Package 'cogmapr'

May 8, 2026

Title Cognitive Mapping Tools Based on Coding of Textual Sources

Version 0.9.5

Date 2026-01-12

Description Functions for building cognitive maps based on qualitative data. Inputs are textual sources (articles, transcription of qualitative interviews of agents,...). These sources have been coded using relations and are linked to (i) a table describing the variables (or concepts) used for the coding and (ii) a table describing the sources (typology of agents, ...). Main outputs are Individual Cognitive Maps (ICM), Social Cognitive Maps (all sources or group of sources) and a list of quotes linked to relations. This package is linked to the work done during the PhD of Frederic M. Vanwindekens (CRA-W / UCL) hold the 13 of May 2014 at University of Louvain in collaboration with the Walloon Agricultural Research Centre (project MIMOSA, MOERMAN fund).

URL https://frdvnw.gitlab.io/cogmapr/

BugReports https://gitlab.com/FrdVnW/cogmapr/-/issues

License GPL-3

Language en-GB

SystemRequirements libcurl: libcurl-devel (rpm) or libcurl4-openssl-dev (deb).

Encoding UTF-8

RoxygenNote 7.3.3

Imports grDevices, methods, utils, grid, graph, Rgraphviz, ggplot2, magrittr, dplyr (>= 0.8.0.1), tidyr

biocViews graph, Rgraphviz

Suggests testthat, car

Maintainer Frédéric M. Vanwindekens <FrdVnW@proton.me>

NeedsCompilation no

Author Frédéric M. Vanwindekens [aut, cre] (ORCID: <https://orcid.org/0000-0002-9117-7543>), Didier Stilmant [aut, ths], Philippe V. Baret [aut, ths]

Repository CRAN

Date/Publication 2026-01-26 13:00:02 UTC

Contents

`ConceptCentrality` *Centralities of concepts*

Description

Compute the centrality of concepts

Usage

```
ConceptCentrality(project, filters = NULL, units = "all", weighted.icm = FALSE)
```

Arguments

`project`	A QDA project, a list as generated by the ProjectCMap function.
`filters`	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
`units`	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
`weighted.icm`	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Compute the centrality of concepts

Value

A data frame with the value of the centrality (n) of vertices.

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptCentrality(my.project)
```

ConceptCentralityIndiv

Centrality of vertices by document

Description

Centrality of vertices by document

Usage

```
ConceptCentralityIndiv(project, min.weight = 1, weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
min.weight	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Centrality of vertices by document

Value

A data frame of Centrality by document (ICM)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptCentralityIndiv(my.project)
```

ConceptIndegree *Indegrees of concepts*

Description

Compute the indegree of concepts

Usage

```
ConceptIndegree(project, filters = NULL, units = "all", weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
filters	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Compute the indegree of concepts

Value

A data frame with the value of the indegree (n) of vertices.

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptIndegree(my.project)
```

ConceptIndegreeIndiv *Indegree of vertices by document*

Description

Indegree of vertices by document

Usage

```
ConceptIndegreeIndiv(project, min.weight = 1, weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
min.weight	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Indegree of vertices by document

Value

A data frame of Indegree by document (ICM)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptIndegreeIndiv(my.project)
```

ConceptIndicators *Concepts indicators of a Social Cognitive Map*

Description

Compute the indicators of concepts of a Social Cognitive Map

Usage

```
ConceptIndicators(project, filters = NULL, units = "all", weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
filters	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Compute the indicators of concepts of a Social Cognitive Map (centrality, indegree, outdegree). It build a user friendly data frame. It includes the 'receiver' and the transmitter character of each vertex. The receiver character of a concept is calculated as the part of the indegree of this concept on its centrality. The transmitter character of a concept is calculated as the part of the outdegree of this concept on its centrality.

(add formulae)

Value

A data frame with the value of some indicators linked to vertices of a map.

