Package 'utility'

July 2, 2014

Type Package

Title Construct, Evaluate and Plot Value and Utility Functions

Version 1.2

Date 2014-01-15

Author Peter Reichert <peter.reichert@eawag.ch>
with contributions by Nele Schuwirth <nele.schuwirth@eawag.ch>

Maintainer Peter Reichert <peter.reichert@eawag.ch>

Description Construct and plot objective hierarchies and associated value and utility functions.
Evaluate the values and utilities and visualize the results as colored objective hierarchies or tables.
Visualize uncertainty by plotting median and quantile intervals within the nodes of objective hierarchies.
Get numerical results of the evaluations in standard R data types for further processing.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2014-01-16 15:39:56

R topics documented:

`utility-package` *Construct, Evaluate and Plot Value and Utility Functions*

Description

Construct and plot objective hierarchies and associated value and utility functions. Evaluate the values and utilities and visualize the results as colored objective hierarchies or tables. Visualize uncertainty by plotting median and quantile intervals within the nodes of the objective hierarchy. Get numerical results of the evaluations in standard R data types for further processing.

Details

Package:	utility
Type:	Package
Version:	1.2
Date:	2014-01-15
License:	GPL (>= 2)

An objective hierarchy and an associated value or utility function is constructed by constructing the nodes of the hierarchy starting from the end nodes and proceeding to the higher hierarchies. Five types of end nodes are distinguished: End nodes of the class `utility.endnode.discrete` define a value or utility function for an attribute that has a finite number of discrete numeric or non-numeric levels. End nodes of the classes `utility.endnode.intpol1d` and `utility.endnode.parfun1d` implement single-attribute value or utility functions that accept a continuous argument. The first of these functions allows the user to specify attribute-value pairs and performs linear interpolation between these points. The second function allows the user to specify any parameteric function that is implemented as a function in R. End nodes of the class `utility.endnode.intpol2d` implement interpolated value or utility functions that are based on two attributes. Finally, end nodes of the class `utility.endnode.cond` implement value or utility functions that assign different value or utility functions to a finite set of attribute combinations. These end nodes can be implemented by using the following constructors.

utility.endnode.discrete.create
utility.endnode.intpol1d.create
utility.endnode.parfun1d.create
utility.endnode.intpol2d.create
utility.endnode.cond.create

To advance to higher hierarchical levels, values or utilities at lower levels must be aggregated to the next higher level. This is done ab aggregation nodes of the class `utility.aggregation`. Such nodes can be implemented by using the following constructor:

utility.aggregation.create

Finally, to provide decision support under uncertainty, values at an adequate level of the objectives hierarchy must be converted to utilities by accounting for the risk attitude of the decision maker. Similar to the single-attribute value or utility functions, this can either be done by linear interpolation with a node of the class `utility.conversion.intpol` or by using a parametric funciton in a node of the class `utility.conversion.parfun`. These conversion nodes can be implemented by the constructors:

utility.conversion.intpol.create
utility.conversion.parfun.create

The definition of the objective hierarchy and the associated value and utility function can then be listed or visualized by using the generic functions

print
summary
plot

which automaticall call the implementation corresponding to the node specified as the first argument:

print.utility.endnode.discrete
print.utility.endnode.intpol1d
print.utility.endnode.parfun1d
print.utility.endnode.intpol2d
print.utility.endnode.cond
print.utility.aggregation
print.utility.conversion.intpol
print.utility.conversion.parfun

summary.utility.endnode.discrete
summary.utility.endnode.intpol1d
summary.utility.endnode.parfun1d
summary.utility.endnode.intpol2d
summary.utility.endnode.cond
summary.utility.aggregation
summary.utility.conversion.intpol

summary.utility.conversion.parfun

plot.utility.endnode.discrete
plot.utility.endnode.intpol1d
plot.utility.endnode.parfun1d
plot.utility.endnode.intpol2d
plot.utility.endnode.cond
plot.utility.aggregation
plot.utility.conversion.intpol
plot.utility.conversion.parfun

The value or utility function can then be evaluated by applying the generic function

evaluate

that again calls automatically the corresponding class-specific function

evaluate.utility.endnode.discrete
evaluate.utility.endnode.intpol1d
evaluate.utility.endnode.parfun1d
evaluate.utility.endnode.intpol2d
evaluate.utility.endnode.cond
evaluate.utility.aggregation
evaluate.utility.conversion.intpol
evaluate.utility.conversion.parfun

This function requires the provision of observed or predicted attributes of the valued system and returns the corresponding values or utilities of all nodes of the hierarchy. These results can then be visualized by providing them to the generic function

plot

in addition to the definition of the objective hierarchy stored in the variable corresponding to the highest node of the hierarchy. Again, this function automatically calls the correct class-specific implementation (the root of the hierarchy will be an aggregation or a conversion node, not an end node):

plot.utility.aggregation
plot.utility.conversion.intpol
plot.utility.conversion.parfun

This proceedure guarantees easy handling with the simple commands `print`, `summary`, `evaluate`, and `plot` and the specific function descriptions provided above are only required to check advanced attributes.

Author(s)

Peter Reichert <peter.reichert@eawag.ch> with contributions by Nele Schuwirth <nele.schuwirth@eawag.ch>

Maintainer: Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

Examples

```
# define discrete end node for width variability
# (attribute "widthvariability_class" with levels "high",
# "moderate" and "none")

widthvar <-
  utility.endnode.discrete.create(
    name.node     = "width variability",
    attrib.levels = data.frame(widthvariability_class=
      c("high","moderate","none")),
    u             = c(1,0.4125,0),
    names.u       = c("u.high","u_moderate","u.none"),
    required      = FALSE,
    utility       = FALSE)

# define 1d interpolation end node for bed modification with
# riprap
# (attribute "bedmodfract_percent" with levels from 0 to 100)

bedmod_riprap <-
  utility.endnode.intpol1d.create(
    name.node   = "bed modification riprap",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,100),
    u           = c(1,0.775,0.5625,0.24),
    required    = FALSE,
    utility     = FALSE)

# define 1d interpolation end node for bed modification with
# other material
# (attribute "bedmodfract_percent" with levels from 0 to 100)

bedmod_other <-
  utility.endnode.intpol1d.create(
```

```
    name.node   = "bed modification other",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,100),
    u           = c(1,0.775,0.5625,0),
    required    = FALSE,
    utility     = FALSE)

# define combination end node for bed modification
# (attributes "bedmodtype_class" and "bedmodfract_percent")

bedmod <-
  utility.endnode.cond.create(
    name.node     = "bed modification",
    attrib.levels = data.frame(bedmodtype_class=
      c("riprap","other")),
    nodes         = list(bedmod_riprap,bedmod_other),
    required      = FALSE,
    utility       = FALSE)

# define 1d interpolation end node for bank modification with
# permeable material
# (attribute "bankmodfract_percent" with levels from 0 to 100)

bankmod_perm <-
  utility.endnode.intpol1d.create(
    name.node   = "bank modification perm",
    name.attrib = "bankmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,60,100),
    u           = c(1,0.8667,0.675,0.4125,0.24),
    required    = FALSE,
    utility     = FALSE)

# define 1d interpolation end node for bank modification with
# impermeable material
# (attribute "bankmodfract_percent" with levels from 0 to 100)

bankmod_imperm <-
  utility.endnode.intpol1d.create(
    name.node   = "bank modification imperm",
    name.attrib = "bankmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,60,100),
    u           = c(1,0.775,0.5625,0.24,0),
    required    = FALSE,
    utility     = FALSE)

# define combination end node for bank modification
# (attributes "bankmodtype_class" and "bankmodfract_percent")

bankmod <-
  utility.endnode.cond.create(
```

```
    name.node     = "bank modification",
    attrib.levels = data.frame(bankmodtype_class=
      c("perm","imperm")),
    nodes         = list(bankmod_perm,bankmod_imperm),
    required      = FALSE,
    utility       = FALSE)

# define 2d interpolation end node for riparian zone width
# (attributes "riparianzonewidth_m" and "riparianzonewidth_m")

riparzone_width <-
  utility.endnode.intpol2d.create(
    name.node   = "riparian zone width",
    name.attrib = c("riverbedwidth_m","riparianzonewidth_m"),
    ranges      = list(c(0,16),c(0,30)),
    isolines    = list(list(x=c(0,16),y=c(0,0)),
                       list(x=c(0,2,10,16),y=c(5,5,15,15)),
                       list(x=c(0,16),y=c(15,15)),
                       list(x=c(0,16),y=c(30,30))),
    u           = c(0.0,0.6,1.0,1.0),
    lead         = 1,
    utility      = FALSE)

# define discrete end node for riparian zone vegetation
# (attriute "riparianzoneveg_class" with levels "natural",
# "seminatural" and "artificial")

riparzone_veg <-
  utility.endnode.discrete.create(
    name.node     = "riparian zone veg.",
    attrib.levels = data.frame(riparianzoneveg_class=
      c("natural","seminatural","artificial")),
    u             = c(1,0.5625,0),
    required      = FALSE,
    utility       = FALSE)

# define aggregation node for riparian zone

riparzone <-
  utility.aggregation.create(
    name.node = "riparian zone",
    nodes     = list(riparzone_width,riparzone_veg),
    name.fun  = "utility.aggregate.cobbdouglas",
    par       = c(1,1),
    required  = FALSE)

# define aggregation node for ecomorphological state

morphol <-
  utility.aggregation.create(
    name.node = "ecomorphology",
    nodes     = list(widthvar,bedmod,bankmod,riparzone),
    name.fun  = "utility.aggregate.mix",
```

```
    par       = c(0.25,0.25,0.25,0.25,0,0,1),
    names.par = c("w_widthvar","w_bedmod","w_bankmod","w_riparzone",
                  "w_add","w_min","w_cobbdouglas"),
    required  = TRUE)

# print individual definitions

print(widthvar)
print(bedmod)

# print all definitions

print(morphol)

# plot objectives hierarchy with attributes

plot(morphol)

# plot individual nodes:

plot(widthvar)
plot(widthvar,par=c(u_moderate=0.2))
plot(bedmod_other)
plot(bankmod)
plot(riparzone_width)

# plot selected node definitions of a hierarchy

plot(morphol,type="nodes",nodes=c("width variability",
                                  "bed modification other",
                                  "bank modification",
                                  "riparian zone width"))

# evaluate value function for data sets and plot colored hierarchies
# and table

attrib_channelized <- data.frame(widthvariability_class = "none",
                                 bedmodtype_class       = "riprap",
                                 bedmodfract_percent    = 50,
                                 bankmodtype_class      = "imperm",
                                 bankmodfract_percent   = 70,
                                 riverbedwidth_m        = 10,
                                 riparianzonewidth_m    = 5,
                                 riparianzoneveg_class  = "seminatural")
attrib_rehab       <- data.frame(widthvariability_class = "high",
                                 bedmodtype_class       = "riprap",
                                 bedmodfract_percent    = 50,
                                 bankmodtype_class      = "imperm",
                                 bankmodfract_percent   = 20,
                                 riverbedwidth_m        = 15,
                                 riparianzonewidth_m    = 15,
                                 riparianzoneveg_class  = "natural")
```

```
res_channelized     <- evaluate(morphol,attrib=attrib_channelized)
res_channelized_add <- evaluate(morphol,attrib=attrib_channelized,
                                par=c(w_add=1,w_min=0,w_cobbdouglas=0))
res_rehab           <- evaluate(morphol,attrib=attrib_rehab)
res_both            <- rbind(res_channelized,res_rehab)
rownames(res_both)  <- c("channelized","rehabilitated")

plot(morphol,u=res_channelized)
plot(morphol,u=res_channelized_add)
plot(morphol,u=res_rehab)
plot(morphol,u=res_rehab,uref=res_channelized)
plot(morphol,u=res_both,type="table",plot.val=FALSE)
plot(morphol,u=res_both,uref=res_channelized,type="table",plot.val=FALSE)

# consideration of uncertain attribute levels (higher uncertainty for
# predicted state after rehabilitation than for observed channelized state):

sampsize <- 2000

attrib_channelized_unc <- data.frame(
  widthvariability_class = rep("high",sampsize),
  bedmodtype_class       = rep("riprap",sampsize),
  bedmodfract_percent    = rnorm(sampsize,mean=50,sd=5),
  bankmodtype_class      = rep("imperm",sampsize),
  bankmodfract_percent   = rnorm(sampsize,mean=70,sd=5),
  riverbedwidth_m        = rep(10,sampsize),
  riparianzonewidth_m    = rep(5,sampsize),
  riparianzoneveg_class  = c("seminatural","artificial")[rbinom(sampsize,1,0.5)+1])

attrib_rehab_unc <- data.frame(
  widthvariability_class = c("moderate","high")[rbinom(sampsize,1,0.5)+1],
  bedmodtype_class       = rep("riprap",sampsize),
  bedmodfract_percent    = rnorm(sampsize,mean=50,sd=15),
  bankmodtype_class      = rep("imperm",sampsize),
  bankmodfract_percent   = rnorm(sampsize,mean=20,sd=5),
  riverbedwidth_m        = rnorm(sampsize,mean=10,sd=2),
  riparianzonewidth_m    = rnorm(sampsize,mean=10,sd=2),
  riparianzoneveg_class  = c("natural","seminatural")[rbinom(sampsize,1,0.5)+1])

res_channelized_unc <- evaluate(morphol,attrib=attrib_channelized_unc)
res_rehab_unc       <- evaluate(morphol,attrib=attrib_rehab_unc)

plot(morphol,u=res_channelized_unc)
plot(morphol,u=res_rehab_unc)
plot(morphol,u=res_rehab_unc,uref=res_channelized_unc)
plot(morphol,u=list(channelized=res_channelized_unc,rehabilitated=res_rehab_unc),
     type="table")
plot(morphol,u=list(channelized=res_channelized_unc,rehabilitated=res_rehab_unc),
     type="table",nodes=c("ecomorphology","riparian zone"))
plot(morphol,u=list(channelized=res_channelized_unc,rehabilitated=res_rehab_unc),
     type="table",levels=2)
plot(morphol,u=list(channelized=res_channelized_unc,rehabilitated=res_rehab_unc),
     type="table",levels=1)
```

```
plot(morphol,u=list(channelized=res_channelized_unc,rehabilitated=res_rehab_unc),
     uref=res_channelized_unc,
     type="table")
```

