

# Package ‘POSetR’

January 17, 2023

**Type** Package

**Title** Partially Ordered Sets in R

**Version** 1.1.1

**Date** 2023-01-16

**Author** Alberto Arcagni [aut, cre],  
Alessandro Avellone [aut],  
Marco Fattore [aut]

**Maintainer** Alberto Arcagni <alberto.arcagni@uniroma1.it>

**Description** Provides a set of basic tools for generating, analyzing, summarizing and visualizing finite partially ordered sets. In particular, it implements flexible and very efficient algorithms for the extraction of linear extensions and for the computation of mutual ranking probabilities and other user-defined functionals, over them. The package is meant as a computationally efficient “engine”, for the implementation of data analysis procedures, on systems of multi-dimensional ordinal indicators and partially ordered data, in the spirit of Fattore, M. (2016) “Partially ordered sets and the measurement of multidimensional ordinal deprivation”, *Social Indicators Research* <[DOI:10.1007/s11205-015-1059-6](https://doi.org/10.1007/s11205-015-1059-6)>, and Fattore M. and Arcagni, A. (2018) “A reduced posetic approach to the measurement of multidimensional ordinal deprivation”, *Social Indicators Research* <[DOI:10.1007/s11205-016-1501-4](https://doi.org/10.1007/s11205-016-1501-4)>.

**License** GPL (>= 2)

**Imports** Rcpp (>= 1.0.0), igraph, methods, Rdpack

**RdMacros** Rdpack

**LinkingTo** Rcpp

**SystemRequirements** C++17

**RoxygenNote** 7.2.1

**Encoding** UTF-8

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2023-01-17 10:40:02 UTC

**R topics documented:**

POSetR-package . . . . .	2
antiChain . . . . .	3
chain . . . . .	3
coverMatrix . . . . .	4
incidenceMatrix . . . . .	5
intersection . . . . .	5
LEapply . . . . .	6
plot.poset . . . . .	9
poset . . . . .	10
poset_from_dataframe . . . . .	11
poset_from_igraph . . . . .	12
poset_from_incidence . . . . .	13
print.poset . . . . .	14
print.summary_poset . . . . .	15
productOrder . . . . .	16
summary.poset . . . . .	17
<b>Index</b>	<b>18</b>

---

POSetR-package	<i>Partially Ordered Sets in R</i>
----------------	------------------------------------

---

**Description**

Provides a set of basic tools for generating, analyzing, summarizing and visualizing finite partially ordered sets. In particular, it implements flexible and very efficient algorithms for the extraction of linear extensions and for the computation of mutual ranking probabilities and other user-defined functionals, over them. The package is meant as a computationally efficient "engine", for the implementation of data analysis procedures, on systems of multidimensional ordinal indicators and partially ordered data, in the spirit of Fattore, M. (2016) "Partially ordered sets and the measurement of multidimensional ordinal deprivation", *Social Indicators Research* <DOI:10.1007/s11205-015-1059-6>, and Fattore M. and Arcagni, A. (2018) "A reduced posetic approach to the measurement of multidimensional ordinal deprivation", *Social Indicators Research* <DOI:10.1007/s11205-016-1501-4>.

**Author(s)**

Alberto Arcagni [aut, cre], Alessandro Avellone [aut], Marco Fattore [aut]

---

antiChain	<i>Generate an anti-chain from a vector of elements' labels</i>
-----------	---

---

**Description**

The function produces an antichain from the vector of elements' labels `elements`. If `elements` is a numeric vector, it is first coerced to a character vector.

An antichain is a poset with no comparabilities, therefore the order of the labels in `elements` does not matter.

**Usage**

```
antiChain(elements)
```

**Arguments**

`elements` a vector of characters with elements' labels.

**Value**

an environment of class `poset`.

**See Also**

function [poset](#) for more details and to create a generic poset and function [chain](#) to create a complete order.

**Examples**

```
antiChain(LETTERS[1:5])
```

---

chain	<i>Generate a complete order from a vector of elements' labels</i>
-------	--

---

**Description**

The function produces a complete order from the vector of elements' labels `elements`. If `elements` is a numeric vector, it is first coerced to a character vector.

A complete order is a poset where all of the elements are comparable; the order of the labels in `elements` defines their position in the chain, from bottom to top.

**Usage**

```
chain(elements)
```

**Arguments**

`elements` a vector of characters listing the elements' labels.

