Package 'rENA'

January 26, 2018

Title Epistemic Network Analysis

Type Package

Author Cody L Marquart [aut, cre],
Zachari Swiecki [aut],
Wesley Collier [aut],
Brendan Eagan [aut],
Roman Woodward [aut],
David Williamson Shaffer [aut]

Maintainer Cody L Marquart <cody.marquart@wisc.edu>

Version 0.1.3

Description ENA (Shaffer, D. W. (2017) Quantitative Ethnography. ISBN: 0578191687) is a method used to identify meaningful and quantifiable patterns in discourse or reasoning. ENA moves beyond the traditional frequency-based assessments by examining the structure of the co-occurrence, or connections in coded data. Moreover, compared to other methodological approaches, ENA has the novelty of (1) modeling whole networks of connections and (2) affording both quantitative and qualitative comparisons between different network models. Shaffer, D.W., Collier, W., & Ruis, A.R. (2016) <doi:10.18608/jla.2016.33.3>.

LazyData TRUE

Depends R (>= 3.0.0)

License GPL-3 | file LICENSE

LinkingTo Rcpp, RcppArmadillo, RcppParallel, RcppEigen

Imports data.table, Rcpp, R6, methods, foreach, stats, plotly, doParallel, parallel, RcppRoll, data.tree, scales, magrittr

Suggests testthat

RoxygenNote 5.0.1

NeedsCompilation yes

Repository CRAN

Date/Publication 2018-01-26 14:59:40 UTC

R topics documented:

`cohens.d` *Cohen's d*

Description

Calculate Conhen's d

Usage

```
cohens.d(x, y)
```

Arguments

x	[TBD]
y	[TBD]

Details

[TBD]

Value

numeric Cohen's d calculation

`ena.accumulate.data` *Accumulate data from a data frame into a set of adjacency (co-occurrence) vectors*

Description

This function initializes an ENAdata object, processing conversations from coded data to generate adjacency (co-occurrence) vectors

Usage

```
ena.accumulate.data(units = NULL, conversation = NULL, codes = NULL,
  metadata = NULL, model = c("EndPoint", "AccumulatedTrajectory",
  "SeparateTrajectory"), weight.by = "binary", window = c("Moving Stanza",
  "Conversation"), window.size.back = 1, window.size.forward = 0,
  mask = NULL, ...)
```

Arguments

`units`	A data frame where the columns are the properties by which units will be identified
`conversation`	A data frame where the columns are the properties by which conversations will be identified
`codes`	A data frame where the columns are the codes used to create adjacency (co-occurrence) vectors
`metadata`	(optional) A data frame with additional columns of metadata to be associated with each unit in the data
`model`	A character, choices: EndPoint (or E), AccumulatedTrajectory (or A), or SeparateTrajectory (or S); default: EndPoint. Determines the ENA model to be constructed
`weight.by`	(optional) A function to apply to values after accumulation
`window`	A character, choices are Conversation (or C), MovingStanzaWindow (or MovingStanza, MSW, MS, S); default MovingStanzaWindow. Determines how stanzas are constructed, which defines how co-occurrences are modeled
`window.size.back`	A positive integer or character (INF or Infinite), default: 1. Determines, for each line in the data frame, the number of previous lines in a conversation to include in the stanza window, which defines how co-occurrences are modeled
`window.size.forward`	(optional) A positive integer, default: 1. Determines, for each line in the data frame, the number of subsequent lines in a conversation to include in the stanza window, which defines how co-occurrences are modeled
`mask`	(optional) A binary matrix of size ncol(codes) x ncol(codes). 0s in the mask matrix row i column j indicates that co-occurrence will not be modeled between code i and code j
`...`	additional parameters addressed in inner function

Details

ENAData objects are created using this function. This accumulation receives separate data frames for units, codes, conversation, and optionally, metadata. It iterates through the data to create an adjacency (co-occurrence) vector corresponding to each unit - or in a trajectory model multiple adjacency (co-occurrence) vectors for each unit.

In the default MovingStanzaWindow model, co-occurrences between codes are calculated for each line k in the data between line k and the window.size.back-1 previous lines and window.size.forward-1 subsequent lines in the same conversation as line k.