`evaluate` *Evaluate Node and Associated Hierarchy*

Description

Generic function to calculate values or utilities at all nodes of a hierarchy for given levels of the attributes.

Usage

```
evaluate(x, ...)
```

Arguments

`x`	node to be evaluated.
`...`	attribute levels have to be provided as an additional argument `attrib`; parameter values can optionally be provided as an additional argument `par`.

Value

Data frame with results of values or utilities at all nodes of the hierarchy for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.parfun1d.create,
utility.endnode.intpol2d.create,
utility.endnode.cond.create,
utility.aggregation.create,
utility.conversion.intpol.create,
utility.conversion.parfun.create

to create the nodes to be evaluated.

Examples

```
# see
help(utility)
# for examples.
```

---

evaluate.utility.aggregation

Evaluate Node and Associated Hierarchy

Description

Calculate values or utilities at all nodes of a hierarchy for given levels of the attributes.

Usage

```
## S3 method for class 'utility.aggregation'
evaluate(x, attrib, par = NA, ...)
```

Arguments

x	node to be evaluated.
attrib	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
...	currently no other arguments are implemented or passed further.

Value

Data frame with results of values or utilities at all nodes of the hierarchy for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

`utility.aggregation.create` to create the node,
`print.utility.aggregation` or `summary.utility.aggregation` to print its definition, and
`plot.utility.aggregation` to plot the node

and

`utility.endnode.discrete.create`,
`utility.endnode.intpol1d.create`,
`utility.endnode.parfun1d.create`,
`utility.endnode.intpol2d.create`,
`utility.endnode.cond.create`,
`utility.conversion.intpol.create`,
`utility.conversion.parfun.create`

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

---

evaluate.utility.conversion.intpol

Evaluate Node and Associated Hierarchy

Description

Calculate values or utilities at all nodes of a hierarchy for given levels of the attributes.

Usage

```
## S3 method for class 'utility.conversion.intpol'
evaluate(x, attrib, par = NA, ...)
```

Arguments

x	node to be evaluated.
attrib	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
...	currently no other arguments are implemented or passed further.

Value

Data frame with results of values or utilities at all nodes of the hierarchy for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

utility.conversion.intpol.create to create the node,
print.utility.conversion.intpol or summary.utility.conversion.intpol to print its definition, and
plot.utility.conversion.intpol to plot the node

and

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.parfun1d.create,

utility.endnode.intpol2d.create,
utility.endnode.cond.create.
utility.aggregation.create,
utility.conversion.parfun.create

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

`evaluate.utility.conversion.parfun`
Evaluate Node and Associated Hierarchy

Description

Calculate values or utilities at all nodes of a hierarchy for given levels of the attributes.

Usage

```
## S3 method for class 'utility.conversion.parfun'
evaluate(x, attrib, par = NA, ...)
```

Arguments

`x`	node to be evaluated.
`attrib`	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
`par`	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
`...`	currently no other arguments are implemented or passed further.

Value

Data frame with results of values or utilities at all nodes of the hierarchy for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

`utility.aggregation.create` to create the node,
`print.utility.aggregation` or `summary.utility.aggregation` to print its definition, and
`plot.utility.aggregation` to plot the node

and

`utility.endnode.discrete.create`,
`utility.endnode.intpol1d.create`,
`utility.endnode.parfun1d.create`,
`utility.endnode.intpol2d.create`,
`utility.endnode.cond.create`,
`utility.aggregation.create`,
`utility.conversion.intpol.create`

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

`evaluate.utility.endnode.cond`

Evaluate Node and Associated Hierarchy

Description

Calculate values or utilities at the node for given levels of the attributes.

Usage

```
## S3 method for class 'utility.endnode.cond'
evaluate(x, attrib, par = NA, ...)
```

Arguments

x	node to be evaluated.
attrib	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
...	currently no other arguments are implemented or passed further.

Value

Numeric vector of results of values or utilities at the node for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

utility.aggregation.create to create the node,
print.utility.aggregation or summary.utility.aggregation to print its definition, and
plot.utility.aggregation to plot the node

and

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.parfun1d.create,
utility.endnode.intpol2d.create,
utility.aggregation.create,

utility.conversion.intpol.create,
utility.conversion.parfun.create

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

`evaluate.utility.endnode.discrete` *Evaluate Node*

Description

Calculate values or utilities at the node for given levels of the attributes.

Usage

```
## S3 method for class 'utility.endnode.discrete'
evaluate(x, attrib, par = NA, ...)
```

Arguments

`x`	node to be evaluated.
`attrib`	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
`par`	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
`...`	currently no other arguments are implemented or passed further.

Value

Numeric vector of results of values or utilities at the node for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

`utility.aggregation.create` to create the node,
`print.utility.aggregation` or `summary.utility.aggregation` to print its definition, and
`plot.utility.aggregation` to plot the node

and

`utility.endnode.intpol1d.create`,
`utility.endnode.parfun1d.create`,
`utility.endnode.intpol2d.create`,
`utility.endnode.cond.create`,
`utility.aggregation.create`,
`utility.conversion.intpol.create`,
`utility.conversion.parfun.create`

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

`evaluate.utility.endnode.intpol1d`
Evaluate Node

Description

Calculate values or utilities at the node for given levels of the attributes.

Usage

```
## S3 method for class 'utility.endnode.intpol1d'
evaluate(x, attrib, par = NA, ...)
```

Arguments

x	node to be evaluated.
attrib	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
...	currently no other arguments are implemented or passed further.

Value

Numeric vector of results of values or utilities at the node for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

utility.aggregation.create to create the node,
print.utility.aggregation or
summary.utility.aggregation to print its definition, and
plot.utility.aggregation to plot the node

and

utility.endnode.discrete.create,
utility.endnode.parfun1d.create,
utility.endnode.intpol2d.create,
utility.endnode.cond.create,

utility.aggregation.create,
utility.conversion.intpol.create,
utility.conversion.parfun.create

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

`evaluate.utility.endnode.intpol2d` *Evaluate Node*

Description

Calculate values or utilities at the node for given levels of the attributes.

Usage

```
## S3 method for class 'utility.endnode.intpol2d'
evaluate(x, attrib, par = NA, ...)
```

Arguments

`x`	node to be evaluated.
`attrib`	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
`par`	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
`...`	currently no other arguments are implemented or passed further.

Value

Numeric vector of results of values or utilities at the node for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

`utility.aggregation.create` to create the node,
`print.utility.aggregation` or `summary.utility.aggregation` to print its definition, and
`plot.utility.aggregation` to plot the node

and

`utility.endnode.discrete.create`,
`utility.endnode.intpol1d.create`,
`utility.endnode.parfun1d.create`,
`utility.endnode.cond.create`,
`utility.aggregation.create`,
`utility.conversion.intpol.create`,
`utility.conversion.parfun.create`

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

`evaluate.utility.endnode.parfun1d`
Evaluate Node

Description

Calculate values or utilities at the node for given levels of the attributes.

Usage

```
## S3 method for class 'utility.endnode.parfun1d'
evaluate(x, attrib, par = NA, ...)
```

Arguments

x	node to be evaluated.
attrib	numeric vector with labelled components providing the levels of a single set of attributes or data frame for which each row provides such a set of attributes.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before evaluation.
...	currently no other arguments are implemented or passed further.

Value

Numeric vector of results of values or utilities at the node for all provided sets of attribute levels.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

utility.aggregation.create to create the node,
print.utility.aggregation or summary.utility.aggregation to print its definition, and
plot.utility.aggregation to plot the node

and

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.intpol2d.create,
utility.endnode.cond.create,
utility.aggregation.create,

utility.conversion.intpol.create,
utility.conversion.parfun.create

to create other nodes.

Examples

```
# see
help(utility)
# for examples.
```

---

plot.utility.aggregation
Plot Node Definition or Underlying Objectives Hierarchy

Description

Plot node definition or underlying objective hierarchy.