**Value**

an environment of class `poset`.

**See Also**

function `poset` for more details and to create a generic poset and function `antiChain` to create an anti-chain.

**Examples**

```
chain(LETTERS[1:5])
```

---

coverMatrix	<i>Poset cover matrix</i>
-------------	---------------------------

---

**Description**

The function returns the cover matrix for objects of class `poset`.

**Usage**

```
coverMatrix(x)
```

**Arguments**

`x` an object of class `poset`.

**Value**

a logical square matrix.

**Examples**

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)
coverMatrix(p)
```

---

incidenceMatrix	<i>Poset incidence matrix</i>
-----------------	-------------------------------

---

**Description**

The function returns the incidence matrix for objects of class [poset](#).

**Usage**

```
incidenceMatrix(x)
```

**Arguments**

x                    an object of class [poset](#).

**Value**

a logical square matrix.

**Examples**

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)
incidenceMatrix(p)
```

---

intersection	<i>Intersection of two posets</i>
--------------	-----------------------------------

---

**Description**

Given two posets X and Y on the same set, `intersection` returns the poset Z defined by  $a < b$  in Z if and only if  $a < b$  in X and  $a < b$  in Y.

**Usage**

```
intersection(x, y)
```

```
x %it% y
```

**Arguments**

x                    an S4 object of class `Rcpp_POSet`, see [poset](#) for details.  
y                    an S4 object of class `Rcpp_POSet`, see [poset](#) for details.

**Value**

an S4 object of class Rcpp\_POSet, see [poset](#) for details

**References**

Davey BA, Priestley HA (2002). *Introduction to lattices and order*. Cambridge university press.

**See Also**

[poset](#)

**Examples**

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)
plot(p)
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "d", "b"
), ncol = 2, byrow = TRUE)
q <- poset(x = dom)
plot(q)
plot(p %it% q)
```

---

LEapply

*Applies scalar functions over the set of linear extensions of a poset and returns the corresponding average values*

---

**Description**

"LEapply" is composed of three main elements: (i) the linear extensions generator, (ii) the application of the argument functions to the linear extensions and (iii) the computation of the averages of the results, for each function separately; see Fattore M (2016). "Partially ordered sets and the measurement of multidimensional ordinal deprivation." *Social Indicators Research*, **128**(2), 835–858..

**Usage**

```
LEapply(x, ...)

