Package 'latChanNet'

October 1, 2019

Type Package

Title Latent Channel Networks

Version 1.0

Date 2019-05-09

Author Clifford Anderson-Bergman

Maintainer Clifford Anderson-Bergman <pistacliffcho@gmail.com>

Description Analysis of undirected networks via Latent Channel Networks

License GPL (>= 2)

bioViews

Imports Rcpp (>= 1.0.1), RcppParallel (>= 4.4.3)

LinkingTo Rcpp, RcppParallel

SystemRequirements GNU make

RoxygenNote 6.1.99.9001

R topics documented:

latChanNet-package *A short title line describing what the package does*

Description

A more detailed description of what the package does. A length of about one to five lines is recommended.

Details

This section should provide a more detailed overview of how to use the package, including the most important functions.

Author(s)

Your Name, email optional.

Maintainer: Your Name <your@email.com>

References

This optional section can contain literature or other references for background information.

See Also

Optional links to other man pages

Examples

```
  ## Not run:
     ## Optional simple examples of the most important functions
     ## These can be in \dontrun{} and \donttest{} blocks.

## End(Not run)
```

computeExpConnects *Compute expected connections for node*

Description

Computes the estimated expected connections through each latent channel

Usage

```
computeExpConnects(i, lcn_mod)
```

Arguments

i	Index of node
lcn_mod	LCN model

computeTheta *Expected Channel Usage*

Description

Compute expected channel usage between two connected nodes.

Usage

```
computeTheta(i, j, lcn_mod)
```

Arguments

i	Index of first node
j	Index of second node
lcn_mod	LCN model

Details

Computes the expected channel usage between two nodes **conditional on the two nodes sharing an edge.**

get_auc *Get auc for a model from edges/notEdges list*

Description

Get auc for a model from edges/notEdges list

Usage

```
get_auc(mod, edges, notEdges)
```

get_both_auc *Get both in sample and out of sample AUC*

Description

Get both in sample and out of sample AUC

Usage

```
get_both_auc(mod, out_edges, out_notEdges, in_edges, in_notEdges)
```

heatmapLCN *Build heatmap from model*

Description

Build heatmap from model

Usage

```
heatmapLCN(
  mod,
  grp,
  minGrpSize = NULL,
  prob_cols = c("black", "grey", "blue"),
  greater_col = "red",
  plotChannelNumber = T,
  xlab = " ",
  ylab = " ",
  sortColumns = T,
  name = "",
  ...
)
```

Arguments

mod	LCN or BKN model
grp	Vector of group categories for each node
minGrpSize	Minimum size of group in both. Smaller groups put in "other"
prob_cols	Colors for color gradient of probability range
greater_col	Color for color gradient beyond 1
xlab	X-axis label
ylab	Y-asix label
...	Additional arguments passed to ComplexHeatmap::Heatmap
reorderChannels	Should Channels be reorder by dependency on grp?

makeLatentModel *Make Latent Structure model*

Description

Make a latent class model. Can be used for predicting unknown edge status and unknown metadata.

Usage

```
makeLatentModel(
  edgeList,
  nDims,
  model = "LCN",
  missingList = NULL,
  metadata = NULL
)
```

Arguments

edgeList	An matrix edgelist. Can be nx2 (both) or nx3 (BKN only)
model	Type of model to fit. Options are "LCN" or "BKN"
missingList	A nx2 matrix edgelist of edges for which the value is unknown
metadata	A data.frame with all factors representing metadata
nDim	Number of latent dimensions to use

Details

Fits either a Latent Channels Network (LCN), or the symmetric low-rank Poisson model of Ball, Karrer and Newman (BKN). The model assumes an undirected graph.

If edges are counts, use the BKN model. The data format for each row should (i,j, count), with i,j as integer IDs starting at 1.

If edges are binary, either a BKN or LCN model may be used, although an LCN model is somewhat more appropriate.

LCN model:

Clifford Anderson-Bergman, Phan Nguyen, and Jose Cadena Pico. "Latent Channel Networks", submitted 2019

BKN model:

Brian Ball, Brian Karrer, and Mark EJ Newman. "Efficient and principled method for detecting communities in networks." Physical Review E 84.3 (2011): 036103.

Examples

```
data(email_data)
# Building model with metadata
df = data.frame(dpt = email_data$nodeDpt)
model = makeLatentModel(email_data$edgeList,
                        10,
                        metadata = df)
# Fitting model
model$fit()

# Predicting a two edge probabilities
predict(model, )
```

predict.LatClass *Predictions from LatClass objects*

Description

Predict edge probabilities and categorical metadata

Usage

```
## S3 method for class 'LatClass'
predict(mod, i, j, type = "pairs")
```

Arguments

mod	LatClass model
i	node index
j	Either an node index or metadata colname name
type	Should node pairs or cross of all combinations be predicted

Examples

```
data(email_data)
df = data.frame(dpt = email_data$nodeDpt)
# Grouping dpt for brevity
df$dpt[df$dpt > 5] = "other"
# Building model and fitting
mod = makeLatentModel(email_data$edgeList,
                      nDims = 10,
                      metadata = df)
mod$fit(fast_em = T)

# Predicting edge pairs
predict(mod, i = 1:3, j = 1:3)

# Predicting all combinations of i and j
predict(mod, i = 1:3, j = 1:3, type = "cross")

# Predicting meta data
predict(mod, i = 1:3, "dpt")
```

simLCN *Simulate Latent Channel Network*

Description

Simulate Latent Channel Network

Usage

```
simLCN(p_mat)
```

Arguments

p_mat	Matrix of channel usage probabilities

`unq_nondiag_flat` *Return only unique non-selfloop samples from flat_index*

Description

Return only unique non-selfloop samples from flat_index

Usage

```
unq_nondiag_flat(flat, max_n)
```

Index