Usage

```
## S3 method for class 'utility.aggregation'
plot(x,
     u           = NA,
     uref        = NA,
     par         = NA,
     type        = c("hierarchy", "table", "node", "nodes"),
     nodes       = NA,
     col         = utility.calc.colors(),
     gridlines   = c(0.2, 0.4, 0.6, 0.8),
     main        = "",
     cex.main    = 1,
     cex.nodes   = 1,
     cex.attrib  = 1,
     f.reaches   = 0.2,
     f.nodes     = 0.2,
     with.attrib = TRUE,
     levels      = NA,
     plot.val    = TRUE,
     ...)
```

Arguments

x	node to be plotted.
u	(optional) vector or data frame with elements or columns labelled according to the nodes of the hierarchy containing values or utilities. Typically, this will be the complete output or an output row of the function

evaluate.utility.aggregation.
This input is only considered if the argument type is specified to be either "hierarchy" or "table". It is then used to color-code the boxes of the hierarchy representing value nodes or the table. If u is a data frame with more than one row and the argument type is equal to "hierarchy", then the median and quantile boxes are plotted for value nodes or the expected utility for utility nodes unless the argument main contains as many elements as the number of rows of u. In the latter case, separate hierarchies with color-coded boxes for value nodes are produced for all rows of u. For type equals "table", this argument can be a list of data frames to make it possible to plot uncertainty ranges from the samples provided in the list.

uref — (optional) vector or data frame with elements or columns labelled according to the nodes of the hierarchy containing values or utilities. Typically, this will be the complete output or an output row of the function
evaluate.utility.aggregation.
This input is only considered if the argument type is specified to be "hierarchy". It is then used to color-code the upper part of the boxes of the hierarchy to allow for a comparison with the results provided by the argument u which are shown in the lower part of the boxes.

par — (optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node. Note that this affects only the node definitions plotted if the argument type is specified to be "node" or "nodes". To color-code hierarchies or tables for different parameter values, the parameters have to be passed to
evaluate.utility.aggregation
before passing the results of this function to this plotting routine.

type — (optional) specifies the type of plot to be produced.
Options: "hierarchy", "table", "node" or "nodes".
"hierarchy": produces a plot of the objectives hierarchy including color-coded results for values or utilities if these values are provided by the arguments u and/or uref.
"table": produces a table with color-coded results for values or utilities if these values are provided by the argument u.
"node": produces a plot of the definition of the current node.
"nodes": produces plots of node definitions for all nodes defined by the attribute nodes.

nodes — (optional) character vector specifying the nodes for which the definitions will be plotted or which will be considered in a table. The default value of NA indicates that all nodes will be plotted. This argument only affects the output if the argument type was indicated to be either "table" or "nodes".

col — (optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to have a non-equidistant color-coding). This attribute is only used for value nodes and if values are provided by the arguments u and/or uref.

gridlines — (optional) numeric vector of levels at which gridlines are plotted in node definitions. This attribute is only used if the argument type is specified to be either "node" or "nodes".

main	(optional) title(s) of the plot. If the argument type is equal to "hierarchy" and the a vector of titles with the same length as the number of rows of the argument u is provided, a color-coded hierarchy is plotted for each row of u. Otherwise, the medians and colored boxes indicating 90% credibility or occurrence ranges are plotted at all nodes.
cex.main	(optional) scaling factor for title of the plot.
cex.nodes	(optional) scaling factor for node labels used in the plot.
cex.attrib	(optional) scaling factor for attribute labels used in the plot.
f.reaches	(optional) fraction of the width of the plot reserved for the row labels of the table if the argument type is equal to "table".
f.nodes	(optional) fraction of the height of the plot reserved for the column labels of the table if the argument type is equal to "table".
with.attrib	(optional) indicates if attributes should be listed if the argument type is equal to "hierarchy".
levels	(optional) how many levels of the hierarchy should be plotted (NA means to plot all levels).
plot.val	(optional) plot value as a vertical line within the box.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines
plot.utility.conversion.intpol
plot.utility.conversion.parfun
are exactly the same so that all hierarchies can be plotted with exactly the same commands irrespective of the type of the top-level node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See utility.aggregation.create for how to construct such a node and evaluate.utility.aggregation for how to evaluate the node.

See utility.calc.colors for an example of how to construct color schemes and utility.get.colors for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

---

plot.utility.conversion.intpol

Plot Node Definition or Underlying Objectives Hierarchy

Description

Plot node definition or underlying objective hierarchy.

Usage

```
## S3 method for class 'utility.conversion.intpol'
plot(x,
     u           = NA,
     uref        = NA,
     par         = NA,
     type        = c("hierarchy", "table", "node", "nodes"),
     nodes       = NA,
     col         = utility.calc.colors(),
     gridlines   = c(0.2, 0.4, 0.6, 0.8),
     main        = "",
     cex.main    = 1,
     cex.nodes   = 1,
     cex.attrib  = 1,
     f.reaches   = 0.2,
     f.nodes     = 0.2,
     with.attrib = TRUE,
     levels      = NA,
     plot.val    = TRUE,
     ...)
```

Arguments

x	node to be plotted.
u	(optional) vector or data frame with elements or columns labelled according to the nodes of the hierarchy containing values or utilities. Typically, this will be the complete output or an output row of the function evaluate.utility.aggregation. This input is only considered if the argument type is specified to be either "hierarchy" or "table". It is then used to color-code the boxes of the hierarchy representing value nodes or the table. If u is a data frame with more than one row and the argument type is equal to "hierarchy", then the median and quantile boxes are plotted for value nodes or the expected utility for utility nodes unless the argument main contains as many elements as the number of rows of u. In the latter case, separate hierarchies with color-coded boxes for value nodes are produced for all rows of u. For type equals "table", this argument can be a list of data frames to make it possible to plot uncertainty ranges from the samples provided in the list.
uref	(optional) vector or data frame with elements or columns labelled according to the nodes of the hierarchy containing values or utilities. Typically, this will be the complete output or an output row of the function evaluate.utility.aggregation. This input is only considered if the argument type is specified to be "hierarchy". It is then used to color-code the upper part of the boxes of the hierarchy to allow for a comparison with the results provided by the argument u which are shown in the lower part of the boxes.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node. Note that this affects only the node definitions plotted if the argument type is specified to be "node" or "nodes". To color-code hierarchies or tables for different parameter values, the parameters have to be passed to evaluate.utility.aggregation before passing the results of this function to this plotting routine.
type	(optional) specifies the type of plot to be produced. Options: "hierarchy", "table", "node" or "nodes". "hierarchy": produces a plot of the objectives hierarchy including color-coded results for values or utilities if these values are provided by the arguments u and/or uref. "table": produces a table with color-coded results for values or utilities if these values are provided by the argument u. "node": produces a plot of the definition of the current node. "nodes": produces plots of node definitions for all nodes defined by the attribute nodes.
nodes	(optional) character vector specifying the nodes for which the definitions will be plotted or which will be considered in a table. The default value of NA indicates that all nodes will be plotted. This argument only affects the output if the argument type was indicated to be either "table" or "nodes".
col	(optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to

	have a non-equidistant color-coding). This attribute is only used for value nodes and if values are provided by the arguments u and/or uref.
gridlines	(optional) numeric vector of levels at which gridlines are plotted in node definitions. This attribute is only used if the argument type is specified to be either "node" or "nodes".
main	(optional) title(s) of the plot. If the argument type is equal to "hierarchy" and the a vector of titles with the same length as the number of rows of the argument u is provided, a color-coded hierarchy is plotted for each row of u. Otherwise, the medians and colored boxes indicating 90% credibility or occurrence ranges are plotted at all nodes.
cex.main	(optional) scaling factor for title of the plot.
cex.nodes	(optional) scaling factor for node labels used in the plot.
cex.attrib	(optional) scaling factor for attribute labels used in the plot.
f.reaches	(optional) fraction of the width of the plot reserved for the row labels of the table if the argument type is equal to "table".
f.nodes	(optional) fraction of the height of the plot reserved for the column labels of the table if the argument type is equal to "table".
with.attrib	(optional) indicates if attributes should be listed if the argument type is equal to "hierarchy".
levels	(optional) how many levels of the hierarchy should be plotted (NA means to plot all levels).
plot.val	(optional) plot value as a vertical line within the box.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines
plot.utility.conversion.parfun
plot.utility.aggregation
are exactly the same so that all hierarchies can be plotted with exactly the same commands irrespective of the type of the top-level node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.conversion.intpol.create` for how to construct such a node and `evaluate.utility.conversion.intpol` for how to evaluate the node.

See `utility.calc.colors` for an example of how to construct color schemes and `utility.get.colors` for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

`plot.utility.conversion.parfun`
Plot Node Definition or Underlying Objectives Hierarchy

Description

Plot node definition or underlying objectives hierarchy.

Usage

```
## S3 method for class 'utility.conversion.parfun'
plot(x,
     u           = NA,
     uref        = NA,
     par         = NA,
     type        = c("hierarchy", "table", "node", "nodes"),
     nodes       = NA,
     col         = utility.calc.colors(),
     gridlines   = c(0.2, 0.4, 0.6, 0.8),
     main        = "",
     cex.main    = 1,
     cex.nodes   = 1,
     cex.attrib  = 1,
     f.reaches   = 0.2,
     f.nodes     = 0.2,
     with.attrib = TRUE,
     levels      = NA,
     plot.val    = TRUE,
     ...)
```

Arguments

x	node to be plotted.
u	(optional) vector or data frame with elements or columns labelled according to the nodes of the hierarchy containing values or utilities. Typically, this will be the complete output or an output row of the function evaluate.utility.aggregation. This input is only considered if the argument type is specified to be either "hierarchy" or "table". It is then used to color-code the boxes of the hierarchy representing value nodes or the table. If u is a data frame with more than one row and the argument type is equal to "hierarchy", then the median and quantile boxes are plotted for value nodes or the expected utility for utility nodes unless the argument main contains as many elements as the number of rows of u. In the latter case, separate hierarchies with color-coded boxes for value nodes are produced for all rows of u. For type equals "table", this argument can be a list of data frames to make it possible to plot uncertainty ranges from the samples provided in the list.
uref	(optional) vector or data frame with elements or columns labelled according to the nodes of the hierarchy containing values or utilities. Typically, this will be the complete output or an output row of the function evaluate.utility.aggregation. This input is only considered if the argument type is specified to be "hierarchy". It is then used to color-code the upper part of the boxes of the hierarchy to allow for a comparison with the results provided by the argument u which are shown in the lower part of the boxes.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node. Note that this affects only the node definitions plotted if the argument type is specified to be "node" or "nodes". To color-code hierarchies or tables for different parameter values, the parameters have to be passed to evaluate.utility.aggregation before passing the results of this function to this plotting routine.
type	(optional) specifies the type of plot to be produced. Options: "hierarchy", "table", "node" or "nodes". "hierarchy": produces a plot of the objectives hierarchy including color-coded results for values or utilities if these values are provided by the arguments u and/or uref. "table": produces a table with color-coded results for values or utilities if these values are provided by the argument u. "node": produces a plot of the definition of the current node. "nodes": produces plots of node definitions for all nodes defined by the attribute nodes.
nodes	(optional) character vector specifying the nodes for which the definitions will be plotted or which will be considered in a table. The default value of NA indicates that all nodes will be plotted. This argument only affects the output if the argument type was indicated to be either "table" or "nodes".
col	(optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to