## S3 method for class 'poset'
LEapply(
  x,
```

```

FUN = "MutualRankingProbability",
...,
generator = c("Al1LE", "BubleyDyer"),
bubleydyer.precision = 10,
bubleydyer.nit = NULL,
bubleydyer.progressBar = TRUE,
degrees = NULL
)

```

## Arguments

<code>x</code>	an environment of class <code>poset</code> , see <a href="#">poset</a> for details.
<code>...</code>	optional arguments to <code>FUN</code> .
<code>FUN</code>	the function, or a list of functions, to be applied to each linear extension: see 'Details'.
<code>generator</code>	a string specifying the method used to generate the linear extensions. The default value is "Al1LE". See section 'Details' below.
<code>bubleydyer.precision</code>	considered only if "BubleyDyer" generator is selected. It corresponds to the number of digit precision of the frequencies in the sampling distributions of linear extensions.
<code>bubleydyer.nit</code>	considered only if "BubleyDyer" generator is selected. Number of iterations in the Bubley-Dyer algorithm, if <code>NULL</code> (default) it is set as indicated in Bubley and Dyer (1999) depending on the value of <code>bubleydyer.precision</code> and the number of elements of the poset.
<code>bubleydyer.progressBar</code>	logical that indicates whether to show a text progress bar or not
<code>degrees</code>	to generate the lexicographic linear extensions of a product order, the poset <code>x</code> describes the dominance (e.g. relative importance) between ordinal variables and <code>degrees</code> is a numerical vector specifying the number of degrees of each variable, represented by in the poset.

## Details

Argument `FUN` must be either a function or a list of functions, each one depending on a vector of characters representing the names of the elements of the poset.

If `degrees` is not `NULL` but a numerical vector as long as the number of elements in the poset, the poset elements are considered as ordinal variables. Therefore `degrees` represents their number of degrees that are represented as integer numbers starting from 0. In this case, `LEapply` generates the lexicographical linear extensions of the product order of the ordeinal variables. Its elements are called profiles and they are obtained by the combination of the degrees of variables separated by a dash. For details about lexicographical linear extensions and profiles see Fattore M, Arcagni A (2018). "A reduced posetic approach to the measurement of multidimensional ordinal deprivation." *Social Indicators Research*, **136**(3), 1053–1070..

Some functions are already implemented in the C++ library and they can be called by their names. Currently, such functions are "MutualRankingProbability", "Separation", and "AverageHeight".

Each function in FUN must return a numerical or logical matrix. Each function can depend on additional arguments that can be passed through ...; such additional arguments must be the same for all the functions in the list.

Argument generator specifies the linear extension generation algorithm. The available generators are "ALLLE", that produces all of the linear extensions of the input poset, and "BubleyDyer", which samples uniformly from the set of linear extensions, through an MCMC algorithm (Bubley and Dyer 1999).

### Value

The average values of the argument functions FUN over the set of linear extensions (or lexicographic ones if degrees argument is not NULL).

### References

- Bubley R, Dyer M (1999). "Faster random generation of linear extensions." *Discrete mathematics*, **201**(1-3), 81–88.
- Fattore M (2016). "Partially ordered sets and the measurement of multidimensional ordinal deprivation." *Social Indicators Research*, **128**(2), 835–858.
- Fattore M, Arcagni A (2018). "A reduced posetic approach to the measurement of multidimensional ordinal deprivation." *Social Indicators Research*, **136**(3), 1053–1070.
- Habib M, Medina R, Nourine L, Steiner G (2001). "Efficient algorithms on distributive lattices." *Discrete Applied Mathematics*, **110**(2-3), 169–187.

### See Also

[poset](#)

### Examples

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)

## Not run:
LEapply(
  x = p,
  FUN = "MutualRankingProbability",
  generator = "ALLLE",
  degrees = c(3, 2, 3, 2)
)

a_rank_dist <- function(le) {
  return(matrix(le == "a"))
}
LEapply(x = p, FUN = a_rank_dist)
## End(Not run)
```



plot.poset

*Plotting the Hasse diagram of a poset***Description**

plot produces an [igraph](#) object and shows the Hasse diagram.

**Usage**

```
## S3 method for class 'poset'
plot(
  x,
  vertex.color = rgb(1, 1, 1, 1),
  vertex.label = x$pointer$elements(),
  vertex.label.color = rgb(0, 0, 0, 1),
  vertex.label.family = "sans",
  edge.color = rgb(0, 0, 0, 1),
  edge.label = NA,
  edge.arrow.mode = "-",
  asp = 0,
  ...,
  equispaced = FALSE,
  show = TRUE
)
```

**Arguments**

x	an S4 object of class Rcpp_POSet, see <a href="#">poset</a> for details.
vertex.color	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
vertex.label	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
vertex.label.color	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
vertex.label.family	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
edge.color	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
edge.label	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
edge.arrow.mode	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
asp	argument of the plot.igraph function, see <a href="#">igraph.plotting</a> for details.
...	additional plotting parameters, see <a href="#">igraph.plotting</a> for details.
equispaced	logical, if TRUE the nodes on the same level of the Hasse diagram are horizontally equispaced.
show	logical, if TRUE (default) the Hasse diagram is plotted.

**Details**

`plot.poset` computes the cover relation and produces the corresponding Directed Acyclic Graph (DAG), as an `igraph` object, returned as invisible output. Function `layout_with_sugiyama` generates the DAG layout with edges oriented from top to bottom. When `equispaced=TRUE`, nodes on the same Hasse diagram level are horizontally equispaced.

The Hasse diagram is displayed by a call to `plot.igraph` (some default argument values are set to get a cleaner plot, by exploiting Hasse diagram properties).

Setting `show = FALSE` produces the `igraph` object, without showing the Hasse diagram.

Note that

**Value**

an `igraph` object.