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptIndicators(my.project)
```

ConceptIndicatorsICM *Concept Indicators of vertices by document (tidy data)*

Description

Concept Indicators of vertices by document (tidy data)

Usage

```
ConceptIndicatorsICM(project, doc_id = "all")
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
doc_id	"all" (default) or the ID of document to include.

Details

Concept Indicators of vertices by document (tidy data)

Value

A data frame (tidy data) with all indicators, their values by document (ICM)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptIndicatorsICM(my.project)
ConceptIndicatorsICM(my.project, 2)
```

ConceptIndicIndiv *Concept Indicators of vertices by document*

Description

Concept Indicators of vertices by document

Usage

```
ConceptIndicIndiv(project, min.weight = 1, weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
min.weight	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Concept Indicators of vertices by document

Value

A data frame of Concept Indicators by document (ICM)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptIndicIndiv(my.project)
```

`ConceptIndicSummary` *Summary table on concept comparisons by indicator*

Description

Summary table on concept comparisons by indicator

Usage

```
ConceptIndicSummary(project, units)
```

Arguments

`project`	A QDA project, a list as generated by the ProjectCMap function.
`units`	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.

Details

This function produce a summary table based on concept comparisons by indicator.

Value

A data frame

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

## More documents are needed for running this function
## ConceptIndicSummary(my.project, units = c("Belgium", "Québec"))
```

ConceptOutdegree *Compute the outdegree of concepts*

Description

Compute the outdegree of concepts## ' @title Outdegrees of concepts

Usage

```
ConceptOutdegree(project, filters = NULL, units = "all", weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
filters	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Value

A data frame with the value of the outdegree (n) of vertices.

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptOutdegree(my.project)
```

ConceptOutdegreeIndiv *Outdegree of vertices by document*

Description

Outdegree of vertices by document

Usage

```
ConceptOutdegreeIndiv(project, min.weight = 1, weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
min.weight	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Outdegree of vertices by document

Value

A data frame of Outdegree by document (ICM)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ConceptOutdegreeIndiv(my.project)
```

ConceptTest *Compare value of concepts indicators between maps*

Description

This function test the differences between the properties of concepts

Usage

```
ConceptTest(project, units, output = "p.value", sep = ">", coder = "qcoder")
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
units	The units to compare
output	"p.value" (default) or "raw.data".
sep	Separation used in the relationships definition. Default is ">" (ex : 1>3)
coder	Coding tool used for this project. Default is "qcoder" (only implemented now)

Details

This function test the differences between the properties of concepts (indegree, outdegree, centrality) between groups of documents (i.e. between social cognitive maps). Till now, only two excluding groups can be tested (ex. document from one country vs another country, from a group of players vs another group of players). It is not possible to compare non exclusive groups (ex. map from one country vs map from one group of players, as some documents can be in the two groups!). For this test, the 'wilcoxon.test' is done. If output = 'p.value', the function returns the results of the tests, one test for each concepts of the map. If output = 'raw.data', the function returns the raw data on which the tests are done, one data frame by concept. This option can be used to export data and perform other statistical tests.

Value

A data frame (if output = "p.value"), a list of data frame (if output = "raw.data").

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

## need more documents
ConceptTest(my.project, units = c("Belgium", "Québec"))
```

ConceptTestSummary *Summary table on concept comparisons*

Description

Summary table on concept comparisons

Usage

```
ConceptTestSummary(project, units, limit.p.value = 0.05)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
units	The units to compare
limit.p.value	A numeric.

Details

This function produce a summary table based on concept comparisons and is reactive to a limit of p.value beyond which differences are considered as significant and are reported in the table

Value

A data frame

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

## More documents are needed for running this function
## ConceptTestSummary(my.project, units = c("Belgium", "Québec"), 0.6)
```

---

coordCMap *Coordinates of the vertices of a Cognitive Map*

Description

Get the coordinates of the vertices of a Cognitive Map. The output of this function can be useful for plotting Cognitive Maps in a personalize ways (as with ggplot2 as done by the ggCMap function of this package)

Usage

```
coordCMap(soc.cmap, layoutType = "neato")
```

Arguments

soc.cmap	An object of class SocCMap, as an output of the SocCMap function
layoutType	Type of graph. See detail in RGraphViz. Can be 'neato', 'dot', 'twopi', 'circo', and 'fdp'. The default is 'neato'.