	have a non-equidistant color-coding). This attribute is only used for value nodes and if values are provided by the arguments u and/or uref.
gridlines	(optional) numeric vector of levels at which gridlines are plotted in node definitions. This attribute is only used if the argument type is specified to be either "node" or "nodes".
main	(optional) title(s) of the plot. If the argument type is equal to "hierarchy" and the a vector of titles with the same length as the number of rows of the argument u is provided, a color-coded hierarchy is plotted for each row of u. Otherwise, the medians and colored boxes indicating 90% credibility or occurrence ranges are plotted at all nodes.
cex.main	(optional) scaling factor for title of the plot.
cex.nodes	(optional) scaling factor for node labels used in the plot.
cex.attrib	(optional) scaling factor for attribute labels used in the plot.
f.reaches	(optional) fraction of the width of the plot reserved for the row labels of the table if the argument type is equal to "table".
f.nodes	(optional) fraction of the height of the plot reserved for the column labels of the table if the argument type is equal to "table".
with.attrib	(optional) indicates if attributes should be listed if the argument type is equal to "hierarchy".
levels	(optional) how many levels of the hierarchy should be plotted (NA means to plot all levels).
plot.val	(optional) plot value as a vertical line within the box.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines
plot.utility.conversion.intpol
plot.utility.aggregation
are exactly the same so that all hierarchies can be plotted with exactly the same commands irrespective of the type of the top-level node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.conversion.parfun.create` for how to construct such a node and `evaluate.utility.conversion.parfun` for how to evaluate the node.

See `utility.calc.colors` for an example of how to construct color schemes and `utility.get.colors` for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

`plot.utility.endnode.cond`

Plot Node Definition

Description

Plot node definition.

Usage

```
## S3 method for class 'utility.endnode.cond'
plot(x,
     par       = NA,
     col       = utility.calc.colors(),
     gridlines = c(0.2, 0.4, 0.6, 0.8),
     main      = "",
     cex.main  = 1,
     nodes     = x$name,
     ...)
```

Arguments

`x`	node to be plotted.
`par`	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node.
`col`	(optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to have a non-equidistant color-coding). This attribute is only used for value nodes.

gridlines	(optional) numeric vector of levels at which gridlines are plotted in the node definition.
main	(optional) title of the plot.
cex.main	(optional) scaling factor for title of the plot.
nodes	(optional) character vector specifying the names of the nodes to be plotted.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines for the other end nodes
plot.utility.endnode.discrete
plot.utility.endnode.intpol1d
plot.utility.endnode.parfun1d
plot.utility.endnode.intpol2d
are as far as possible the same so that all end nodes can be plotted with the same commands irrespective of the type of the end node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See utility.endnode.cond.create for how to construct such a node and evaluate.utility.endnode.cond for how to evaluate the node.

See utility.calc.colors for an example of how to construct color schemes and utility.get.colors for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

plot.utility.endnode.discrete
Plot Node Definition

Description

Plot node definition.

Usage

```
## S3 method for class 'utility.endnode.discrete'
plot(x,
     par       = NA,
     col       = utility.calc.colors(),
     gridlines = c(0.2, 0.4, 0.6, 0.8),
     main      = "",
     cex.main  = 1,
     ...)
```

Arguments

x	node to be plotted.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node.
col	(optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to have a non-equidistant color-coding). This attribute is only used for value nodes.
gridlines	(optional) numeric vector of levels at which gridlines are plotted in the node definition.
main	(optional) title of the plot.
cex.main	(optional) scaling factor for title of the plot.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines for the other end nodes
plot.utility.endnode.intpol1d
plot.utility.endnode.parfun1d
plot.utility.endnode.intpol2d
plot.utility.endnode.cond
are as far as possible the same so that all end nodes can be plotted with the same commands irrespective of the type of the end node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.discrete.create` for how to construct such a node and `evaluate.utility.endnode.discrete` for how to evaluate the node.

See `utility.calc.colors` for an example of how to construct color schemes and `utility.get.colors` for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

`plot.utility.endnode.intpol1d`
Plot Node Definition

Description

Plot node definition.

Usage

```
## S3 method for class 'utility.endnode.intpol1d'
plot(x,
     par       = NA,
     col       = utility.calc.colors(),
     gridlines = c(0.2, 0.4, 0.6, 0.8),
     main      = "",
     cex.main  = 1,
     ...)
```

Arguments

x	node to be plotted.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node.
col	(optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to have a non-equidistant color-coding). This attribute is only used for value nodes.
gridlines	(optional) numeric vector of levels at which gridlines are plotted in the node definition.
main	(optional) title of the plot.
cex.main	(optional) scaling factor for title of the plot.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines for the other end nodes
plot.utility.endnode.discrete
plot.utility.endnode.parfun1d
plot.utility.endnode.intpol2d
plot.utility.endnode.cond
are as far as possible the same so that all end nodes can be plotted with the same commands irrespective of the type of the end node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See utility.endnode.intpol1d.create for how to construct such a node and evaluate.utility.endnode.intpol1d for how to evaluate the node.

See utility.calc.colors for an example of how to construct color schemes and utility.get.colors for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

`plot.utility.endnode.intpol2d` *Plot Node Definition*

Description

Plot node definition.

Usage

```
## S3 method for class 'utility.endnode.intpol2d'
plot(x,
     par       = NA,
     col       = utility.calc.colors(),
     gridlines = c(0.2, 0.4, 0.6, 0.8),
     main      = "",
     cex.main  = 1,
     ...)
```

Arguments

x	node to be plotted.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node.
col	(optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to have a non-equidistant color-coding). This attribute is only used for value nodes.
gridlines	(optional) numeric vector of levels at which gridlines are plotted in the node definition. Not used for this type of node.
main	(optional) title of the plot.
cex.main	(optional) scaling factor for title of the plot.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines for the other end nodes
plot.utility.endnode.discrete
plot.utility.endnode.parfun1d
plot.utility.endnode.intpol2d
plot.utility.endnode.cond
are as far as possible the same so that all end nodes can be plotted with the same commands irrespective of the type of the end node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See utility.endnode.intpol1d.create for how to construct such a node and evaluate.utility.endnode.intpol1d for how to evaluate the node.

See utility.calc.colors for an example of how to construct color schemes and utility.get.colors for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

---

plot.utility.endnode.parfun1d

Plot Node Definition

Description

Plot node definition.

Usage

```
## S3 method for class 'utility.endnode.parfun1d'
plot(x,
     par       = NA,
     col       = utility.calc.colors(),
     gridlines = c(0.2, 0.4, 0.6, 0.8),
     main      = "",
     cex.main  = 1,
     ...)
```

Arguments

x	node to be plotted.
par	(optional) labelled numeric parameter vector providing parameters to modify the value or utility function before plotting the node.
col	(optional) character vector of colors to be used to color the interval between zero and unity in equidistant sections (use repetitions of the same color if you want to have a non-equidistant color-coding). This attribute is only used for value nodes.
gridlines	(optional) numeric vector of levels at which gridlines are plotted in the node definition.
main	(optional) title of the plot.
cex.main	(optional) scaling factor for title of the plot.
...	additional arguments passed to the R plotting routine.

Note

Note that the plotting routines for the other end nodes
plot.utility.endnode.discrete
plot.utility.endnode.intpol1d
plot.utility.endnode.intpol2d
plot.utility.endnode.cond
are as far as possible the same so that all end nodes can be plotted with the same commands irrespective of the type of the end node.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node and `evaluate.utility.endnode.parfun1d` for how to evaluate the node.

See `utility.calc.colors` for an example of how to construct color schemes and `utility.get.colors` for how to get colors for specifed value levels.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.aggregation`

Print Definitions of Node and Associated Hierarchy

Description

Print definition of node and associated hierarchy.

Usage

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

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.aggregation.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.conversion.intpol`
Print Definitions of Node and Associated Hierarchy

Description

Print definition of node and associated hierarchy.

Usage

```
## S3 method for class 'utility.conversion.intpol'
print(x, ...)
```

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.conversion.intpol.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.conversion.parfun`
Print Definitions of Node and Associated Hierarchy

Description

Print definition of node and associated hierarchy.

Usage

```
## S3 method for class 'utility.conversion.parfun'
print(x, ...)
```

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.conversion.parfun.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.endnode.cond`
Print Node Definition

Description

Print node defintion.

Usage

```
## S3 method for class 'utility.endnode.cond'
print(x, ...)
```

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.cond.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.endnode.discrete`
Print Node Definition

Description

Print node defintion.

Usage

```
## S3 method for class 'utility.endnode.discrete'
print(x, ...)
```

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.discrete.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.endnode.intpol1d`
Print Node Definition

Description

Print node defintion.

Usage

```
## S3 method for class 'utility.endnode.intpol1d'
print(x, ...)
```

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.intpol1d.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.endnode.intpol2d`
Print Node Definition

Description

Print node defintion.

Usage

```
## S3 method for class 'utility.endnode.intpol2d'
print(x, ...)
```

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.intpol2d.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`print.utility.endnode.parfun1d`
Print Node Definition

Description

Print node defintion.

Usage

```
## S3 method for class 'utility.endnode.parfun1d'
print(x, ...)
```

Arguments

`x`	node to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.aggregation`
Print Summary of Definitions of Node and Associated Hierarchy

---

Description

Print summary of definition of node and associated hierarchy.

Usage

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

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.aggregation.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.conversion.intpol`

Print Summary of Definitions of Node and Associated Hierarchy

---

Description

Print summary of definition of node and associated hierarchy.

Usage

```
## S3 method for class 'utility.conversion.intpol'
summary(object, ...)
```

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.conversion.intpol.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.conversion.parfun`
Print Summary of Definitions of Node and Associated Hierarchy

---

Description

Print summary of definition of node and associated hierarchy.

Usage

```
## S3 method for class 'utility.conversion.parfun'
summary(object, ...)
```

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.conversion.parfun.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.endnode.cond`

Print Summary of Node Definition

Description

Print summary of node definition.

Usage

```
## S3 method for class 'utility.endnode.cond'
summary(object, ...)
```

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.cond.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.endnode.discrete`

Print Summary of Node Definition

Description

Print summary of node definition.

Usage

```
## S3 method for class 'utility.endnode.discrete'
summary(object, ...)
```

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.discrete.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.endnode.intpol1d`

Print Summary of Node Definition

Description

Print summary of node definition.

Usage

```
## S3 method for class 'utility.endnode.intpol1d'
summary(object, ...)
```

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.intpol1d.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.endnode.intpol2d`

Print Summary of Node Definition

Description

Print summary of node definition.

Usage

```
## S3 method for class 'utility.endnode.intpol2d'
summary(object, ...)
```

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.intpol2d.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`summary.utility.endnode.parfun1d`
Print Summary of Node Definition

Description

Print summary of node definition.