**See Also**

`poset`, `igraph`, `igraph.plotting`

**Examples**

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)
hasse <- plot(p)
class(hasse)
```

---

poset

*Generates a Partially Ordered SET from the list of dominances*

---

**Description**

Function `poset` creates a poset from a dominance list. `x` argument is a two-column matrix, each row defines a pair of comparable elements, where the element in the first column is dominated by or coincide with the element in the second column. If the elements of `x` are numeric, they are first coerced to character and used as elements labels.

**Usage**

```
poset(x = NULL, elements = unique(as.character(x)))
```

**Arguments**

`x` an object of class `matrix` with two columns, listing the dominances, by rows.  
`elements` a vector of characters listing all the labels of the elements.

**Details**

A partial order relation is reflexive, transitive and anti-symmetric. Given the dominance list provided by the user, the function produces the smallest poset comprising them (reflexive and transitive closure); in case the dominances provided by the user imply non-trivial cycles, violating anti-symmetry, the function returns an error.

By default `elements` is equal to all the different labels available in `x`. If some elements are incomparable, list all of the elements in `elements` or include self-comparabilities in `x`. Notice that antichains can be created in a simpler way, by function [antiChain](#).

**Value**

an environment of class `poset`; this class contains a pointer with different C++ methods used by other functions of the package.

**See Also**

in the package are available functions that simplify the creation of particular posets: [antiChain](#) to create a poset without comparabilities, [chain](#) to create a complete order, [poset\\_from\\_dataframe](#) to create a poset from a statistical population.

**Examples**

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
poset(x = dom)
poset(x = dom, elements = letters[1:5])
```

---

`poset_from_dataframe` *Generates a Partially Ordered SET from a data frame through the component-wise comparability relation*

---

**Description**

This function generates a poset from an object `x` of class `data.frame` that lists by rows all the observations in a statistical population and by columns the values of different orderable variables (i.e. vectors of class `ordered` or `numeric`).

**Usage**

```
poset_from_dataframe(x, collapse = "-")
```

**Arguments**

`x` an object of class `data.frame`.

`collapse` an object of class `character` that defines the separator between observations with the same profile, by default `"-"`. For more information see the details section.

**Details**

The poset is generated through the component-wise comparability relation (Davey and Priestley 2002). Note that some observations may have the same profile (i.e. the same values for all the variables) therefore they have to be included into the same element of the poset, than the label of such element correspond to the labels of the statistical units separated by the collapse character.

**Value**

an S4 object of class `Rcpp_POSet`; this class contains different C++ methods used by other functions of the package.

**References**

Davey BA, Priestley HA (2002). *Introduction to lattices and order*. Cambridge university press.

**See Also**

[poset](#).

**Examples**

```
set <- ordered(letters[1:5])
set.seed(0)
n <- 30
x <- data.frame(
  A = sample(set, n, replace = TRUE),
  B = sample(set, n, replace = TRUE),
  C = sample(set, n, replace = TRUE)
)

p <- poset_from_dataframe(x, collapse = ",")

plot(p, equispaced = TRUE, vertex.frame.color = "lightgray", vertex.color = "lightgray")
```

---

poset_from_igraph	<i>Generates a Partially Ordered SET from a Directed Acyclic Graph (DAG)</i>
-------------------	--

---

**Description**

This function generates a poset from an object `g`. The function checks if the graph is directed and cyclic in order it represent the cover-relation.

**Usage**

```
poset_from_igraph(g)
```

**Arguments**

`g` an igraph object

**Details**

Note that the Hasse diagram represents the cover relation from the top to the bottom, therefore its representation is the opposite of the plot of the graph with Sugiyama layout.

**Value**

an environment of class `poset`.

**References**

Davey BA, Priestley HA (2002). *Introduction to lattices and order*. Cambridge university press.

**See Also**

[poset](#).

**Examples**

```
library(igraph)
g <- make_tree(10)
ly <- igraph::layout_with_sugiyama(g)$layout
g$layout <- ly
plot(g)
p <- poset_from_igraph(g)
plot(p)
```

---

`poset_from_incidence` *Generates a Partially Ordered SET from an incidence matrix*

---

**Description**

This function generates a poset from an incidence matrix `x`. Such matrix is a named and logical that is TRUE if the row element is dominated by the column one, FALSE otherwise. Such incidence relation should be reflexive, anti-symmetric, and transitive..

**Usage**

```
poset_from_incidence(x)
```

**Arguments**

x a logical matrix.

**Value**

an environment of class poset.

**References**

Davey BA, Priestley HA (2002). *Introduction to lattices and order*. Cambridge university press.

**See Also**

[poset](#).

