M)
```

**Arguments**

- `M`: Matrix

**Value**

matrix with diagonal of NAs.
**rmna**

*Removes NAs from matrix.*

**Description**

Removes NAs from matrix.

**Usage**

\[ \text{rmna}(M) \]

**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**

\[ \text{scaleTs}(X) \]

**Arguments**

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

**Value**

S centered and scaled matrix.
**Description**

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

**Usage**

```r
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.

**Description**

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

**Usage**

```r
stepwise.combine(forward_matrix, backward_matrix)
```
**Arguments**

- **forward_matrix**
  The winning sets of parents using a Forward Selection model search. A matrix with dimension \( Nn+2 \times Nn \), rows 1:Nn are the parents (ones and zeros), rows (Nn+1):(Nn+2) are the LPL and discount factor.

- **backward_matrix**
  The winning sets of parents using a Backward Elimination model search. A matrix with dimension \( Nn+2 \times Nn \), 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**

```r
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 \( T \times \) 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

binom.nettest, 2

center, 3
corTs, 3

dlm.lpl, 4, 13
dlmLplCpp, 5

exhaustive.search, 5, 13

getAdjacency, 6
getModel, 6
getThreshAdj, 7
getWinner, 7
gplotMat, 8

mdm.group, 8
model.generator, 9
myts, 9

node, 9, 13

patel, 10
patel.group, 11
perf, 11
perm.test, 12
priors.spec, 12

read.subject, 13
reshapeTs, 14
rmdiag, 14
rmna, 15

scaleTs, 15
stepwise.backward, 16
stepwise.combine, 16
stepwise.forward, 17
subject, 13, 18

utestdata, 18