Usage

```
## S3 method for class 'utility.endnode.parfun1d'
summary(object, ...)
```

Arguments

`object`	node of which a summary is to be printed.
`...`	currently no other arguments are implemented or passed further.

Note

In the current version of the package, the methods `print` and `summary` provide the same output.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node.

Examples

```
# see
help(utility)
# for examples.
```

`updatepar` *Update Parameters in Node Definitions*

Description

Generic function to update parameters in all node defintions of the hierarchy defined by the given node.

Usage

```
updatepar(x, ...)
```

Arguments

`x`	node to be updated.
`...`	parameter values can be provided by an additional argument `par`.

Value

The node or node hierarchy with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.parfun1d.create,
utility.endnode.intpol2d.create,
utility.endnode.cond.create,
utility.aggregation.create,
utility.conversion.intpol.create,
utility.conversion.parfun.create

for how to construct the nodes and

updatepar.utility.endnode.discrete
updatepar.utility.endnode.intpol1d
updatepar.utility.endnode.parfun1d
updatepar.utility.endnode.intpol2d
updatepar.utility.endnode.cond
updatepar.utility.aggregation
updatepar.utility.conversion.intpol
updatepar.utility.conversion.parfun
for the updates of the specific nodes.

---

updatepar.utility.aggregation

Update Parameters in Node Definitions

Description

Update parameters in all node defintions of the hierarchy defined by the node.

Usage

```
## S3 method for class 'utility.aggregation'
updatepar(x, par=NA, ...)
```

Arguments

x	node to be updated.
par	parameter vector with labelled parameters to be updated.
...	currently no other arguments are implemented or passed further.

Value

The node hierarchy with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See utility.aggregation.create for how to construct such a node and

updatepar.utility.endnode.discrete
updatepar.utility.endnode.intpol1d
updatepar.utility.endnode.parfun1d
updatepar.utility.endnode.intpol2d
updatepar.utility.endnode.cond
updatepar.utility.conversion.intpol
updatepar.utility.conversion.parfun
for analogous updates of other nodes

`updatepar.utility.conversion.intpol` *Update Parameters in Node Definitions*

Description

Update parameters in all node defintions of the hierarchy defined by the node.

Usage

```
## S3 method for class 'utility.conversion.intpol'
updatepar(x, par=NA, ...)
```

Arguments

`x`	node to be updated.
`par`	parameter vector labelled with parameter values to be updated.
`...`	currently no other arguments are implemented or passed further.

Value

The node hierarchy with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node and

`updatepar.utility.endnode.discrete`
`updatepar.utility.endnode.intpol1d`

updatepar.utility.endnode.parfun1d
updatepar.utility.endnode.intpol2d
updatepar.utility.endnode.cond
updatepar.utility.aggregation
updatepar.utility.conversion.parfun
for analogous updates of other nodes

`updatepar.utility.conversion.parfun` *Update Parameters in Node Definitions*

Description

Update parameters in all node defintions of the hierarchy defined by the node.

Usage

```
## S3 method for class 'utility.conversion.parfun'
updatepar(x, par=NA, ...)
```

Arguments

`x`	node to be updated.
`par`	parameter vector with labelled parameters to be updated.
`...`	currently no other arguments are implemented or passed further.

Value

The node hierarchy with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See utility.endnode.parfun1d.create for how to construct such a node and

updatepar.utility.endnode.discrete
updatepar.utility.endnode.intpol1d
updatepar.utility.endnode.parfun1d
updatepar.utility.endnode.intpol2d
updatepar.utility.endnode.cond
updatepar.utility.aggregation
updatepar.utility.conversion.intpol
for analogous updates of other nodes

---

updatepar.utility.endnode.cond

Update Parameters in Node Definitions

Description

Update parameters in all node defintions used to define the node.

Usage

```
## S3 method for class 'utility.endnode.cond'
updatepar(x, par=NA, ...)
```

Arguments

x	node to be updated.
par	parameter vector with labelled parameters to be updated.
...	currently no other arguments are implemented or passed further.

Value

The node with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node and

`updatepar.utility.endnode.discrete`
`updatepar.utility.endnode.intpol1d`
`updatepar.utility.endnode.parfun1d`
`updatepar.utility.endnode.intpol2d`
`updatepar.utility.aggregation`
`updatepar.utility.conversion.intpol`
`updatepar.utility.conversion.parfun`
for analogous updates of other nodes

`updatepar.utility.endnode.discrete`
Update Parameters in Node Definition

Description

Update parameters in node defintion.

Usage

```
## S3 method for class 'utility.endnode.discrete'
updatepar(x, par=NA, ...)
```

Arguments

`x`	node to be updated.
`par`	parameter vector with labelled parameters to be updated.
`...`	currently no other arguments are implemented or passed further.

Value

The node with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node and

`updatepar.utility.endnode.intpol1d`
`updatepar.utility.endnode.parfun1d`
`updatepar.utility.endnode.intpol2d`
`updatepar.utility.endnode.cond`
`updatepar.utility.aggregation`
`updatepar.utility.conversion.intpol`
`updatepar.utility.conversion.parfun`
for analogous updates of other nodes

---

updatepar.utility.endnode.intpol1d

Update Parameters in Node Definition

Description

Update parameters in node defintion.

Usage

```
## S3 method for class 'utility.endnode.intpol1d'
updatepar(x, par=NA, ...)
```

Arguments

`x`	node to be updated.
`par`	parameter vector with labelled parameters to be updated.
`...`	currently no other arguments are implemented or passed further.

Value

The node with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node and

`updatepar.utility.endnode.discrete`
`updatepar.utility.endnode.parfun1d`
`updatepar.utility.endnode.intpol2d`
`updatepar.utility.endnode.cond`
`updatepar.utility.aggregation`
`updatepar.utility.conversion.intpol`
`updatepar.utility.conversion.parfun`
for analogous updates of other nodes

`updatepar.utility.endnode.intpol2d`
Update Parameters in Node Definition

Description

Update parameters in node defintion.

Usage

```
## S3 method for class 'utility.endnode.intpol2d'
updatepar(x, par=NA, ...)
```

Arguments

x	node to be updated.
par	parameter vector with labelled parameters to be updated.
...	currently no other arguments are implemented or passed further.

Value

The node with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See `utility.endnode.parfun1d.create` for how to construct such a node and

`updatepar.utility.endnode.discrete`
`updatepar.utility.endnode.intpol1d`
`updatepar.utility.endnode.parfun1d`
`updatepar.utility.endnode.cond`
`updatepar.utility.aggregation`
`updatepar.utility.conversion.intpol`
`updatepar.utility.conversion.parfun`
for analogous updates of other nodes

`updatepar.utility.endnode.parfun1d`

Update Parameters in Node Definition

Description

Update parameters in node defintion.

Usage

```
## S3 method for class 'utility.endnode.parfun1d'
updatepar(x, par=NA, ...)
```

Arguments

x	node to be updated.
par	parameter vector with labelled parameters to be updated.
...	currently no other arguments are implemented or passed further.

Value

The node with updated parameters is returned.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See utility.endnode.parfun1d.create for how to construct such a node and

updatepar.utility.endnode.discrete
updatepar.utility.endnode.intpol1d
updatepar.utility.endnode.intpol2d
updatepar.utility.endnode.cond
updatepar.utility.aggregation
updatepar.utility.conversion.intpol
updatepar.utility.conversion.parfun
for analogous updates of other nodes

`utility.aggregate.add` *Additive aggregation of values or utilities*

Description

Function to perform an additive aggregation (weighted mean) of values or utilities.

Usage

```
utility.aggregate.add(u, par)
```

Arguments

`u`	numeric vector of values or utilities to be aggregated.
`par`	numeric vector of weights for calculating the weighted mean of the values provided in the argument `u`. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector `u`, the weights of the non-missing components will be rescaled to sum to unity.

Value

numeric value representing the weighted mean of the components of `u`.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

`utility.aggregation.create`

Alternative aggregation techniques:

utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.add(c(0.2,0.8), par=c(1,1))
```

`utility.aggregate.cobbdouglas`
Cobb-Douglas aggregation of values or utilities

Description

Function to perform a Cobb-Douglas aggregation (weighted geometric mean) of values or utilities.

Usage

```
utility.aggregate.cobbdouglas(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights for calculating the weighted geometric mean of the values provided in the argument u. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Note

This is the same function as utility.aggregate.geo

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.cobbdouglas(c(0.2,0.8), par=c(1,1))
```

`utility.aggregate.geo` *Geometric aggregation of values or utilities*

Description

Function to perform a geometric aggregation (weighted geometric mean) of values or utilities.

Usage

```
utility.aggregate.geo(u, par)
```

Arguments

`u`	numeric vector of values or utilities to be aggregated.
`par`	numeric vector of weights for calculating the weighted geometric mean of the values provided in the argument `u`. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector `u`, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Note

This is the same function as `utility.aggregate.cobbdouglas`

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.geo(c(0.2,0.8), par=c(1,1))
```

---

utility.aggregate.geooff

Geometric aggregation of values or utilities with offset

Description

Function to perform a geometric aggregation (weighted geometric mean) of values or utilities with offset. The offset is added to the arguments and subtracted from the result.

Usage

```
utility.aggregate.geooff(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights appended by an offset for calculating the weighted geometric mean minus an offset of the values provided in the argument u plus the offset. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.geooff(c(0.2,0.8), par=c(1,1))
```

`utility.aggregate.harmo` *Harmonic aggregation of values or utilities*

Description

Function to perform a harmonic aggregation (weighted harmonic mean) of values or utilities.

Usage

```
utility.aggregate.harmo(u, par)
```

Arguments

`u`	numeric vector of values or utilities to be aggregated.
`par`	numeric vector of weights for calculating the weighted harmonic mean of the values provided in the argument `u`. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector `u`, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.harmo(c(0.2,0.8), par=c(1,1))
```

---

utility.aggregate.harmooff

Harmonic aggregation of values or utilities with offset

Description

Function to perform a harmonic aggregation (weighted harmonic mean) of values or utilities with offset. The offset is added to the arguments and subtracted from the result.

Usage

```
utility.aggregate.harmooff(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights appended by an offset for calculating the weighted harmonic mean minus an offset of the values provided in the argument u plus the offset. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.harmooff(c(0.2,0.8), par=c(1,1))
```

`utility.aggregate.max` *Maximum aggregation of values or utilities*

Description

Function to perform a maximum aggregation of values or utilities.

Usage

```
utility.aggregate.max(u, par = NA)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	unused argument used for compatibility with other aggregation techniques that require parameters.

Value

maximum of the components of u.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

`utility.aggregation.create`

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.max(c(0.2,0.8))
```

`utility.aggregate.min` *Minimum aggregation of values or utilities*

Description

Function to perform a minimum aggregation of values or utilities.

Usage

```
utility.aggregate.min(u, par = NA)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	unused argument used for compatibility with other aggregation techniques that require parameters.