In the Conversation model, co-occurrences between codes are calculated across all lines in each conversation. Adjacency (co-occurrence) vectors are constructed for each unit u by summing the co-occurrences for the lines that correspond to u.

Options for how the data is accumulated are endpoint, which produces one adjacency (co-occurrence) vector for each until summing the co-occurrences for all lines, and two trajectory models: AccumulatedTrajectory and SeparateTrajectory. Trajectory models produce an adjacency (co-occurrence) model for each conversation for each unit. In a SeparateTrajectory model, each conversation is modeled as a separate network. In an AccumulatedTrajectory model, the adjacency (co-occurrence) vector for the current conversation includes the co-occurrences from all previous conversations in the data.

Value

ENAdata object with data [adjacency (co-occurrence) vectors] accumulated from the provided data frames.

See Also

ENAdata, ena.make.set

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
  window.size.back = 4
)
```

ena.conversation *Find conversations by unit*

Description

Find rows of conversations by unit

Usage

```
ena.conversation(data, units, units.by, conversation.by, window, codes = NULL)
```

Arguments

data	[TBD]
units	[TBD]
units.by	[TBD]
conversation.by	[TBD]
window	[TBD]
codes	[TBD]

Details

[TBD]

Value

list containing the accumulation and set

ena.correlations *Calculate the correlations*

Description

Calculate both Spearman and Pearson correlations for the provided ENAset

Usage

```
ena.correlations(enaset)
```

Arguments

enaset	ENAset to run correlations on

Value

Matrix of 2 columns, one for each correlation method, with the corresponding correlations per dimension as the rows.

ena.group *Compute summary statistic for groupings of units using given method (typically, mean)*

Description

Computes summary statistics for groupings (given as vector) of units in ena data using given method (typically, mean); computes summary statistic for point locations and edge weights for each grouping

Usage

```
ena.group(enaset = NULL, by = NULL, method = mean)
```

Arguments

enaset	An ENAset (optional)
by	A vector of values the same length as units. Uses rotated points for group positions and normed data to get the group edge weights
method	A function that is used on grouped points. Default: mean()

Value

A list containing names, points, and edge weights for each of the unique groups formed by the function

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
  window.size.back = 4
)

set = ena.make.set(
  enadata = accum
)

means = ena.group(set, by=accum$metadata$Condition)
```

ena.make.set *Generate ENA Set*

Description

Generates an ENA model by constructing a dimensional reduction of adjacency (co-occurrence) vectors in an ENA data object

Usage

```
ena.make.set(enadata, dimensions = 2, norm.by = sphere_norm_c,
  rotation.by = ena.svd, rotation.params = NULL, rotation.set = NULL,
  endpoints.only = T, node.position.method = lws.positions.sq, ...)
```

Arguments

enadata	ENAdata that will be used to generate an ENA model
dimensions	The number of dimensions to include in the dimensional reduction
norm.by	A function to be used to normalize adjacency (co-occurrence) vectors before computing the dimensional reduction, default: sphere_norm_c()
rotation.by	A function to be used to compute the dimensional reduction, default: ena.svd()
rotation.params	(optional) A character vector containing additional parameters for the function in rotation.by, if needed
rotation.set	A previously-constructed ENARotationSet object to use for the dimensional reduction
endpoints.only	A logical variable which determines whether to only show endpoints for trajectory models
node.position.method	A function to be used to determine node positions based on the dimensional reduction, default: lws.position.es()
...	additional parameters addressed in inner function

Details

This function generates an ENAset object from an ENAdata object. Takes the adjacency (co-occurrence) vectors from enadata, computes a dimensional reduction (projection), and calculates node positions in the projected ENA space. Returns location of the units in the projected space, as well as locations for node positions, and normalized adjacency (co-occurrence) vectors to construct network graphs

