Package ‘ess’

May 31, 2021

Title Efficient Stepwise Selection in Decomposable Models
Version 1.1.2
Description An implementation of the ESS algorithm following Amol Deshpande, Minos Garofalakis, Michael I Jordan (2013) <arXiv:1301.2267>. The ESS algorithm is used for model selection in decomposable graphical models.

URL https://github.com/mlindsk/ess
Depends R (>= 3.5.0)
License GPL-3
Encoding UTF-8
LazyData true
Imports Rcpp, igraph
LinkingTo Rcpp
RoxygenNote 7.1.1
Suggests tinytest

BugReports https://github.com/mlindsk/ess/issues
SystemRequirements C++11
NeedsCompilation yes
Author Mads Lindskou [aut, cre]
Maintainer Mads Lindskou <mads@math.aau.dk>
Repository CRAN
Date/Publication 2021-05-31 07:40:08 UTC

R topics documented:

ess-package .......................................................... 2
adj_lst .............................................................. 3
adj_mat ............................................................. 3
as_adj_lst ......................................................... 4
as_adj_mat ......................................................... 4
Description

The class of graphical models is a family of probability distributions for which conditional dependencies can be read off from a graph. If the graph is decomposable, the maximum likelihood estimates of the parameters in the model can be shown to be on exact form. This is what enables ESS to be fast and efficient for model selection in decomposable graphical models.

Author(s)

Maintainer: Mads Lindskou <mads@math.aau.dk>

See Also

Useful links:

- [https://github.com/mlindsk/ess](https://github.com/mlindsk/ess)
adj_lst

Description
Extracts the adjacency list of a gengraph

Usage
adj_lst(x)

## S3 method for class 'gengraph'
adj_lst(x)

Arguments
x gengraph

Value
An adjacency list

adj_mat

Description
Extracts the adjacency matrix of a gengraph object

Usage
adj_mat(x)

## S3 method for class 'gengraph'
adj_mat(x)

Arguments
x gengraph object

Value
An adjacency matrix
as_adj_lst

Converts an adjacency matrix to an adjacency list

**Description**

Converts an adjacency matrix to an adjacency list

**Usage**

```r
as_adj_lst(A)
```

**Arguments**

- `A` Adjacency matrix

---

as_adj_mat

Converts an adjacency list to an adjacency matrix

**Description**

Converts an adjacency list to an adjacency matrix

**Usage**

```r
as_adj_mat(adj)
```

**Arguments**

- `adj` Adjacency list

**Value**

An adjacency matrix

**Examples**

```r
adj <- list(a = c("b", "d"), b = c("a", "c", "d"), c = c("b", "d"), d = c("a", "c", "b"))
as_adj_mat(adj)
```
as_igraph

**Description**
Convert a gengraph object to an igraph object

**Usage**

```r
as_igraph(x)
```

```r
## S3 method for class 'gengraph'
as_igraph(x)
```

**Arguments**

- `x`: gengraph object

**Value**

An igraph object

---

components

**Description**
Finds the components of a graph

**Usage**

```r
components(adj)
```

**Arguments**

- `adj`: Adjacency list or gengraph object

**Value**

A list where the elements are the components of the graph
Dermatology Database

Description
This data set contains 358 observations (we have removed 8 with missing values). It contains 12 clinical attributes and 21 histopathological attributes. The age attribute has been discretized. The class variable "ES" has six levels; each describing a skin disease.

Usage
derma

Format
An object of class tbl_df (inherits from tbl, data.frame) with 358 rows and 35 columns.

References
Dermatology Data Set

dfs
Depth First Search

Description
Finds the elements in the component of root

Usage
dfs(adj, root)

Arguments
adj A named adjacency list of a decomposable graph
root The node from which the component should be found

Value
All nodes connected to root

Examples
x <- list(a = c("b", "d"), b = c("a", "d"), c = c("b", "a"),
          d = c("e", "f"), e = c("d", "f"), f = c("d", "e"))
dfs(x, "a")
Simulate observations from a decomposable graphical model

**Description**

Simulate observations from a decomposable graphical model

**Usage**

```r
dgm_sim_from_graph(g, lvls, nsim = 1000, cell_rate = 0.5)
```

**Arguments**

- `g` An adjacency list
- `lvls` Named list with levels of the discrete variables
- `nsim` Number of simulations
- `cell_rate` Control discrete cell probabilities

**Value**

This function returns a matrix of dimension where each row correspond to a simulated observation from a DGM represented by `g`.