Value

A data frame with three variable :

$vertex The number of the vertex)

$x The x coordinate of the vertex

$y The y coordinate of the vertex

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

edg.scm <- EdgSocCMap(my.project, min.weight=6, weighted.icm=TRUE)

scm <- SocCMap(edg.scm, my.project, label = "name", shape = "plaintext")
coordCMap(scm)
```

`data.ggCMap` *Data collection for ggCMap*

Description

Get all important data for plotting a Cognitive Map in ggplot

Usage

```
data.ggCMap(
  project,
  min.weight = 1,
  filters = NULL,
  units = "all",
  weighted.icm = FALSE,
  label = "name",
  minlen = 1,
  fontsize = 16,
  shape = "box",
  layoutType = "neato",
  vertex.filter = NULL,
  edge.filter = NULL,
  limit.to.filters = FALSE,
  level = 0
)
```

Arguments

`project`	A QDA project, a list as generated by the ProjectCMap function.
`min.weight`	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
`filters`	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
`units`	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
`weighted.icm`	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.
`label`	A character string that defines the text that will be print in the variables (vertex) of the cognitive maps. It can be "num", "name" or "numname" (which is of the form "NUM - Name"). The default is "num"
`minlen`	A graphical parameter that defines a relative length between the variable of the cognitive maps. See help from RGraphViz package.
`fontsize`	The fontsize of vertices (concepts), in r-base plot

shape	The shape of the vertices (concepts), in r-base plot
layoutType	Type of graph. See detail in RGraphViz. Can be 'neato', 'dot', 'twopi', 'circo', and 'fdp'. The default is 'neato'.
vertex.filter	A vector of integers or characters given the 'id' of vertices (concepts) that will be included in the map. By default, all vertices are included (vertex.filter = NULL)
edge.filter	A vector of characters given the name "i~j" of edges (relationships from "i" to "j") that will be included in the map. By default, all edges are included (edge.filter = NULL)
limit.to.filters	A logical that will impact the position of the vertices. FALSE (the default) will filter vertices and edges (vertex.filter, edge.filter) keeping the position they would have in the unfiltered cognitive map (interesting with background). TRUE will fully re-compute the position of the vertices, building a cognitive map in its own (better readability).
level	0 or 1. Filter the edge/vertices at x level around the filtered edges/vertices (==Not implemented yet==)

Details

Get all important data for plotting a Cognitive Map in ggplot

Value

A list of two data frames : -edges -vertex. In each of these data frames, the main columns are linked to the coordinates of vertex (x, y, x.from, y.from, x.to, y.to)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

data.ggCMap(my.project)
data.ggCMap(my.project, min.weight = 3)
data.ggCMap(my.project, edge.filter = "4")
data.ggCMap(my.project, units = "Belgium")
```

---

df.quotes.scm.concept *Extract the quotes of a project linked to selected concepts*

Description

Extract the quotes of a project linked to selected concepts

Usage

```
df.quotes.scm.concept(project, units, selected.concept)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
selected.concept	A vector of character/integer value(s), the id(s) of one or many concept(s) of the map

Details

Extract the quotes of a project linked to selected concepts. Units is a mandatory parameter as this function was initially developed for given the quotes linked to significantly different concepts between groups of documents (i.e. units).

Value

A data frame of relationships and quotes

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

df.quotes.scm.concept(my.project, "Québec", 4)
df.quotes.scm.concept(my.project, "Québec", "2")
```

---

df.quotes.scm.edge *Extract the quotes of a project linked to selected relationships*

Description

Extract the quotes of a project linked to selected relationships

Usage

```
df.quotes.scm.edge(project, units, selected.edge)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
selected.edge	A vector of character value(s), the names of one or many relationship(s) of the map

Details

Extract the quotes of a project linked to selected relationships. Units is a mandatory parameter as this function was initially developed for given the quotes linked to significantly different relationships between groups of documents (i.e. units).

Value

A data frame of relationships and quotes

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

df.quotes.scm.edge(my.project, "Belgium", "1~2")
```

EdgCMap *Extract all edges from a Qualitative Data Analysis project*

Description

This function opens a Qualitative Data Analysis (QDA) project and extracts edge information.

