

# Package ‘ess’

May 24, 2020

**Title** Efficient Stepwise Selection in Decomposable Models

**Version** 1.0

**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, Matrix

**LinkingTo** Rcpp

**RoxygenNote** 7.1.0

**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** 2020-05-24 15:20:05 UTC

## R topics documented:

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

components . . . . .	5
derma . . . . .	5
dfs . . . . .	6
dgm_sim_from_graph . . . . .	6
entropy . . . . .	7
fit_components . . . . .	8
fit_graph . . . . .	9
gengraph . . . . .	10
is_decomposable . . . . .	11
make_complete_graph . . . . .	12
make_null_graph . . . . .	12
mcs . . . . .	13
plot.gengraph . . . . .	13
print.gengraph . . . . .	14
print.tree . . . . .	14
rip . . . . .	15
subgraph . . . . .	15
walk . . . . .	16
walk.bwd . . . . .	17
walk.fwd . . . . .	18
<b>Index</b>	<b>20</b>

---

 ess-package

*ess: Efficient Stepwise Selection in Decomposable Models*


---

## 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>
- Report bugs at <https://github.com/mlindsk/ess/issues>

---

adj_lst	<i>Adjacency List</i>
---------	-----------------------

---

**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	<i>Adjacency Matrix</i>
---------	-------------------------

---

**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	<i>Converts an adjacency matrix to an adjacency list</i>
------------	--

---

**Description**

Converts an adjacency matrix to an adjacency list

**Usage**

```
as_adj_lst(A)
```

**Arguments**

A	Adjacency matrix
---	------------------

---

as_adj_mat	<i>Converts an adjacency list to an adjacency matrix</i>
------------	--

---

**Description**

Converts an adjacency list to an adjacency matrix

**Usage**

```
as_adj_mat(adj)
```

**Arguments**

adj	Adjacency list
-----	----------------

**Value**

An adjacency matrix

**Examples**

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

---

components	<i>Finds the components of a graph</i>
------------	--

---

**Description**

Finds the components of a graph

**Usage**

```
components(adj)
```

**Arguments**

adj           Adjacency list or gengraph object

**Value**

A list where the elements are the components of the graph

---

derma	<i>Dermatology Database</i>
-------	-----------------------------

---

**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")
```

---

dgm\_sim\_from\_graph *Simulate observations from a decomposable graphical model*

---

**Description**

Simulate observations from a decomposable graphical model

**Usage**

```
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**

```
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)

# Converting the g to a gengraph object and plot

d <- data.frame(A = "", B = "", X = "", Y = "") # auxillary data
g <- gengraph(d, adj = g)
plot(g)
```

---

entropy

*Joint Entropy*

---

**Description**

Calculates the joint entropy over discrete variables in `df`

**Usage**

```
entropy(df, thres = 5)
```

**Arguments**

<code>df</code>	data.frame
<code>thres</code>	A threshold mechanism for choosing between two different ways of calculating the entropy. Can Speed up the procedure with the "correct" value.

**Value**

A number representing the entropy of the variables in df.

**Examples**

```
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**

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

**Arguments**

df	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 ( $0 = \text{AIC}$ and $1 = \text{BIC}$ )
as_gen	Logical. Convert to gengraph or not. If true, the graph can be plotted.
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**

A gengraph object



**See Also**

[fit\\_graph](#), [adj\\_lst.gengraph](#), [adj\\_mat.gengraph](#), [walk.fwd](#), [walk.bwd](#), [gengraph](#)

---

fit_graph	<i>Fit a decomposable graphical model</i>
-----------	---

---

**Description**

A generic method for structure learning in decomposable graphical models

**Usage**

```
fit_graph(
  df,
  type = "fwd",
  adj = NULL,
  q = 0.5,
  trace = TRUE,
  thres = 5,
  wrap = TRUE
)
```

**Arguments**

df	Character data.frame
type	Character ("fwd", "bwd", "tree" or "tfwd")
adj	Adjacency list of a decomposable graph
q	Penalty term in the stopping criterion ( $0 = \text{AIC}$ and $1 = \text{BIC}$ )
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

**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**

<https://arxiv.org/abs/1301.2267>, <https://doi.org/10.1109/ictai.2004.100>

**See Also**

[fit\\_components](#), [adj\\_lst.gengraph](#), [adj\\_mat.gengraph](#), [walk.fwd](#), [walk.bwd](#), [gengraph](#)

**Examples**