Value

minimum of the components of u.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.min(c(0.2,0.8))
```

---

utility.aggregate.mix *Mixed aggregation of values and utilities*

Description

Function to perform a mixed aggregation of values and utilities. The mixture consists of a weighted mean of the additive, minimum and geometric aggregation techniques.

Usage

```
utility.aggregate.mix(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights for calculating the weighted mean of the values provided in the argument u followed by the three weights of the additive, minimum and geometric aggregation techniques. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,

utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult.

Examples

```
utility.aggregate.mix(c(0.2,0.8),par=c(1,1 , 1,0,0))
utility.aggregate.mix(c(0.2,0.8),par=c(1,1 , 0,1,0))
utility.aggregate.mix(c(0.2,0.8),par=c(1,1 , 0,0,1))
utility.aggregate.mix(c(0.2,0.8),par=c(1,1 , 1,1,1))
```

---

utility.aggregate.mult

Multiplicative aggregation of values or utilities

Description

Function to perform a multiplicative aggregation of values or utilities.

Usage

```
utility.aggregate.mult(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights for calculating the multiplicative combination of the values provided in the argument u.

Value

numeric value corresponding to the multiplicative aggregation of the values provided in the vector u.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mix.

Examples

```
utility.aggregate.mult(c(0.2,0.8),par=c(0.3,0.3))
```

`utility.aggregate.revgeo`

Reverse geometric aggregation of values or utilities

Description

Function to perform a reverse geometric aggregation (unity minus the weighted geometric mean of unity minus the arguments) of values or utilities.

Usage

```
utility.aggregate.revgeo(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights for calculating the reverse weighted geometric mean of the values provided in the argument u. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeooff,
utility.aggregate.harmo,

utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.revgeo(c(0.2,0.8), par=c(1,1))
```

`utility.aggregate.revgeooff` *Reverse geometric aggregation of values or utilities with offset*

Description

Function to perform a reverse geometric aggregation (unity minus the weighted geometric mean of unity minus the arguments) of values or utilities with offset.

Usage

```
utility.aggregate.revgeooff(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights for calculating the reverse weighted geometric mean of the values provided in the argument u. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.revharmooff,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.revgeooff(c(0.2,0.8), par=c(1,1))
```

---

utility.aggregate.revharmo

Reverse harmonic aggregation of values or utilities

Description

Function to perform a reverse harmonic aggregation (unity minus the weighted harmonic mean of unity minus the arguments) of values or utilities.

Usage

```
utility.aggregate.revharmo(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights for calculating the reverse weighted harmonic mean of the values provided in the argument u. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmooff,

utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.revharmo(c(0.2,0.8), par=c(1,1))
```

`utility.aggregate.revharmooff`
Reverse harmonic aggregation of values or utilities with offset

---

Description

Function to perform a reverse harmonic aggregation (unity minus the weighted harmonic mean of unity minus the arguments) of values or utilities with offset.

Usage

```
utility.aggregate.revharmooff(u, par)
```

Arguments

u	numeric vector of values or utilities to be aggregated.
par	numeric vector of weights for calculating the reverse weighted harmonic mean of the values provided in the argument u. The weights need not be normalized, they will be normalized before use. In case of missing values in the vector u, the weights of the non-missing components will be rescaled to sum to unity.

Value

The function returns the aggregated value or utility.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Constructor of aggregation node:

utility.aggregation.create

Alternative aggregation techniques:

utility.aggregate.add,
utility.aggregate.min,
utility.aggregate.max,
utility.aggregate.cobbdouglas,
utility.aggregate.geo,
utility.aggregate.geooff,
utility.aggregate.revgeo,
utility.aggregate.revgeooff,
utility.aggregate.harmo,
utility.aggregate.harmooff,
utility.aggregate.revharmo,
utility.aggregate.mult,
utility.aggregate.mix.

Examples

```
utility.aggregate.revharmooff(c(0.2,0.8), par=c(1,1))
```

---

utility.aggregation.create

Construct an aggregation node

Description

Function to construct an aggregation node for value or utlity functions.

Usage

```
utility.aggregation.create(name.node,
                           nodes,
                           name.fun,
                           par,
                           names.par    = rep(NA, length(par)),
                           required     = FALSE,
                           num.required = 1,
                           col          = "black",
                           shift.levels = 0)
```

Arguments

name.node	name of the node to be constructed as a character string.
nodes	list of nodes to be aggregated.
name.fun	name of the function to be used for aggregation. This function must accept the arguments u and par which pass a vector of values or utilities to be aggregated and the parameters of the function, respectively. The function must then return the corresponding aggregated value or utility. Examples of functions provided by the package are utility.aggregate.add for additive aggregation utility.aggregate.min for minimum aggregation utility.aggregate.max for maximum aggregation utility.aggregate.geo or utility.aggregate.cobbdouglas for geometric or Cobb-Douglas aggregation utility.aggregate.geooff for geometric aggregation with offset utility.aggregate.revgeo for reverse geometric aggregation, utility.aggregate.revgeooff for reverse geometric aggregation with offset, utility.aggregate.harmo for harmonic aggregation, utility.aggregate.harmooff for harmonic aggregation with offset, utility.aggregate.revharmo for reverse harmonic aggregation, utility.aggregate.revharmooff for reverse harmonic aggregation with offset, utility.aggregate.mult for multiplicative aggregation utility.aggregate.mix for a mixture of additive, minimum, and geometric aggregation.
par	numeric vector of parameter values to be passed to the function specified unter name.fun.
names.par	(optional) vector of parameter names corresponging to the vector of values specified under par. Only required to provide access to the values through a named parameter vector.
required	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is TRUE, aggregation at the next higher level is not possible if this node returns NA. Default value is FALSE.

num.required	number of lower-level values or utilities that must at least be available to make the evaluation possible.
col	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is "black".
shift.levels	(optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is 0.

Value

The function returns the created object of type utility.aggregation with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

print.utility.aggregation,
summary.utility.aggregation,
evaluate.utility.aggregation and
plot.utility.aggregation.

Create end nodes with

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.intpol2d.create,
utility.endnode.parfun1d.create, or
utility.endnode.cond.create.

Create conversion nodes with

utility.conversion.intpol.create, or
utility.conversion.parfun.create.

Examples

```
# define discrete end node for width variability
# (attribute "widthvariability_class" with levels "high",
# "moderate" and "none")

widthvar <-
  utility.endnode.discrete.create(
    name.node     = "width variability",
    attrib.levels = data.frame(widthvariability_class=
                               c("high","moderate","none")),
    u             = c(1,0.4125,0),
    names.u       = c("u.high","u_moderate","u.none"),
    required      = FALSE,
    utility       = FALSE)

# define 1d interpolation end node for bed modification with
# riprap
# (attribute "bedmodfract_percent" with levels from 0 to 100)

bedmod_riprap <-
  utility.endnode.intpol1d.create(
    name.node   = "bed modification riprap",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,100),
    u           = c(1,0.775,0.5625,0.24),
    required    = FALSE,
    utility     = FALSE)

# define 1d interpolation end node for bed modification with
# other material
# (attribute "bedmodfract_percent" with levels from 0 to 100)

bedmod_other <-
  utility.endnode.intpol1d.create(
    name.node   = "bed modification other",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,100),
    u           = c(1,0.775,0.5625,0),
    required    = FALSE,
    utility     = FALSE)

# define combination end node for bed modification
# (attributes "bedmodtype_class" and "bedmodfract_percent")

bedmod <-
  utility.endnode.cond.create(
```

```
    name.node     = "bed modification",
    attrib.levels = data.frame(bedmodtype_class=
                               c("riprap","other")),
    nodes         = list(bedmod_riprap,bedmod_other),
    required      = FALSE,
    utility       = FALSE)

# define 1d interpolation end node for bank modification with
# permeable material
# (attribute "bankmodfract_percent" with levels from 0 to 100)

bankmod_perm <-
  utility.endnode.intpol1d.create(
    name.node   = "bank modification perm",
    name.attrib = "bankmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,60,100),
    u           = c(1,0.8667,0.675,0.4125,0.24),
    required    = FALSE,
    utility     = FALSE)

# define 1d interpolation end node for bank modification with
# impermeable material
# (attribute "bankmodfract_percent" with levels from 0 to 100)

bankmod_imperm <-
  utility.endnode.intpol1d.create(
    name.node   = "bank modification imperm",
    name.attrib = "bankmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,60,100),
    u           = c(1,0.775,0.5625,0.24,0),
    required    = FALSE,
    utility     = FALSE)

# define combination end node for bank modification
# (attributes "bankmodtype_class" and "bankmodfract_percent")

bankmod <-
  utility.endnode.cond.create(
    name.node     = "bank modification",
    attrib.levels = data.frame(bankmodtype_class=
                               c("perm","imperm")),
    nodes         = list(bankmod_perm,bankmod_imperm),
    required      = FALSE,
    utility       = FALSE)

# define 2d interpolation end node for riparian zone width
# (attributes "riparianzonewidth_m" and "riparianzonewidth_m")

riparzone_width <-
  utility.endnode.intpol2d.create(
    name.node   = "riparian zone width",
```

```
    name.attrib = c("riverbedwidth_m","riparianzonewidth_m"),
    ranges      = list(c(0,16),c(0,30)),
    isolines    = list(list(x=c(0,16),y=c(0,0)),
                       list(x=c(0,2,10,16),y=c(5,5,15,15)),
                       list(x=c(0,16),y=c(15,15)),
                       list(x=c(0,16),y=c(30,30))),
    u           = c(0.0,0.6,1.0,1.0),
    lead         = 1,
    utility      = FALSE)

# define discrete end node for riparian zone vegetation
# (attriute "riparianzoneveg_class" with levels "natural",
# "seminatural" and "artificial")

riparzone_veg <-
  utility.endnode.discrete.create(
    name.node     = "riparian zone veg.",
    attrib.levels = data.frame(riparianzoneveg_class=
                               c("natural","seminatural","artificial")),
    u             = c(1,0.5625,0),
    required      = FALSE,
    utility       = FALSE)

# define aggregation node for riparian zone

riparzone <-
  utility.aggregation.create(
    name.node = "riparian zone",
    nodes     = list(riparzone_width,riparzone_veg),
    name.fun  = "utility.aggregate.cobbdouglas",
    par       = c(1,1),
    required  = FALSE)

# define aggregation node for ecomorphological state

morphol <-
  utility.aggregation.create(
    name.node = "ecomorphology",
    nodes     = list(widthvar,bedmod,bankmod,riparzone),
    name.fun  = "utility.aggregate.mix",
    par       = c(0.25,0.25,0.25,0.25,0,0,1),
    names.par = c("w_widthvar","w_bedmod","w_bankmod","w_riparzone",
                  "w_add","w_min","w_cobbdouglas"),
    required  = TRUE)

# print individual definitions

print(widthvar)
print(bedmod)

# print all definitions

print(morphol)
```

```
# plot objectives hierarchy with attributes

plot(morphol)

# plot individual nodes:

plot(widthvar)
plot(widthvar,par=c(u_moderate=0.2))
plot(bedmod_other)
plot(bankmod)
#plot(riparzone_width)

# plot selected node definitions of a hierarchy

plot(morphol,type="nodes",nodes=c("width variability",
                                  "bed modification other",
                                  "bank modification"))