**Examples**

```r
g = list(
  A = c("B", "X", "Y"),
  B = c("A", "Y"),
  X = c("A", "Y"),
  Y = c("A", "X", "B")
)

lvls <- list(
  A = c("0", "1"),
  B = c("0", "1"),
  X = c("a", "b", "c"),
  Y = c("0", "1", "2")
)

dgm_sim_from_graph(g, lvls, nsim = 10)
```

#
entropy

**Joint Entropy**

**Description**
Calculates the joint entropy over discrete variables in df

**Usage**

```r
entropy(df, thres = 5, npc = new.env())
```

**Arguments**

- `df`  data.frame
- `thres` A threshold mechanism for choosing between two different ways of calculating the entropy. Can Speed up the procedure with the “correct” value.
- `npc` An environment. If supplied, the number of positive cells in the underlying pmf will be stored in the environment with the name value

**Value**
A number representing the entropy of the variables in df.

**Examples**

```r
entropy(derma[1:100, 1:3])
```

---

**fit_components**

*Fit a decomposable graphical model on each component*

**Description**
Structure learning in decomposable graphical models on several components

**Usage**

```r
fit_components(
    df, comp, type = "fwd", q = 0.5,
    trace = FALSE, thres = 5,
    wrap = TRUE
)
```
Arguments

- **df**: Character data.frame
- **comp**: A list with character vectors. Each element in the list is a component in the graph (using expert knowledge)
- **type**: Character ("fwd", "bwd", "tree" or "tfwd")
- **q**: Penalty term in the stopping criterion where 0 = AIC and 1 = BIC. Anything in between is referred to as qic
- **trace**: Logical indicating whether or not to trace the procedure
- **thres**: A threshold mechanism for choosing between two different ways of calculating the entropy.
- **wrap**: Logical specifying if the result of a run with type = "tree" should be converted to a "fwd" object

Value

An adjacency list object

See Also

- `fit_graph`, `adj_lst.gengraph`, `adj_mat.gengraph`, `walk.fwd`, `walk.bwd`, `gengraph`

---

**fit_graph**

*Fit a decomposable graphical model*

Description

A generic method for structure learning in decomposable graphical models

Usage

```r
fit_graph(
  df,
  type = "fwd",
  q = 0.5,
  trace = FALSE,
  sparse_qic = FALSE,
  thres = 5,
  wrap = TRUE
)
```
Arguments

df  Character data.frame

type  Character ("fwd", "bwd", "tree" or "tfwd")

q  Penalty term in the stopping criterion where 0 = AIC and 1 = BIC. Anything in between is referred to as qic

trace  Logical indicating whether or not to trace the procedure

sparse_qic  Logical. If nrow(df) is small, the tables tends to be sparse. In these cases the usual penalty term of AIC and BIC is often too restrictive. If sparse_qic is TRUE this penalty is computed according to a sparse criteria. The criteria resembles the usual penalty as nrow(df) grows.

thres  A threshold mechanism for choosing between two different ways of calculating the entropy.

wrap  logical specifying if the result of a run with type = "tree" should be converted to a "fwd" object

Details

The types are

- "fwd": forward selection
- "bwd": backward selection
- "tree": Chow-Liu tree (first order interactions only)
- "tfwd": A combination of "tree" and "fwd". This can speed up runtime considerably in high dimensions.

Using adj_lst on an object returned by fit_graph gives the adjacency list corresponding to the graph. Similarly one can use adj_mat to obtain an adjacency matrix. Applying the rip function on an adjacency list returns the cliques and separators of the graph.

Value

A gengraph object representing a decomposable graph.

References


See Also

adj_lst, adj_mat, as_igraph, gengraph
Examples

g <- fit_graph(derma)
print(g)
plot(g)

# Adjacency matrix and adjacency list
adjm <- adj_mat(g)
adjl <- adj_lst(g)

Description

A generic and extendable structure for decomposable graphical models

Usage

gengraph(df, type = "fwd", q = 0.5, sparse_qic = TRUE)

Arguments

df Character data.frame
type Character ("fwd", "bwd", "tree" or "tfwd")
q Penalty term in the stopping criterion where 0 = AIC and 1 = BIC. Anything in between is referred to as qic
sparse_qic Logical. If nrow(df) is small, the tables tends to be sparse. In these cases the usual penalty term of AIC and BIC is often too restrictive. If sparse_qic is TRUE this penalty is computed according to a sparse criteria. The criteria resembles the usual penalty as nrow(df) grows.