Value

ENAset class object that can be further processed for analysis or plotting

See Also

ena.accumulate.data, ENAset

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
  window.size.back = 4
)

set = ena.make.set(
  enadata = accum
)

set.means.rotated = ena.make.set(
  enadata = accum,
  rotation.by = ena.rotate.by.mean,
  rotation.params = list(
      accum$metadata$Condition=="FirstGame",
      accum$metadata$Condition=="SecondGame"
  )
)
```

---

ena.plot *Generate a plot of an ENAset*

Description

Generates an a plot from a given ENA set object

Usage

```
ena.plot(enaset, title = "ENA Plot", dimensions = c(1, 2),
  dimension.labels = c("X", "Y"), dimension.show.variance = T,
  end.points = F, flip.axis.x = F, flip.axis.y = F, font.size = 10,
  font.color = "#000000", font.family = c("Arial", "Courier New",
  "Times New Roman"), scale.to = c("network", "points"), ...)
```

Arguments

enaset	The ENAset that will be used to generate a plot
title	A character used for the title of the plot, default: ENA Plot
dimensions	A numeric vector determining the dimensions from the projected ENA space to include in the plot, default: c(1,2)
dimension.labels	A character vector containing labels for the axes, default: c(X, Y)
dimension.show.variance	A logical indicating whether to display values for variance accounted for by each dimension in the plot; default: T
end.points	A logical variable that determines whether to show only endpoints for trajectory models; default: F
flip.axis.x	A logical that controls whether the x-axis is inverted before plotting, default: F
flip.axis.y	A logical that controls whether the y-axis is inverted before plotting, default: F
font.size	An integer determining the font size for graph labels, default: 10
font.color	A character determining the color of label font, default: black
font.family	A character determining the font type, choices: Arial, Courier New, Times New Roman, default: Arial
scale.to	Either network or points. Default: "network"
...	additional parameters addressed in inner function

Details

This function defines the axes and other features of a plot for displaying an ENAset; generates an ENAplot object that can used to plot points, network graphs, and other information from an ENAset

Value

ENAplot used for plotting an ENAset

See Also

ena.make.set, ena.plot.points

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
```

```
  window.size.back = 4
)

set = ena.make.set(
  enadata = accum
)

plot = ena.plot(set)

group1.points = set$points.rotated[set$enadata$units$Condition == "FirstGame",]
plot = ena.plot.points(plot, points = group1.points);
print(plot);
```

`ena.plot.group` *Plot of ENA set groups*

Description

Plot a point based on a summary statistic computed from a given method (typically, mean) for a set of points in a projected ENA space

Usage

```
ena.plot.group(enaplot, points = NULL, method = "mean", labels = NULL,
  colors = "black", shape = c("square", "triangle-up", "diamond", "circle"),
  confidence.interval = c("none", "crosshairs", "box"),
  outlier.interval = c("none", "crosshairs", "box"), label.offset = NULL,
  label.font.size = NULL, label.font.color = NULL,
  label.font.family = NULL, show.legend = T, ...)
```

Arguments

`enaplot`	ENAplot object to use for plotting
`points`	A matrix or data.frame where columns contain coordinates of points in a projected ENA space
`method`	A function for computing a summary statistic for each column of points
`labels`	A character which will be the label for the group's point
`colors`	A character, determines color of the group's point, default: enaplot$color
`shape`	A character, determines shape of the group's point, choices: square, triangle, diamond, circle, default: square
`confidence.interval`	A character that determines how the confidence interval is displayed, choices: none, box, crosshair, default: none
`outlier.interval`	A character that determines how outlier interval is displayed, choices: none, box, crosshair, default: none

label.offset — character: top left (default), top center, top right, middle left, middle center, middle right, bottom left, bottom center, bottom right

label.font.size — An integer which determines the font size for label, default: enaplot\$font.size

label.font.color — A character which determines the color of label, default: enaplot\$font.color

label.font.family — A character which determines font type, choices: Arial, Courier New, Times New Roman, default: enaplot\$font.family

show.legend — Logical indicating whether to show the point labels in the in legend

... — Additional parameters

Details

Plots a point based on a summary statistic for a group (typically, mean)

Value

The ENAplot provided to the function, with its plot updated to include the new group point.