Usage

```
EdgCMap(project, sep = ">", coder = "qcoder")
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
sep	(==Depreciated, with RQDA project==) A character string (often a single character) that is used in RQDA in order to express the relationships between two variables. Default is "_" if codes used in RQDA are of the form : "x_y" (*i.e.* relationship from x to y), but it is possible to use ">", "->", "–>" or even "—>"
coder	A character string indicating the coding tool used for coding the QDA. The only tool supported now is 'qcoder'. Earlier version of cogmapr worked with "RQDA" project (no more maintained).

Details

The coding used in the QDA have to be done using the 'cogmap-dev' branch of the qcoder package (github : 'FrdVnW/qcoder'). devtools::install_github('FrdVnW/qcoder', ref = "cogmap-dev", upgrade = 'never')

Value

A data.frame with the relationships identified in the interviews. Each relationships ('coding_id') is linked to an agent ('doc_id'), an edge's name ('edge'), the variable at the origin of the relationships ('concept_from'), the variable at the end of the relationships ('concept_to'), other properties of the relationships ('coding_sign', 'coding_weight', 'coding_class' and 'document_part') and the quotes linked to relationships ('selected_text').

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

EdgCMap(my.project)
```

EdgIndCMap *Extract all edges from a Qualitative Data Analysis project for individual cognitive mapping*

Description

This function opens a Qualitative Data Analysis (QDA) project and extracts edge information for individual cognitive mapping

Usage

```
EdgIndCMap(project, min.weight = 1, weighted.icm = FALSE)
```

Arguments

`project`	A QDA project, a list as generated by the ProjectCMap function.
`min.weight`	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
`weighted.icm`	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

The coding used in the QDA have to be done using the 'cogmap-dev' branch of the qcoder package (github : 'FrdVnW/qcoder'). devtools::install_github('FrdVnW/qcoder', ref = "cogmap-dev", upgrade = 'never')

Value

A data.frame

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

EdgIndCMap(my.project)
```

EdgSocCMap *Aggregation of the relationships identified in a serie of Individual Cognitive Maps*

Description

This function will produce a data frame that contains all relationships of a serie of Individual Cognitive Maps. The weights of these relationships are calculated.

Usage

```
EdgSocCMap(
  project,
  min.weight = 1,
  filters = NULL,
  units = "all",
  weighted.icm = FALSE
)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
min.weight	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
filters	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

The function can be used to produce a data frame that contains only the relation with a minimum weight or that concerns only a type of agents.

Value

A data frame with four or five variables : ##'

$edge The name of the relationship, of the generic form *x~y*

$FACTOR If used, the factor the parameter *variable*]. This variable contains then the levels of the factor defined in the parameter *group* used as subset criteria

$weight The weight of each relationship

$from The number of the vertex at the origin of the relationship

$to The number of the vertex at the end of the relationship

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

EdgSocCMap(my.project)
EdgSocCMap(my.project)
EdgSocCMap(my.project, min.weight=3)
EdgSocCMap(my.project, min.weight=6, weighted.icm=TRUE)
```

---

ggCMap *Plot a social cognitive map using ggplot2*

Description

Plotting the a social cognitive map using ggplot2

Usage