# evaluate value function for data sets and plot colored hierarchies
# and table

attrib_channelized <- data.frame(widthvariability_class = "none",
                                 bedmodtype_class       = "riprap",
                                 bedmodfract_percent    = 50,
                                 bankmodtype_class      = "imperm",
                                 bankmodfract_percent   = 70,
                                 riverbedwidth_m        = 10,
                                 riparianzonewidth_m    = 5,
                                 riparianzoneveg_class  = "seminatural")
attrib_rehab       <- data.frame(widthvariability_class = "high",
                                 bedmodtype_class       = "riprap",
                                 bedmodfract_percent    = 50,
                                 bankmodtype_class      = "imperm",
                                 bankmodfract_percent   = 20,
                                 riverbedwidth_m        = 15,
                                 riparianzonewidth_m    = 15,
                                 riparianzoneveg_class  = "natural")

res_channelized     <- evaluate(morphol,attrib=attrib_channelized)
res_channelized_add <- evaluate(morphol,attrib=attrib_channelized,
                                par=c(w_add=1,w_min=0,w_cobbdouglas=0))
res_rehab           <- evaluate(morphol,attrib=attrib_rehab)
res_both            <- rbind(res_channelized,res_rehab)
rownames(res_both)  <- c("channelized","rehabilitated")

plot(morphol,u=res_channelized)
plot(morphol,u=res_channelized_add)
plot(morphol,u=res_rehab)
plot(morphol,u=res_rehab,uref=res_channelized)
plot(morphol,u=res_both,type="table")

# consideration of uncertain attribute levels (higher uncertainty for
```

```
# predicted state after rehabilitation than for observed channelized state):

sampsize <- 2000

attrib_channelized_unc <- data.frame(
  widthvariability_class = rep("high",sampsize),
  bedmodtype_class       = rep("riprap",sampsize),
  bedmodfract_percent    = rnorm(sampsize,mean=50,sd=5),
  bankmodtype_class      = rep("imperm",sampsize),
  bankmodfract_percent   = rnorm(sampsize,mean=70,sd=5),
  riverbedwidth_m        = rep(10,sampsize),
  riparianzonewidth_m    = rep(5,sampsize),
  riparianzoneveg_class  = c("seminatural","artificial")[rbinom(sampsize,1,0.5)+1])

attrib_rehab_unc <- data.frame(
  widthvariability_class = c("moderate","high")[rbinom(sampsize,1,0.5)+1],
  bedmodtype_class       = rep("riprap",sampsize),
  bedmodfract_percent    = rnorm(sampsize,mean=50,sd=15),
  bankmodtype_class      = rep("imperm",sampsize),
  bankmodfract_percent   = rnorm(sampsize,mean=20,sd=5),
  riverbedwidth_m        = rnorm(sampsize,mean=10,sd=2),
  riparianzonewidth_m    = rnorm(sampsize,mean=10,sd=2),
  riparianzoneveg_class  = c("natural","seminatural")[rbinom(sampsize,1,0.5)+1])

res_channelized_unc <- evaluate(morphol,attrib=attrib_channelized_unc)
res_rehab_unc       <- evaluate(morphol,attrib=attrib_rehab_unc)

plot(morphol,u=res_channelized_unc)
plot(morphol,u=res_rehab_unc)
plot(morphol,u=res_rehab_unc,uref=res_channelized_unc)
```

`utility.calc.colors` *Color Scheme for Value Functions*

Description

Function to calculate a color scheme for value functions.

Usage

```
utility.calc.colors(n = 5)
```

Arguments

`n`	number of colors.

Details

For `n = 5` this function produces the standard colors red, orange, yellow, green and blue as used in river assessment programs. These colors are provided in a lighter version to improve readability of black text in front of the colored background. For large values of `n` quasi-continuous transitions are defined between these colors. Any other vector of colors can be used by the plotting routines.

Value

Character vector of colors.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See
`plot.utility.endnode.discrete`
`plot.utility.endnode.intpol1d`
`plot.utility.endnode.parfun1d`
`plot.utility.endnode.intpol2d`
`plot.utility.endnode.cond`
`plot.utility.aggregation`
`plot.utility.conversion.intpol`
`plot.utility.conversion.parfun`
for the use of such color vectors in plotting functions and
`utility.get.colors`
for getting colors corresponding to specified values.

Examples

```
utility.calc.colors(5)
utility.calc.colors(100)
```

`utility.conversion.intpol.create` *Construct an interpolation conversion node*

Description

Function to construct a node converting values into utilities by interpolation.

Usage

```
utility.conversion.intpol.create(name.node,
                                 node,
                                 x,
                                 u,
                                 names.x      = rep(NA, length(x)),
                                 names.u      = rep(NA, length(u)),
                                 required     = FALSE,
                                 col          = "black",
                                 shift.levels = 0)
```

Arguments

`name.node`	name of the node to be constructed as a character string.
`node`	value node that is to be converted into a utility node.
`x`	numeric vector of values for which the utility is known.
`u`	numeric vector of utilities corresponding to the values given in the previous argument `x`.
`names.x`	(optional) vector of character strings with names of the components of the numeric vector `x` specified above. Only required to provide access to the values through a named parameter vector.
`names.u`	(optional) vector of character strings with names of the components of the numeric vector `u` specified above. Only required to provide acces through a named parameter vector.
`required`	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is `TRUE`, aggregation at the next higher level is not possible if this node returns NA. Default value is `FALSE`.
`col`	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is `"black"`.
`shift.levels`	(optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is `0`.

Value

The function returns the created object of type `utility.conversion.intpol1` with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

`print.utility.conversion.intpol`,
`summary.utility.conversion.intpol`,
`evaluate.utility.conversion.intpol` and
`plot.utility.conversion.intpol`.

Create other conversion nodes with `utility.conversion.parfun.create`. Create end nodes with

`utility.endnode.discrete.create`,
`utility.endnode.parfun1d.create`,
`utility.endnode.intpol2d.create`,
`utility.endnode.parfun1d.create`, or
`utility.endnode.cond.create`.

Create aggregation nodes with

`utility.aggregation.create`.

`utility.conversion.parfun.create`
Construct a parametric function conversion node

Description

Function to construct a node converting values into utilities by a parametric function.

Usage

```
utility.conversion.parfun.create(name.node,
                                 node,
                                 name.fun,
                                 par,
                                 names.par    = rep(NA, length(par)),
                                 required     = FALSE,
                                 col          = "black",
                                 shift.levels = 0)
```

Arguments

name.node	name of the node to be constructed as a character string.
node	value node that is to be converted into a utility node.
name.fun	name of the parametric function to be evaluated as a character string. The parametric function must have the arguments u and par which pass a vector of values and a vector of parameters to the function, respectively. The function has to return a vector of corresponding utilities.
par	numeric vector of parameter values to be passed to the function specified unter name.fun.
names.par	(optional) vector of parameter names corresponging to the vector of values specified under par. Only required to provide access to the values through a named parameter vector.
required	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is TRUE, aggregation at the next higher level is not possible if this node returns NA. Default value is FALSE.
col	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is "black".
shift.levels	(optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is 0.

Value

The function returns the created object of type utility.conversion.parfun with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

`print.utility.conversion.parfun`,
`summary.utility.conversion.parfun`,
`evaluate.utility.conversion.parfun` and
`plot.utility.conversion.parfun`.

Create other conversion nodes with `utility.conversion.intpol.create`. Create end nodes with

`utility.endnode.discrete.create`,
`utility.endnode.parfun1d.create`,
`utility.endnode.intpol2d.create`,
`utility.endnode.parfun1d.create`, or
`utility.endnode.cond.create`.

Create aggregation nodes with

`utility.aggregation.create`.

`utility.endnode.cond.create`

Construct a conditional end node

Description

Function to construct a node that makes a choice between given end nodes based on the levels of discrete attributes.

Usage

```
utility.endnode.cond.create(name.node,
                            attrib.levels,
                            nodes,
                            utility      = TRUE,
                            required     = FALSE,
                            col          = "black",
                            shift.levels = 0)
```

Arguments

name.node	name of the node to be constructed as a character string.
attrib.levels	data frame with attribute names as column names and all discrete attribute level combinations in the rows. This may be a dependence on any number of attributes. As combinatorics can lead to a very large number of possible combinations, the node should not depend on a too large number of attributes, in particular if each attribute has many different levels expressed by numbers or character strings.
nodes	list of the length of the number of columns of the data frame specifed as argument attrib.levels above containing the nodes to be associated with the attribute level combinations specified in the rows of attrib.levels.
utility	(optional) logical variable indicating if a value function (FALSE) or a utility function (TRUE) is created. Default value is TRUE.
required	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is TRUE, aggregation at the next higher level is not possible if this node returns NA. Default value is FALSE.
col	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is "black".
shift.levels	(optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is 0.

Value

The function returns the created object of type utility.endnode.cond with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

print.utility.endnode.cond,
summary.utility.endnode.cond,
evaluate.utility.endnode.cond and
plot.utility.endnode.cond.

Create other end nodes with

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.parfun1d.create, or
utility.endnode.intpol2d.create,

Create other types of nodes with

utility.aggregation.create,
utility.conversion.intpol.create, or
utility.conversion.parfun.create.

Examples

```
bedmod_riprap <-
  utility.endnode.intpol1d.create(
    name.node   = "bed modification riprap",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,100),
    u           = c(1,0.775,0.5625,0.24),
    required    = FALSE,
    utility     = FALSE)

bedmod_other <-
  utility.endnode.intpol1d.create(
    name.node   = "bed modification other",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,100),
    u           = c(1,0.775,0.5625,0),
    required    = FALSE,
    utility     = FALSE)

bedmod <-
  utility.endnode.cond.create(
    name.node     = "bed modification",
    attrib.levels = data.frame(bedmodtype_class=
                               c("riprap","other")),
    nodes         = list(bedmod_riprap,bedmod_other),
    required      = FALSE,
```

```
      utility       = FALSE)

  print(bedmod)
  plot(bedmod)
```

`utility.endnode.discrete.create`
Construct a discrete value or utility end node

Description

Function to construct a discrete value or utility end node.

Usage

```
utility.endnode.discrete.create(name.node,
                                attrib.levels,
                                u,
                                names.u      = rep(NA, length(u)),
                                utility      = TRUE,
                                required     = FALSE,
                                col          = "black",
                                shift.levels = 0)
```

Arguments

`name.node`	name of the node to be constructed as a character string.
`attrib.levels`	data frame with attribute names as column names and all discrete attribute level combinations in the rows. This may be a dependence on any number of attributes. As combinatorics can lead to a very large number of possible combinations, the node should not depend on a too large number of attributes, in particular if each attribute has many different levels expressed by numbers or character strings.
`u`	numeric vector of the length of the number of columns of the data frame specifed as argument `attrib.levels` above specifying the values or utilities corresponding to the rows of `attrib.levels`.
`names.u`	(optional) vector of character strings with names of the components of the numeric vector `u` specified above. Only required to provide access to the values through a named parameter vector.
`utility`	(optional) logical variable indicating if a value function (`FALSE`) or a utility function (`TRUE`) is created. Default value is `TRUE`.
`required`	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is `TRUE`, aggregation at the next higher level is not possible if this node returns NA. Default value is `FALSE`.
`col`	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is `"black"`.

`shift.levels` (optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is `0`.

Value

The function returns the created object of type `utility.endnode.discrete` with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

`print.utility.endnode.discrete`,
`summary.utility.endnode.discrete`,
`evaluate.utility.endnode.discrete` and
`plot.utility.endnode.discrete`.

Create other end nodes with

`utility.endnode.intpol1d.create`,
`utility.endnode.parfun1d.create`,
`utility.endnode.intpol2d.create`, or
`utility.endnode.cond.create`.

Create other types of nodes with

`utility.aggregation.create`,
`utility.conversion.intpol.create`, or
`utility.conversion.parfun.create`.

Examples