See Also

ena.plot, ena.plot.points

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
  window.size.back = 4
)

set = ena.make.set(
  enadata = accum,
  rotation.by = ena.rotate.by.mean,
  rotation.params = list(
      accum$metadata$Condition=="FirstGame",
      accum$metadata$Condition=="SecondGame"
  )
)

plot = ena.plot(set)
```

```
unitNames = set$enadata$units

### Subset rotated points and plot Condition 1 Group Mean
first.game = unitNames$Condition == "FirstGame"
first.game.points = set$points.rotated[first.game,]
plot = ena.plot.group(plot, first.game.points, labels = "FirstGame",
    colors = "red", confidence.interval = "box")

### Subset rotated points and plot Condition 2 Group Mean
second.game = unitNames$Condition == "SecondGame"
second.game.points = set$points.rotated[second.game,]
plot = ena.plot.group(plot, second.game.points, labels = "SecondGame",
    colors  = "blue", confidence.interval = "box")

print(plot);
```

`ena.plot.network` *Plot an ENA network*

Description

Plot an ENA network: nodes and edges

Usage

```
ena.plot.network(enaplot = NULL, network = NULL,
  node.positions = enaplot$enaset$node.positions,
  adjacency.key = namesToAdjacencyKey(rownames(node.positions)),
  colors = c(pos = "red", "blue"), show.all.nodes = T, threshold = 0,
  thin.lines.in.front = T, opacity = c(0.3, 1), saturation = c(0.25, 1),
  thickness = c(0.1, 1), node.size = c(3, 10), range = c(min(network),
  max(network)), labels = rownames(node.positions), label.offset = NULL,
  label.font.size = enaplot$get("font.size"),
  label.font.color = enaplot$get("font.color"),
  label.font.family = enaplot$get("font.family"), legend.name = NULL,
  legend.include.edges = F, scale.weights = T, ...)
```

Arguments

`enaplot`	ENAplot object to use for plotting
`network`	dataframe or matrix containing the edge weights for the network graph; typically comes from ENAset$line.weights
`node.positions`	matrix containing the positiions of the nodes. Defaults to enaplot$enaset$node.positions
`adjacency.key`	matrix containing the adjacency key for looking up the names and positions
`colors`	A String or vector of colors for positive and negative line weights. E.g. red or c(pos= red, neg = blue), default: c(pos= red, neg = blue)

show.all.nodes	A Logical variable, default: true
threshold	A vector of numeric min/max values, default: (0,1). Edge weights below the min value will not be displayed; edge weights above the max value will be shown at the max value.
thin.lines.in.front	A logical, default: true
opacity	A vector of numeric min/max values for opacity, default: (0.3,1)
saturation	A vector of numeric min/max values for saturation, default: (0.25, 1)
thickness	A vector of numeric min/max values for thickness, default: (0, 1)
node.size	A lower and upper bound used for scaling the size of the nodes, default c(0, 20)
range	A vector of min/max values. Options are numeric, set.min, set.max, plot.min, plot.max, default: (set.min, set.max). Determines the line weight that corresponds to the thinnest (min) and thickest (max) lines in the network graph
labels	A character vector of node labels, default: code names
label.offset	A numeric vector of an x and y value to offset labels from the coordinates of the points
label.font.size	An integer which determines the font size for graph labels, default: enaplot$font.size
label.font.color	A character which determines the color of label font, default: enaplot$font.color
label.font.family	A character which determines font type, choices: Arial, Courier New, Times New Roman, default: enaplot$font.family
legend.name	A character name to include in the legend. Not included in legend when NULL. Default: NULL
legend.include.edges	Logical value indicating if the edges should be included in the plot
scale.weights	Logical indicating to scale the supplied network
...	Additional parameters

Details

lots a network graph, including nodes (taken from codes in the ENAplot) and the edges (provided in network)

Value

The ENAplot provided to the function, with its plot updated to include the nodes and provided connecting lines.

See Also

ena.plot, ena.plot.points

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
  window.size.back = 4
)

set = ena.make.set(
  enadata = accum,
  rotation.by = ena.rotate.by.mean,
  rotation.params = list(
      accum$metadata$Condition=="FirstGame",
      accum$metadata$Condition=="SecondGame"
  )
)

plot = ena.plot(set)

unitNames = set$enadata$units

### Subset rotated points and plot Condition 1 Group Mean
first.game = unitNames$Condition == "FirstGame"
first.game.points = set$points.rotated[first.game,]
plot = ena.plot.group(plot, first.game.points, labels = "FirstGame",
    colors = "red", confidence.interval = "box")

### Subset rotated points and plot Condition 2 Group Mean
second.game = unitNames$Condition == "SecondGame"
second.game.points = set$points.rotated[second.game,]
plot = ena.plot.group(plot, second.game.points, labels = "SecondGame",
    colors  = "blue", confidence.interval = "box")

### get mean network plots
first.game.lineweights = set$line.weights[first.game,]
first.game.mean = colMeans(first.game.lineweights)

second.game.lineweights = set$line.weights[second.game,]
second.game.mean = colMeans(second.game.lineweights)

subtracted.network = first.game.mean - second.game.mean
plot = ena.plot.network(plot, network = subtracted.network)
print(plot)
```