```
ggCMap(
  data,
  size.concepts = 4,
  size.labels = 4,
  size.edges = 4,
  size.arrows = 4,
  alpha.arrows = 0.3
)
```

Arguments

`data`	A list, the output of the 'data.ggCMap' function, containing all useful vertex and edge information for the cognitive maps.
`size.concepts`	Size of the dot linked to vertices
`size.labels`	Size of the labels of vertices
`size.edges`	Size of the labels of the weight of edges
`size.arrows`	Size of arrows (head)
`alpha.arrows`	The transparency of arrows.

Details

Plotting the a social cognitive map using ggplot2

Value

A ggplot of a social cognitive map

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

df.scm <- data.ggCMap(my.project, edge.filter = "4")
ggCMap(df.scm)
```

---

ggCMap.bg *Ghost cognitive map*

Description

ggplot background of a map (used in highlighted maps)

Usage

```
ggCMap.bg(
  data,
  size.concepts = 4,
  size.labels = 4,
  size.edges = 4,
  size.arrows = 4,
  map.color = "grey50"
)
```

Arguments

data	A list, the output of the 'data.ggCMap' function, containing all useful vertex and edge information for the cognitive maps.
size.concepts	Size of the dot linked to vertices
size.labels	Size of the labels of vertices
size.edges	Size of the labels of the weight of edges
size.arrows	Size of arrows (head)
map.color	The unique color of all concepts and labels of the ghost map

Details

ggplot background of a map (used in highlighted maps). It is like a ghost map.

Value

A plot

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

df.scm <- data.ggCMap(my.project, edge.filter = "4")
ggCMap.bg(df.scm)
```

---

ggCMap.hl *Highlighted cognitive map*

Description

A cognitive map where some concepts or relationships are highlighted

Usage

```
ggCMap.hl(
  project,
  min.weight = 1,
  filters = NULL,
  units = "all",
  weighted.icm = FALSE,
  label = "name",
  minlen = 1,
  fontsize = 16,
  shape = "box",
  layoutType = "neato",
  vertex.filter = NULL,
  edge.filter = NULL,
  limit.to.filters = FALSE,
  level = 0,
  size.concepts = 4,
  size.labels = 4,
  size.edges = 4,
  size.arrows = 4,
  alpha.arrows = 0.3,
  map.color = grDevices::grey(0.8)
)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
min.weight	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
filters	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.
label	A character string that defines the text that will be print in the variables (vertex) of the cognitive maps. It can be "num", "name" or "numname" (which is of the form "NUM - Name"). The default is "num"
minlen	A graphical parameter that defines a relative length between the variable of the cognitive maps. See help from RGraphViz package.
fontsize	The fontsize of vertices (concepts), in r-base plot
shape	The shape of the vertices (concepts), in r-base plot
layoutType	Type of graph. See detail in RGraphViz. Can be 'neato', 'dot', 'twopi', 'circo', and 'fdp'. The default is 'neato'.
vertex.filter	A vector of integers or characters given the 'id' of vertices (concepts) that will be included in the map. By default, all vertices are included (vertex.filter = NULL)
edge.filter	A vector of characters given the name "i~j" of edges (relationships from "i" to "j") that will be included in the map. By default, all edges are included (edge.filter = NULL)
limit.to.filters	A logical that will impact the position of the vertices. FALSE (the default) will filter vertices and edges (vertex.filter, edge.filter) keeping the position they would have in the unfiltered cognitive map (interesting with background). TRUE will fully re-compute the position of the vertices, building a cognitive map in its own (better readability).
level	0 or 1. Filter the edge/vertices at x level around the filtered edges/vertices (==Not implemented yet==)
size.concepts	Size of the dot linked to vertices
size.labels	Size of the labels of vertices
size.edges	Size of the labels of the weight of edges
size.arrows	Size of arrows (head)
alpha.arrows	The transparency of arrows.
map.color	The unique color of all concepts and labels of the ghost map

Details

A cognitive map where some concepts or relationships are highlighted. The highlighted elements are those who are filtered (see edge.filter, vertex.filter). A background (ghost map) will be show by default. The parameter 'limit.to.filters' can be set as 'TRUE' for only showing the filtered elements.