```
widthvar <-
  utility.endnode.discrete.create(
    name.node     = "width variability",
    attrib.levels = data.frame(widthvariability_class=
                               c("high","moderate","none")),
    u             = c(1,0.4125,0),
    names.u       = c("u.high","u_moderate","u.none"),
    required      = FALSE,
    utility       = FALSE)

print(widthvar)
plot(widthvar)
```

---

utility.endnode.intpol1d.create

Construct a single-attribute interpolation end node

Description

Function to construct a single-attribute interpolation end node.

Usage

```
utility.endnode.intpol1d.create(name.node,
                                name.attrib,
                                range,
                                x,
                                u,
                                names.x      = rep(NA, length(x)),
                                names.u      = rep(NA, length(u)),
                                utility      = TRUE,
                                required     = FALSE,
                                col          = "black",
                                shift.levels = 0)
```

Arguments

name.node	name of the node to be constructed as a character string.
name.attrib	name of the attribute on which the value or utility function depends as a character string.
range	numeric vector with two components specifying the minimum and the maximum of the attribute range.
x	numeric vector of attribute values for which the value or utility is known.
u	numeric vector of values or utilities corresponding to the attribute values given in the previous argument x.

`names.x`	(optional) vector of character strings with names of the components of the numeric vector `x` specified above. Only required to provide access to the values through a named parameter vector.
`names.u`	(optional) vector of character strings with names of the components of the numeric vector `u` specified above. Only required to provide acces through a named parameter vector.
`utility`	(optional) logical variable indicating if a value function (`FALSE`) or a utility function (`TRUE`) is created. Default value is `TRUE`.
`required`	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is `TRUE`, aggregation at the next higher level is not possible if this node returns NA. Default value is `FALSE`.
`col`	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is `"black"`.
`shift.levels`	(optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is `0`.

Value

The function returns the created object of type `utility.endnode.intpol1d` with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

`print.utility.endnode.intpol1d`,
`summary.utility.endnode.intpol1d`,
`evaluate.utility.endnode.intpol1d` and
`plot.utility.endnode.intpol1d`.

Create other end nodes with

utility.endnode.discrete.create,
utility.endnode.parfun1d.create,
utility.endnode.intpol2d.create, or
utility.endnode.cond.create.

Create other types of nodes with

utility.aggregation.create,
utility.conversion.intpol.create, or
utility.conversion.parfun.create.

Examples

```
bedmod_other <-
  utility.endnode.intpol1d.create(
    name.node   = "bed modification other",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    x           = c(0,10,30,100),
    u           = c(1,0.775,0.5625,0),
    required    = FALSE,
    utility     = FALSE)

print(bedmod_other)
plot(bedmod_other)
```

---

utility.endnode.intpol2d.create

Construct a two-attribute interpolation end node

Description

Function to construct a two-attribute interpolation end node.

Usage

```
utility.endnode.intpol2d.create(name.node,
                                name.attrib,
                                ranges,
                                isolines,
                                u,
                                names.u      = rep(NA, length(u)),
                                lead         = 0,
                                utility      = TRUE,
                                required     = FALSE,
```

```
col          = "black",
shift.levels = 0)
```

Arguments

name.node	name of the node to be constructed as a character string.
name.attrib	names of the attributes on which the value or utility function depends as a vector of two character strings.
ranges	list of two numeric vectors with two components each specifying the minimum and the maximum of the range of the corresponding attribute.
isolines	list of isoline definitions. Each definition consists of a list with elements x and y that each represents a numeric vector of x- (=first attribute) and y- (second attribute) values to characterize the shape of the isoline.
u	numeric vector of the same length as the outer list of the argument isolines specifying the corresponding values or utilities.
names.u	(optional) vector of character strings with names of the components of the numeric vector u specified above. Only required to provide acces through a named parameter vector.
lead	numeric value specifying which variable is the lead variable for interpolation. 1 indicates linear interpolation between isolines along lines with constant value of the first attribute, 2 along lines with constant values of the second attribute, and zero indicates to take the average of these two interpolation schemes.
utility	(optional) logical variable indicating if a value function (FALSE) or a utility function (TRUE) is created. Default value is TRUE.
required	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is TRUE, aggregation at the next higher level is not possible if this node returns NA. Default value is FALSE.
col	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is "black".
shift.levels	(optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is 0.

Value

The function returns the created object of type utility.endnode.intpol2d with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

print.utility.endnode.intpol2d,
summary.utility.endnode.intpol2d,
evaluate.utility.endnode.intpol2d and
plot.utility.endnode.intpol2d.

Create other end nodes with

utility.endnode.discrete.create,
utility.endnode.intpol1d.create,
utility.endnode.parfun1d.create, or
utility.endnode.cond.create.

Create other types of nodes with

utility.aggregation.create,
utility.conversion.intpol.create, or
utility.conversion.parfun.create.

Examples

```
riparzone_width <-
  utility.endnode.intpol2d.create(
    name.node   = "riparian zone width",
    name.attrib = c("riverbedwidth_m","riparianzonewidth_m"),
    ranges      = list(c(0,16),c(0,30)),
    isolines    = list(list(x=c(0,16),y=c(0,0)),
                       list(x=c(0,2,10,16),y=c(5,5,15,15)),
                       list(x=c(0,16),y=c(15,15)),
                       list(x=c(0,16),y=c(30,30))),
    u           = c(0.0,0.6,1.0,1.0),
    lead         = 1,
    utility      = FALSE)
```

```
print(riparzone_width)
plot(riparzone_width)
```

`utility.endnode.parfun1d.create` *Construct a single-attribute parametric function end node*

Description

Function to construct a single-attribute parametric function end node.

Usage

```
utility.endnode.parfun1d.create(name.node,
                                name.attrib,
                                range,
                                name.fun,
                                par,
                                names.par    = rep(NA, length(par)),
                                utility      = TRUE,
                                required     = FALSE,
                                col          = "black",
                                shift.levels = 0)
```

Arguments

`name.node`	name of the node to be constructed as a character string.
`name.attrib`	name of the attribute on which the value or utility function depends as a character string.
`range`	numeric vector with two components specifying the minimum and the maximum of the attribute range.
`name.fun`	name of the parametric function to be evaluated as a character string. The parametric function must have the arguments `attrib` and `par` which pass a vector of attribute levels and a vector of parameters to the function, respectively. The function has to return a vector of corresponding values or utilities.
`par`	numeric vector of parameter values to be passed to the function specified unter `name.fun`.
`names.par`	(optional) vector of parameter names corresponging to the vector of values specified under `par`. Only required to provide access to the values through a named parameter vector.
`utility`	(optional) logical variable indicating if a value function (`FALSE`) or a utility function (`TRUE`) is created. Default value is `TRUE`.
`required`	(optional) logical variable indicating if the value of this node is required for aggregation at the next higher level. If this variable is `TRUE`, aggregation at the next higher level is not possible if this node returns NA. Default value is `FALSE`.

`col`	(optional) color used for plotting the bounding box of the node in the objective hierarchy. Default value is `"black"`.
`shift.levels`	(optional) number of hierarchical levels by which the node in the objective hierarchy is shifted to make a branch fit better to other branches. Default value is `0`.

Value

The function returns the created object of type `utility.endnode.parfun1d` with the properties specified in the arguments of the function.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Tradeoffs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

Print, evaluate and plot the node with

`print.utility.endnode.parfun1d`,
`summary.utility.endnode.parfun1d`,
`evaluate.utility.endnode.parfun1d` and
`plot.utility.endnode.parfun1d`.

Create other end nodes with

`utility.endnode.discrete.create`,
`utility.endnode.intpol1d.create`,
`utility.endnode.intpol2d.create`, or
`utility.endnode.cond.create`.

Create other types of nodes with

`utility.aggregation.create`,

utility.conversion.intpol.create, or
utility.conversion.parfun.create.

Examples

```
bedmod_other <-
  utility.endnode.parfun1d.create(
    name.node   = "bed modification other",
    name.attrib = "bedmodfract_percent",
    range       = c(0,100),
    name.fun    = "utility.fun.exp",
    par         = c(-1,100,0),
    required    = FALSE,
    utility     = FALSE)

print(bedmod_other)
plot(bedmod_other)
```

---

utility.fun.exp *Exponential function for value or utility functions*

Description

Exponential function for value or utility functions.

Usage

```
utility.fun.exp(attrib, par)
```

Arguments

attrib	vector of attribute levels to calculate corresponding value or utility.
par	Vector of parameters: par[1]: absolute risk aversion par[2]: minimum of attribute range (default = 0) par[3]: maximum of attribute range (default = 1)

Details

The function evaluates the expression
(1-exp(-par[1]*(a-par[2])/(par[3]-par[2])))/(1-exp(-par[1])).

Value

Vector of values or utilities corresponding to the attributes passed by argument a

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See the node constructors
`utility.endnode.intpol1d.create` and `utility.conversion.intpol.create`
in which this function can be used.

Examples

```
utility.fun.exp(0:10/10,par=c(2,0,1))
```

---

utility.get.colors *Get Color Corresponding to Specified Value Levels*

Description

Function to get the colors from a given color scheme at specific value levels.

Usage

```
utility.get.colors(u,col=utility.calc.colors())
```

Arguments

u	value level representing the evaluation ot a value function (this value level has to be between zero and unity).
col	color scheme (vector of colors to be used for a division of the interval between zero and unity into equal intervals)).

Value

Character vector of colors.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

See Also

See
utility.calc.colors

Examples

```
utility.get.colors(c(0,0.5,1))
```

---

utility.structure *Extract Structure of Objectives Hierarchy*

Description

Function to extract the structure of an objectives hierarchy.

Usage

```
utility.structure(node)
```

Arguments

node	object containing the utility or value function.

Value

Data frame containing structural information of the objectives hierarchy.

Author(s)

Peter Reichert <peter.reichert@eawag.ch>

References

Short description of the package:

Reichert, P., Schuwirth, N. and Langhans, S., Constructing, evaluating and visualizing value and utility functions for decision support, Environmental Modelling & Software 46, 283-291, 2013.

Textbooks on the use of utility and value functions in decision analysis:

Keeney, R. L. and Raiffa, H. Decisions with Multiple Objectives - Preferences and Value Trade-offs. John Wiley & Sons, 1976.

Eisenfuehr, F., Weber, M. and Langer, T., Rational Decision Making, Springer, Berlin, 2010.

Index