`ena.plot.points` *Plot points on an ENAplot*

Description

Plot all or a subset of the points of an ENAplot using the plotly plotting library

Usage

```
ena.plot.points(enaplot, points = NULL, point.size = 5, labels = NULL,
  label.offset = "top left", label.group = "Points",
  label.font.size = NULL, label.font.color = NULL,
  label.font.family = NULL, shape = "circle", colors = default.colors[1],
  confidence.interval.values = NULL, confidence.interval = c("none",
  "crosshairs", "box"), outlier.interval.values = NULL,
  outlier.interval = c("none", "crosshairs", "box"), show.legend = T, ...)
```

Arguments

`enaplot`	ENAplot object to use for plotting
`points`	A dataframe of matrix where the first two column are X and Y coordinates
`point.size`	A data.frame or matrix where the first two column are X and Y coordinates of points to plot in a projected ENA space defined in ENAplot
`labels`	A character vector of point labels, length nrow(points); default: NULL
`label.offset`	character: top left (default), top center, top right, middle left, middle center, middle right, bottom left, bottom center, bottom right
`label.group`	A character vector used to group the labels in the legend
`label.font.size`	An integer which determines the font size for point labels, default: enaplot$font.size
`label.font.color`	A character which determines the color of label font, default: enaplot$font.color
`label.font.family`	A character which determines label font type, choices: Arial, Courier New, Times New Roman, default: enaplot$font.family
`shape`	A character which determines the shape of point markers, choices: square, triangle, diamond, circle, default: circle
`colors`	A character vector of the point marker colors; if one given it is used for all, otherwise must be same length as points; default: black
`confidence.interval.values`	A matrix/dataframe where columns are CI x and y values for each point
`confidence.interval`	A character determining markings to use for confidence intervals, choices: none, box, crosshair, default: none

outlier.interval.values
: A matrix/dataframe where columns are OI x and y values for each point

outlier.interval
: A character determining markings to use for outlier interval, choices: none, box, crosshair, default: none

show.legend
: Logical indicating whether to show the point labels in the in legend

...
: additional parameters addressed in inner function

Value

ENAplot The ENAplot provided to the function, with its plot updated to include the new points.

See Also

ena.plot, ENAplot, ena.plot.group

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("Condition","GroupName")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post")],
  codes = RS.data[,codeNames],
  window.size.back = 4
)

set = ena.make.set(
  enadata = accum,
  rotation.by = ena.rotate.by.mean,
  rotation.params = list(
      accum$metadata$Condition=="FirstGame",
      accum$metadata$Condition=="SecondGame"
  )
)

plot = ena.plot(set)

group1.points = set$points.rotated[set$enadata$units$Condition == "FirstGame",]
group2.points = set$points.rotated[set$enadata$units$Condition == "SecondGame",]
plot = ena.plot.points(plot, points = group1.points);
plot = ena.plot.points(plot, points = group2.points);
print(plot);
```

`ena.plot.trajectory` *Plot of ENA trajectories*

Description

Function used to plot trajectories

Usage