Value

A plot

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

ggCMap.hl(my.project, vertex.filter = 2)
```

GraphIndicators *Graph indicators of a social cognitive map*

Description

Compute the graph indicators of a Social Cognitive Map (at graph level)

Usage

```
GraphIndicators(project, filters = NULL, units = "all", weighted.icm = FALSE)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
filters	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.
weighted.icm	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.

Details

Compute some indicators from the graph theory and applies them to a Social Cognitive Map :

- dimension : the number of vertices
- n_transmitter : the number of transmitter vertices
- n_receiver : the number of receiver vertices

- n_ordinary : the number of ordinary vertices (transmitter & receiver)
- connections : the number of edges
- density : ...
- complexity_a : ....
- complexity_b : ...
- hierarchy : ...

(== add formulae ==)

Source : Oezesmi & Oezesmi, 2004

Value

A data frame with the value of some indicators linked to the map

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

GraphIndicators(my.project)
GraphIndicators(my.project, units = "Belgium")
GraphIndicators(my.project, units = "Québec")
```

GraphIndicatorsICM *Graph Indicators of vertices by document (tidy data)*

Description

Graph Indicators of vertices by document (tidy data)

Usage

```
GraphIndicatorsICM(project, doc_id = "all")
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
doc_id	"all" (default) or the ID of document to include.

Details

Graph Indicators of vertices by document (tidy data)

Value

A data frame (tidy data) with all indicators, their values by document (ICM)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

GraphIndicatorsICM(my.project)
GraphIndicatorsICM(my.project, 2)
```

`GraphIndicatorsTable` *Table of graph indicators of a social cognitive map*

Description

Table of graph indicators of a social cognitive map

Usage

```
GraphIndicatorsTable(df.graph.indic)
```

Arguments

`df.graph.indic` A data frame, as the output of the function 'GraphIndicators'

Details

Table of graph indicators of a social cognitive map

Value

A data frame of graph theory indicator, easier to read (long format)

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

df.graph.indic <- GraphIndicators(my.project)
GraphIndicatorsTable(df.graph.indic)
```

IndCMap *Individual Cognitive Mapping*

Description

Formatting the data for plotting an Individual Cognitive Map

Usage

```
IndCMap(project, doc.id)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
doc.id	The id of a document

Value

a 'IndCMap' object, a list containing various information that could be use for plotting an Individual Cognitive Map. The most important elements are :

"vertex" A list of information on Cognitive Map's variables (i.e. vertices or concepts)

"edg" A list of information about relationships

"graph" A graphNEL object

"eAttrs" A list of graphical attributes of edges

"nAttrs" A list of graphical attributes of nodes (vertices)

"gAttrs" A list of graphical attributes of the whole graph

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

IndCMap(my.project, 1)
```

plotIndCMap *Plotting an Individual Cognitive Map*

Description

Plotting an Individual Cognitive Map

Usage

```
plotIndCMap(
  ind.cmap,
  layoutType = "neato",
  main = paste("Individual map -", ind.cmap[["agent"]][["name"]]),
  ...
)
```

Arguments

ind.cmap	An object of class IndCMap, as an output of the IndCMap function
layoutType	Type of graph. See detail in RGraphViz. Can be 'neato', 'dot', 'twopi', 'circo', and 'fdp'. The default is 'neato'.
main	The title of the map. By default it is "Individual map - Agent's name"
...	other graphical parameters

Value

A plot

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

icm <- IndCMap(my.project, 1)

plotIndCMap(icm)
```

plotSocCMap *Plotting a Social Cognitive Map*

Description

Plotting a Social Cognitive Map

Usage