```
g <- fit_graph(derma, trace = FALSE, q = 0)
print(g)
plot(g)

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

# Cliques in the graph
rip(adjl)$C

# Components of the graph
components(adjl) # only one here
```

---

gengraph

*A generic and extendable structure for decomposable graphical models*

---

**Description**

A generic structure for decomposable graphical models

**Usage**

```
gengraph(df, type = "gen", adj = NULL, q = 0.5, ...)
```

**Arguments**

df	data.frame
type	character ("fwd", "bwd", "tree", "tfwd", "gen")
adj	A user-specified adjacency list
q	Penalty term in the stopping criterion ( $\emptyset$ = AIC and 1 = BIC)
...	Not used (for extendibility)

**Value**

A graph 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	<i>A test for decomposability in undirected graphs</i>
-----------------	--

---

**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:
adj1 <- list(a = c("b", "d"), b = c("a", "c"), c = c("b", "d"), d = c("a", "c"))
is_decomposable(adj1) # 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

**Examples**

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

---

make\_null\_graph         *Make a null graph*

---

**Description**

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

**Usage**

```
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**

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

---

mcs	<i>Maximum Cardinality Search</i>
-----	-----------------------------------

---

**Description**

Maximum Cardinality Search

**Usage**

```
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**

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

---

plot.gengraph	<i>Plot</i>
---------------	-------------

---

**Description**

A wrapper around igraphs plot method for gengraph objects

**Usage**

```
## 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**

```
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	<i>Print</i>
----------------	--------------

---

**Description**

A print method for gengraph objects

**Usage**

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

**Arguments**

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

---

print.tree	<i>Print</i>
------------	--------------

---

**Description**

A print method for tree objects

**Usage**

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

**Arguments**

x	A tree object
...	Not used (for S3 compatability)

---

rip	<i>Runining Intersection Property</i>
-----	---------------------------------------

---

**Description**

Given a decomposable graph, this functions 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**

```
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**

```
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	<i>Subgraph</i>
----------	-----------------

---

**Description**

Construct a subgraph with a given set of nodes removed

**Usage**

```
subgraph(x, g)
```

**Arguments**

x	Character vector of nodes
g	Adjacency list (named) or a adjacency matrix with dimnames given as the nodes

**Value**

An adjacency list or adjacency matrix.

**Examples**

```
adj1 <- 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
g <- gengraph(d, type = "gen", adj = adj1)
plot(g)
subgraph(c("c", "b"), adj1)
subgraph(c("b", "d"), as_adj_mat(adj1))
```

---

walk	<i>Stepwise model selection</i>
------	---------------------------------

---

**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 ( $0 = \text{AIC}$ and $1 = \text{BIC}$ )
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 ( $0 = \text{AIC}$ and $1 = \text{BIC}$ )
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

```
d <- derma[, 10:25]

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

---

walk.fwd	<i>Stepwise efficient forward selection in decomposable graphical models</i>
----------	--

---

## Description

Stepwise efficient forward selection in decomposable graphical models

## Usage

```
## 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 ( $0 = \text{AIC}$ and $1 = \text{BIC}$ )
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

<https://arxiv.org/abs/1301.2267>, <https://doi.org/10.1109/ictai.2004.100>

## 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

## \*Topic **datasets**

derma, 5

adj\_lst, 3

adj\_lst.gengraph, 9–11

adj\_mat, 3

adj\_mat.gengraph, 9–11

as\_adj\_lst, 4

as\_adj\_mat, 4

components, 5

derma, 5

dfs, 6

dgm\_sim\_from\_graph, 6

entropy, 7

ess (ess-package), 2

ess-package, 2

fit\_components, 8, 10

fit\_graph, 9, 9, 11, 16–18

gengraph, 9, 10, 10, 16–18

is\_decomposable, 11, 15

make\_complete\_graph, 12

make\_null\_graph, 12

mcs, 13, 15

plot.gengraph, 13

print.gengraph, 14

print.tree, 14

rip, 15

subgraph, 15

walk, 16

walk.bwd, 9–11, 17, 18

walk.fwd, 9–11, 16, 17, 18