**Examples**

```
x <- c(
  TRUE, FALSE, FALSE, FALSE, FALSE, FALSE,
  TRUE, TRUE, FALSE, FALSE, FALSE, FALSE,
  TRUE, TRUE, TRUE, FALSE, FALSE, FALSE,
  TRUE, FALSE, FALSE, TRUE, FALSE, FALSE,
  TRUE, TRUE, FALSE, TRUE, TRUE, FALSE,
  TRUE, TRUE, TRUE, TRUE, TRUE, TRUE
)
x <- matrix(x, nrow = 6, ncol = 6)
rownames(x) <- colnames(x) <- LETTERS[1:6]
p <- poset_from_incidence(x)
plot(p)
```

---

print.poset

*Method for the print function that shows the poset elements and comparabilities*

---

**Description**

print prints the list of poset elements and all of the strict dominances in it.

**Usage**

```
## S3 method for class 'poset'
print(x, max = NULL, ...)
```

**Arguments**

x an object of class Rcpp\_POSet.  
 max a non-null value for max specifies the approximate maximum number of entries to be printed. The default, NULL, uses [getOption\("max.print"\)](#): see that help page for more details.  
 ... further arguments passed to or from other methods.

**Value**

nothing

**Examples**

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)
print(p)
```

---

print.summary\_poset    *Method for the print function that shows the poset summary*

---

**Description**

print prints the poset summary.

**Usage**

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

**Arguments**

x                    an object of class summary\_poset.  
...                   further arguments passed to or from other methods.

**Value**

nothing

**See Also**

[summary\\_poset](#)

**Examples**

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)
summary(p)

summary(chain(1:4))
summary(antiChain(LETTERS[1:5]))
```

---

productOrder	<i>Product order between two posets</i>
--------------	---

---

### Description

The function returns the product poset of two posets X and Y.

### Usage

```
productOrder(x, y, sep = "-")
```

```
x %po% y
```

### Arguments

x	an S4 object of class poset, see <a href="#">poset</a> for details.
y	an S4 object of class poset, see <a href="#">poset</a> for details.
sep	a character object indicating the separator to be used to paste profiles names.

### Value

an environment of class poset, see [poset](#) for details

### References

Davey BA, Priestley HA (2002). *Introduction to lattices and order*. Cambridge university press.

### See Also

[poset](#)

### Examples

```
dom <- matrix(c(
  "a", "b",
  "c", "b",
  "b", "d"
), ncol = 2, byrow = TRUE)
p <- poset(x = dom)
q <- chain(1:3)
plot(p %po% q)
```



---

summary.poset	<i>Poset summary</i>
---------------	----------------------

---

**Description**

Method of the [summary](#) function for objects of class [poset](#).

**Usage**

```
## S3 method for class 'poset'  
summary(object, ...)
```

**Arguments**

`object` an object of class [poset](#).  
`...` further arguments passed to or from other methods.

**Value**

An S3 object of class `summary_poset` listing and counting the poset elements, their strict dominances and their incomparabilities.

**Examples**

```
dom <- matrix(c(  
  "a", "b",  
  "c", "b",  
  "b", "d"  
) , ncol = 2, byrow = TRUE)  
p <- poset(x = dom)  
summary(p)  
  
summary(chain(1:4))  
summary(antiChain(LETTERS[1:5]))
```

# Index

`%it%` (intersection), [5](#)  
`%po%` (productOrder), [16](#)

`antiChain`, [3](#), [4](#), [11](#)

C++Object-class (poset), [10](#)  
`chain`, [3](#), [3](#), [11](#)  
`coverMatrix`, [4](#)

`getOption`, [14](#)

`igraph`, [9](#), [10](#)  
`igraph.plotting`, [9](#), [10](#)  
`incidenceMatrix`, [5](#)  
`intersection`, [5](#)

`layout_with_sugiyama`, [10](#)  
`LEapply`, [6](#)

`plot.igraph`, [10](#)  
`plot.poset`, [9](#)  
`pointerRebuild` (poset), [10](#)  
`POSet` (poset), [10](#)  
`poset`, [3–10](#), [10](#), [12–14](#), [16](#), [17](#)  
`poset-class` (poset), [10](#)  
`poset_from_dataframe`, [11](#), [11](#)  
`poset_from_igraph`, [12](#)  
`poset_from_incidence`, [13](#)  
`POSetR` (poset), [10](#)  
`POSetR-package`, [2](#)  
`print.poset`, [14](#)  
`print.summary_poset`, [15](#)  
`productOrder`, [16](#)

`Rcpp_POSet-class` (poset), [10](#)

`summary`, [17](#)  
`summary.poset`, [15](#), [17](#)