```
plotSocCMap(soc.cmap, layoutType = "neato", ..., main = "Social map")
```

Arguments

soc.cmap	An object of class SocCMap, as an output of the SocCMap function
layoutType	Type of graph. See detail in RGraphViz. Can be 'neato', 'dot', 'twopi', 'circo', and 'fdp'. The default is 'neato'.
...	other graphical parameters
main	The title of the map. By default it is "Individual map - Agent's name"

Value

A plot

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

edg.scm <- EdgSocCMap(my.project, min.weight=6, weighted.icm=TRUE)
scm <- SocCMap(edg.scm, my.project)
plotSocCMap(scm)
scm <- SocCMap(edg.scm, my.project, label = "name", shape = "plaintext")
plotSocCMap(scm)
```

ProjectCMap *Import a Qualitative Data Analysis project*

Description

This function import and format the data from a coding tool in a Qualitative Data Analysis.

Usage

```
ProjectCMap(main_path, project_name, coder = "qcoder", sep = ">")
```

Arguments

main_path	The main to your QDA project. It must ended with '/'
project_name	The name of your project (as defined in 'qcoder', the name of the subfolder in your main_path). This name is also used in some filenames in the data_frame subsubfolders.
coder	A character string indicating the coding tool used for coding the QDA. The only tool supported now is 'qcoder'. Earlier version of cogmapr worked with "RQDA" project (no more maintained).
sep	(==Depreciated, with RQDA project==) A character string (often a single character) that is used in RQDA in order to express the relationships between two variables. Default is "_" if codes used in RQDA are of the form : "x_y" (*i.e.* relationship from x to y), but it is possible to use ">", "->", "–>" or even "—>"

Details

This function import and format the data from a coding tool in a Qualitative Data Analysis (QDA).

Value

A list

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)
```

QuotesCMap *Extract all quotes of a QDA project*

Description

Extract all quotes of a QDA project

Usage

```
QuotesCMap(project)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.

Details

This function creates a data frame with all quotes of a Qualitative Data Analysis (QDA) project

Value

A data frame with relationships and quotes

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

QuotesCMap(my.project)
```

QuotesIndCMap *Extract all quotes of a document (or an Individual Cognitive Map)*

Description

Extract all quotes of a document

Usage

```
QuotesIndCMap(project, doc.id)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
doc.id	The id of a document (id of documents can be found in the data frame "documents" in the QDA project)

Details

This function creates a data frame with all quotes of a one document of a Qualitative Data Analysis (QDA) project

Value

A data frame with relationships and quotes

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

QuotesIndCMap(my.project, 2)
```

QuotesSocCMap *Extract all quotes of documents (or a Social Cognitive Map)*

Description

Extract all quotes of a group of documents (or of an Social Cognitive Map)

Usage

```
QuotesSocCMap(project, min.weight = 1, filters = NULL, units = "all")
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
min.weight	A integer that will determine the minimum (>=) weight of relationships that will be taken into account. Relationships with a lower weight (<) will not be shown. Default is set to 1 (*i.e.* all relationships are shown).
filters	A list of named strings that will filter the relationships showed in the SCM. e.g. =list(coding_class = "A_coding_class", document_part = "A_document_part")=. To date, these filters are linked to the nature of relationships.
units	A string vector giving the names of the units (i.e. classes linked to documents) that will be include in the SCM. It is a second type of filter.

Details

This function creates a data frame with all quotes of a a group of documents of a Qualitative Data Analysis (QDA) project

Value

A data frame of relationships and quotes

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

QuotesSocCMap(my.project)
QuotesSocCMap(my.project, units = 'Québec')
```

`RelationshipTest` *Compare relationships weight between maps*

Description

This function test the differences between the properties of relationships

Usage