```
ena.plot.trajectory(enaplot, points, by = NULL, labels = NULL,
  labels.show = c("Always", "Hover", "Both"), names = NULL,
  label.offset = NULL, label.font.size = enaplot$get("font.size"),
  label.font.color = enaplot$get("font.color"),
  label.font.family = c("Arial", "Courier New", "Times New Roman"),
  shape = c("circle", "square", "triangle-up", "diamond"),
  colors = rep(I("black"), length(unique(by))), confidence.interval = NULL,
  confidence.interval.values = NULL, outlier.interval = NULL,
  outlier.interval.values = NULL, default.hidden = F)
```

Arguments

`enaplot`	ENAplot object to use for plotting
`points`	dataframe of matrix - first two column are X and Y coordinates, each row is a point in a trajectory
`by`	vector used to subset points into individual trajectories, length nrow(points)
`labels`	character vector - point labels, length nrow(points)
`labels.show`	A character choice: Always, Hover, Both. Default: Both
`names`	character vector - labels for each trajectory of points, length length(unique(by))
`label.offset`	A numeric vector of an x and y value to offset labels from the coordinates of the points
`label.font.size`	An integer which determines the font size for labels, default: enaplot\$font.size
`label.font.color`	A character which determines the color of label font, default: enaplot\$font.color
`label.font.family`	A character which determines font type, choices: Arial, Courier New, Times New Roman, default: enaplot\$font.family
`shape`	A character which determines the shape of markers, choices: square, triangle, diamond, circle, default: circle
`colors`	A character, determines marker color, default: enaplot\$color
`confidence.interval`	A character that determines which confidence interval type to use, choices: none, box, crosshair, default: none

`confidence.interval.values`
: A matrix/dataframe where columns are CI x and y values for each point

`outlier.interval`
: A character that determines which outlier interval type to use, choices: none, box, crosshair, default: none

`outlier.interval.values`
: A matrix/dataframe where columns are OI x and y values for each point

`default.hidden`
: A logical indicating if the trajectories should start hidden (click on the legend to show them) Default: FALSE

Value

The `ENAplot` provided to the function, with its plot updated to include the trajectories

See Also

`ena.plot`

Examples

```
data(RS.data)

codeNames = c('Data','Technical.Constraints','Performance.Parameters',
  'Client.and.Consultant.Requests','Design.Reasoning','Collaboration');

accum = ena.accumulate.data(
  units = RS.data[,c("UserName","Condition")],
  conversation = RS.data[,c("GroupName","ActivityNumber")],
  metadata = RS.data[,c("CONFIDENCE.Change","CONFIDENCE.Pre","CONFIDENCE.Post","C.Change")],
  codes = RS.data[,codeNames],
  window.size.back = 4,
  model = "A"
);

set = ena.make.set(accum);

unitNames = set$enadata$units

### Subset rotated points and plot Condition 1 Group Mean
first.game = unitNames$Condition == "FirstGame"
first.game.points = set$points.rotated[first.game,]

### Subset rotated points and plot Condition 2 Group Mean
second.game = unitNames$Condition == "SecondGame"
second.game.points = set$points.rotated[second.game,]

### get mean network plots
first.game.lineweights = set$line.weights[first.game,]
first.game.mean = colMeans(first.game.lineweights)

second.game.lineweights = set$line.weights[second.game,]
```

```
second.game.mean = colMeans(second.game.lineweights)

subtracted.network = first.game.mean - second.game.mean

# Plot dimension 1 against ActivityNumber metadata
dim.by.activity = cbind(
    set$points.rotated[,1],
    set$enadata$trajectories$step$ActivityNumber*.8/14-.4  #scale down to dimension 1
)

plot = ena.plot(set)
plot = ena.plot.network(plot, network = subtracted.network, legend.name="Network")
plot = ena.plot.trajectory(
  plot,
  points = dim.by.activity,
  names = unique(set$enadata$units$UserName),
  by = set$enadata$units$UserName
);
print(plot)
```

`ena.rotate.by.mean` *ENA Rotate by mean*

Description

Computes a dimensional reduction from a matrix of points such that the first dimension of the projected space passes through the means of two groups in a the original space. Subsequent dimensions of the projected space are computed using ena.svd

Usage