Value

A gengraph object with child class type used for model selection.

See Also

adj_lst.gengraph, adj_mat.gengraph, fit_graph, walk.fwd, walk.bwd

Examples

gengraph(derma, type = "fwd")
gengraph(derma, type = "bwd")
is_decomposable  A test for decomposability in undirected graphs

Description

This function returns TRUE if the graph is decomposable and FALSE otherwise.

Usage

is_decomposable(adj)

Arguments

adj          Adjacency list of an undirected graph

Value

Logical describing whether or not adj is decomposable

Examples

# 4-cycle:
adj <- list(a = c("b", "d"), b = c("a", "c"), c = c("b", "d"), d = c("a", "c"))
is_decomposable(adj)  # FALSE

# Two triangles:
adj2 <- list(a = c("b", "d"), b = c("a", "c", "d"), c = c("b", "d"), d = c("a", "c", "b"))
is_decomposable(adj2)  # TRUE

make_complete_graph  Make a complete graph

Description

A helper function to make an adjacency list corresponding to a complete graph

Usage

make_complete_graph(nodes)

Arguments

nodes          A character vector containing the nodes to be used in the graph

Value

An adjacency list of a complete graph
**make_null_graph**

**Examples**

```r
d <- derma[, 5:8]
cg <- make_complete_graph(colnames(d))
```

---

**Description**

A helper function to make an adjacency list corresponding to a null graph (no edges)

**Usage**

```r
make_null_graph(nodes)
```

**Arguments**

- `nodes` A character vector containing the nodes to be used in the graph

**Value**

An adjacency list the null graph with no edges

**Examples**

```r
d <- derma[, 5:8]
ng <- make_null_graph(colnames(d))
```

---

**mcs**

*Maximum Cardinality Search*

**Description**

Maximum Cardinality Search

**Usage**

```r
mcs(adj, check = TRUE)
```

**Arguments**

- `adj` A named adjacency list of a decomposable graph
- `check` Boolean: check if `adj` is decomposable

**Details**

If `adj` is not the adjacency list of a decomposable graph an error is raised
Value

A list with a perfect numbering of the nodes and a perfect sequence of sets

Examples

```r
x <- list(a = c("b", "d"), b = c("a", "c", "d"), c = c("b", "d"), d = c("a", "c", "b"))
mcs(x)
```

Description

A wrapper around igraphs plot method for gengraph objects

Usage

```r
## S3 method for class 'gengraph'
plot(x, vc = NULL, ...)
```

Arguments

- `x`: A gengraph object
- `vc`: Named character vector; the names are the vertices and the elements are the colors of the nodes
- `...`: Extra arguments. See the igraph package

Value

No return value, called for side effects

Examples

```r
d <- derma[, 10:25]
g <- fit_graph(d)
vs <- colnames(d)
vcol <- structure(vector("character", length(vs)), names = vs)
vcol[1:4] <- "lightsteelblue2"
vcol[5:7] <- "orange"
vcol[8:16] <- "pink"
plot(g, vcol)
```
print.gengraph

Description

A print method for gengraph objects

Usage

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

Arguments

x A gengraph object
...
Not used (for S3 compatibility)

print.tree

Description

A print method for tree objects

Usage

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

Arguments

x A tree object
...
Not used (for S3 compatibility)
**Rip**  

**Running Intersection Property**

**Description**

Given a decomposable graph, this function finds a perfect numbering on the vertices using maximum cardinality search, and hereafter returns a list with two elements: “C” - A RIP-ordering of the cliques and “S” - A RIP ordering of the separators.