```
RelationshipTest(
  project,
  units,
  output = "p.value",
  weighted.icm = FALSE,
  sep = ">",
  coder = "qcoder"
)
```

Arguments

`project`	A QDA project, a list as generated by the ProjectCMap function.
`units`	The units to compare
`output`	"p.value" (default) or "raw.data".
`weighted.icm`	A boolean. If FALSE, the weight of the relationships in the ICM will be fixed to 1.
`sep`	(==Depreciated, with RQDA project==) A character string (often a single character) that is used in RQDA in order to express the relationships between two variables. Default is "_" if codes used in RQDA are of the form : "x_y" (*i.e.* relationship from x to y), but it is possible to use ">", "->", "–>" or even "—>"
`coder`	A character string indicating the coding tool used for coding the QDA. The only tool supported now is 'qcoder'. Earlier version of cogmapr worked with "RQDA" project (no more maintained).

Details

This function test the differences between the weight of relationships between groups of documents (i.e. between social cognitive maps). Till now, only two excluding groups can be tested (ex. document from one country vs another country, from a group of players vs another group of players). It is not possible to compare non exclusive groups (ex. map from one country vs map from one group of players, as some documents can be in the two groups!). For this test, the 'fisher.test' is done. If output = 'p.value', the function returns the results of the tests, one test for each relationships of the map. If output = 'raw.data', the function returns the raw data on which the tests are done, one data frame by concept. This option can be used to export data and perform other statistical tests.

If more then 2 groups, 'anova' can be used as test (to be confirmed).

Value

A data frame (if output = "p.value"), a list of data frame (if output = "raw.data").

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

## need more documents
RelationshipTest(my.project, units = c("Belgium", "Québec"))
```

RelationshipTestSummary

Summary table on relationship comparisons

Description

Summary table on relationship comparisons

Usage

```
RelationshipTestSummary(project, units, limit.p.value = 0.05)
```

Arguments

project	A QDA project, a list as generated by the ProjectCMap function.
units	The units to compare
limit.p.value	A numeric.

Details

This function produce a summary table based on relationship comparisons and is reactive to a limit of p.value beyond which differences are considered as significant and are reported in the table

Value

A data frame

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

## Here 0.6 is used only for producing an output. No signif. diff. is reported.
RelationshipTestSummary(my.project, units = c("Belgium", "Québec"), 0.6)
```

RemoveCodings *Remove codings of a QDA project*

Description

Remove codings of a QDA project

Usage

```
RemoveCodings(project, codings_id)
```

Arguments

project	A QDA project (as created by the qcoder package)
codings_id	A vector of integer corresponding with the id of the codings to remove

Details

This function removes one or many codings of a Qualitative Data Analysis (QDA) project. The codings are listed using the 'id' of codings.

Value

A QDA project

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

my.cleansed.project <- RemoveCodings(my.project, 1)
```

SocCMap *Social Cognitive Mapping*

Description

Formatting the data for plotting an Social Cognitive Map

Usage

```
SocCMap(
  data.edges.soc,
  project,
  label = "num",
  minlen = 1,
  fontsize = 16,
  shape = "box"
)
```

Arguments

data.edges.soc	A data.frame as produced by the EdgSocCMap function
project	A QDA project, a list as generated by the ProjectCMap function.
label	A character string that defines the text that will be print in the variables (vertex) of the cognitive maps. It can be "num", "name" or "numname" (which is of the form "NUM - Name"). The default is "num"
minlen	A graphical parameter that defines a relative length between the variable of the cognitive maps. See help from RGraphViz package.
fontsize	The fontsize of vertices (concepts), in r-base plot
shape	The shape of the vertices (concepts), in r-base plot

Value

a 'SocCMap' object, a list containing various information that could be use for plotting an Individual Cognitive Map. The most important elements are :

"vertex" A list of information on Cognitive Map's variables (i.e. vertices)

"edg" A list of information about relationships ##'

"graph" A graphNEL object

"eAttrs" A list of graphical attributes of edges

"nAttrs" A list of graphical attributes of nodes (vertices)

"gAttrs" A list of graphical attributes of the whole graph

Examples

```
project_name <- "a_new_project"
main_path <- paste0(system.file("testdata", package = "cogmapr"), '/')
my.project <- ProjectCMap(main_path, project_name)

edg.scm <- EdgSocCMap(my.project, min.weight=6, weighted.icm=TRUE)
SocCMap(edg.scm, my.project)
```

Index