```
ena.rotate.by.mean(enaset, groups)
```

Arguments

`enaset`	An `ENAset`
`groups`	A list containing two logical vectors of length `nrow(ENA.set$ena.data$units)`, where each vector defines whether a unit is in one of the two groups whose means are used to determine the dimensional reduction

Value

`ENARotationSet`

ena.svd *ENA SVD*

Description

ENA method computing a dimensional reduction of points in an ENA set using SVD

Usage

```
ena.svd(enaset, ...)
```

Arguments

enaset	An ENAset
...	Unused, necessary for ena.make.set

ENAdata *ENAdata R6class*

Description

ENAdata R6class

Usage

```
ENAdata
```

Format

An object of class R6ClassGenerator of length 24.

Fields

raw A data frame constructed from the unit, convo, code, and metadata parameters of ena.accumulate.data

adjacency.vectors A data frame of adjacency (co-occurrence) vectors by row

accumulated.adjacency.vectors A data frame of adjacency (co-occurrence) vectors accumulated per unit

model The type of ENA model: EndPoint, Accumulated Trajectory, or Separate Trajectory

units A data frame of columns that were combined to make the unique units. Includes column for trajectory selections. (unique)

unit.names A vector of unique unit values

metadata A data frame of unique metadata for each unit

trajectories A list: units - data frame, for a given row tells which trajectory it's a part; step - data frame, where along the trajectory a row sits

`codes` A vector of code names

`function.call` The string representation of function called and parameters provided

`function.params` A list of all parameters sent to function call

`ENAplot` *ENAset R6class*

Description

ENAset R6class

Usage

```
ENAplot
```

Format

An object of class `R6ClassGenerator` of length 24.

Fields

`enaset` - The `ENAset` object from which the ENAplot was constructed

`plot` - The plotly object used for data visualization

`ENARotationSet` *ENARotationSet R6class*

Description

ENARotationSet R6class

Usage

```
ENARotationSet
```

Format

An object of class `R6ClassGenerator` of length 24.

ENAset *ENAset R6class*

Description

ENAset R6class

Usage

```
ENAset
```

Format

An object of class `R6ClassGenerator` of length 24.

Fields

`enadata` An `ENAdata` object originally used to create the set

`points.raw` A data frame containing accumulated adjacency (co-occurrence) vectors per unit

`points.normed.centered` A data frame of centered normed accumulated adjacency (co-occurrence) vectors for each unit

`points.rotated` A data frame of point positions for number of dimensions specified in ena.make.set (i.e., the centered, normed, and rotated data)

`line.weights` A data frame of connections strengths per unit (Data frame of normed accumulated adjacency (co-occurrence) vectors for each unit)

`node.positions` - A data frame of positions for each code

`codes` - A vector of code names

`rotation.set` - An `ENARotationSet` object

`correlation` - A data frame of spearman and pearson correlations for each dimension specified

`variance` - A vector of variance accounted for by each dimension specified

`centroids` - A matrix of the calculated centroid positions

`function.call` - The string representation of function called

`function.params` - A list of all parameters sent to function call

group.stats *Group Stats*

Description

Generate comparison stats for groups within set

Usage

```
group.stats(groupOne, groupTwo)
```

Arguments

groupOne	[TBD]
groupTwo	[TBD]

Details

[TBD]

Value

list containing all of the statistics

namesToAdjacencyKey *Names to Adjacency Key*

Description

Convert a vector of strings, representing names of a square matrix, to an adjacency

Usage

```
namesToAdjacencyKey(vector)
```

Arguments

vector	Vector representing the names of a square matrix

Details

Returns a matrix of 2 rows by choose(length(vector), 2) columns

rENA *rENA creates ENA sets*

Description

rENA is used to create and visualize network models of discourse and other phenomena from coded data using Epistemic Network Analysis (ENA). A more complete description of the methods will be provided with the next release. See also XXXXX

RS.data *Coded Rescushell Chat Data*

Description

A dataset containing sample chat data from the Rescushell Virtual Internship

Usage

```
RS.data
```

Format

An object of class `data.frame` with 3824 rows and 20 columns.

Index