**Usage**

```r
rip(adj, check = TRUE)
```

**Arguments**

- `adj`  
  A named adjacency list of a decomposable graph

- `check`  
  Boolean: check if adj is decomposable

**Value**

A list with cliques and separators of `adj`

**See Also**

- `mcs`
- `is_decomposable`

**Examples**

```r
x <- list(a = c("b", "d"), b = c("a", "c", "d"), c = c("b", "d"), d = c("a", "c", "b"))
y <- rip(x)
# Cliques:
y$C
# Separators:
y$S
```

---

**Subgraph**

**Description**

Construct a subgraph with a given set of nodes removed

**Usage**

```r
subgraph(x, g)
```
walk

Arguments

x  Character vector of nodes

args  Adjacency list (named) or a adjacency matrix with dimnames given as the nodes

Value

An adjacency list or adjacency matrix.

Examples

adj <- list(a = c("b", "d"), b = c("a", "c", "d"), c = c("b", "d"), d = c("a", "c", "b"))
d <- data.frame(a = "", b = "", c = "", d = "")  # Toy data so we can plot the graph
subgraph(c("c", "b"), adj)
subgraph(c("b", "d"), as_adj_mat(adj))

walk  Stepwise model selection

Description

Stepwise model selection in decomposable graphical models

Usage

walk(x, df, q, thres)

Arguments

x  fwd or bwd objects

df  data.frame

q  Penalty term in the stopping criterion (0 = AIC and 1 = BIC)

thres  A threshold mechanism for choosing between two different ways of calculating the entropy. Can Speed up the procedure with the "correct" value.

Details

A fwd (or bwd) object can be created using the gengraph constructor with type = "fwd".

Value

A fwd or bwd object with one additional edge than the input object.

See Also

fit_graph, walk.fwd, gengraph
Examples

d <- derma[, 10:25]
g <- gengraph(d, type = "fwd")
s <- walk(g, d)
print(s)
plot(s)
adj_lst(s)
adj_mat(s)

walk.bwd Stepwise backward selection

Description

Stepwise backward selection in decomposable graphical models

Usage

## S3 method for class 'bwd'
walk(x, df, q = 0.5, thres = 5)

Arguments

x gengraph
df data.frame
q Penalty term in the stopping criterion (0 = AIC and 1 = BIC)
thres A threshold mechanism for choosing between two different ways of calculating the entropy. Can Speed up the procedure with the "correct" value.

Details

A bwd object can be created using the gengraph constructor with type = "bwd"

Value

A bwd object; a subclass of gengraph) used for backward selection.

See Also

fit_graph, walk.fwd, gengraph
Examples

```r
d <- derma[, 10:25]
g <- gengraph(d, type = "bwd")
s <- walk(g, d)
print(s)
plot(s)
adj_lst(s)
adj_mat(s)
```

Description

Stepwise efficient forward selection in decomposable graphical models

Usage

```r
## S3 method for class 'fwd'
walk(x, df, q = 0.5, thres = 5)
```

Arguments

- `x`: A `fwd` object
- `df`: data.frame
- `q`: Penalty term in the stopping criterion (0 = AIC and 1 = BIC)
- `thres`: A threshold mechanism for choosing between two different ways of calculating the entropy. Can Speed up the procedure with the "correct" value.

Details

A `fwd` object can be created using the `gengraph` constructor with `type = "fwd"

Value

A `fwd` object; a subclass of `gengraph` used for forward selection.

References


See Also

`fit_graph`, `walk.bwd`, `gengraph`
Examples

d <- derma[, 10:25]

g <- gengraph(d, type = "fwd")
s <- walk(g, d)
print(s)
plot(s)
adj_lst(s)
adj_mat(s)
Index

* datasets
  * derma, 6
  * adj_lst, 3, 10
  * adj_lst.gengraph, 9, 11
  * adj_mat, 3, 10
  * adj_mat.gengraph, 9, 11
  * as_adj_lst, 4
  * as_adj_mat, 4
  * asigraph, 5, 10
  * components, 5
  * derma, 6
  * dfs, 6
  * dgm_sim_from_graph, 7
  * entropy, 8
  * ess (ess-package), 2
  * ess-package, 2
  * fit_components, 8
  * fit_graph, 9, 9, 11, 17–19
  * gengraph, 9, 10, 11, 17–19
  * is_decomposable, 12, 16
  * make_complete_graph, 12
  * make_null_graph, 13
  * mcs, 13, 16
  * plot.gengraph, 14
  * print.gengraph, 15
  * print.tree, 15
  * rip, 16
  * subgraph, 16
  * walk, 17
  * walk.bwd, 9, 11, 18, 19
  * walk.fwd, 9, 11, 17, 18, 19