Package ‘behaviorchange’

March 5, 2023

Type Package
Title Tools for Behavior Change Researchers and Professionals
Version 0.5.5
Maintainer Gjalt-Jorn Peters <behaviorchange@opens.science>
License GPL (>= 3)
Description Contains specialised analyses and visualisation tools for behavior change science. These facilitate conducting determinant studies (for example, using confidence interval-based estimation of relevance, CIBER, or CIBERlite plots, see Crutzen, Noijen & Peters (2017) <doi:10/ghtfz9>), systematically developing, reporting, and analysing interventions (for example, using Acyclic Behavior Change Diagrams), and reporting about intervention effectiveness (for example, using the Numbers Needed for Change, see Gruijters & Peters (2017) <doi:10/jzkt>), and computing the required sample size (using the Meaningful Change Definition, see Gruijters & Peters (2020) <doi:10/ghpx8>). This package is especially useful for researchers in the field of behavior change or health psychology and to behavior change professionals such as intervention developers and prevention workers.

URL https://r-packages.gitlab.io/behaviorchange

BugReports https://gitlab.com/r-packages/behaviorchange/-/issues
Encoding UTF-8
LazyData true
RoxygenNote 7.2.3
Depends R (>= 3.5.0)
R topics documented:

Imports BiasedUrn (>= 1.07), data.tree (>= 0.7.5), DiagrammeR (>= 1.0.0), DiagrammeRsvg (>= 0.1.0), ggplot2 (>= 2.2.1), googlesheets4 (>= 0.2.0), gridExtra (>= 2.3), gtable (>= 0.2.0), knitr (>= 1.0), methods (>= 3.0), rmdpartials (>= 0.5.0), ufs (>= 0.3.2), viridis (>= 0.5.1), yum (>= 0.0.1)

Suggests htmltools, kableExtra, openxlsx, png (>= 0.1), rmarkdown, rstudioapi, rsvg, webshot

VignetteBuilder knitr

NeedsCompilation no

Author Gjalt-Jorn Peters [aut, cre] (https://orcid.org/0000-0002-0336-9589), Rik Crutzen [ctb] (https://orcid.org/0000-0002-3731-6610), Stefan Gruijters [ctb] (https://orcid.org/0000-0003-0141-0071)

Repository CRAN

Date/Publication 2023-03-04 23:20:02 UTC

R topics documented:

abcd ................................................................. 3
abcd_specs_examples ........................................... 7
apply_graph_theme .............................................. 8
cat0 ................................................................. 9
CIBER ................................................................. 9
CIBERlite ........................................................... 15
complecs .......................................................... 16
complecs_to_precede .......................................... 19
convert.threshold.to.er ..................................... 21
determinantStructure .......................................... 25
determinant_selection_table .................................. 26
detStructAddVarLabels ....................................... 33
dMCD ................................................................. 36
lm_rSq_ci .......................................................... 37
nnc ................................................................. 38
opts ................................................................. 41
partypanelData .................................................. 42
pies ................................................................. 43
repeatStr .......................................................... 44
vecTxt ............................................................. 45
wrapVector ........................................................ 46

Index 47
Acyclic Behavior Change Diagram

Description

This function generates an acyclic behavior change diagram (ABCD) from a specification in a Google sheet or .csv file. An ABCD is a logic model that illustrates the assumptions underlying a behavior change intervention. Specifically, the ABCD shows the assumed causal and structural assumptions, thereby showing what is assumed to cause what (e.g., which elements of the intervention are assumed to influence which aspects of the target population's psychology?) and what is assumed to consist of what (e.g., which determinants are assumed to contain which specific aspects of the target population's psychology?).

Usage

```r
abcd(
  specs,
  specCols = c("bcps", "cnds", "apps", "sdts", "dets", "pobs", "behs"),
  localBackup = NULL,
  title = "Acyclic Behavior Change Diagram\n\n",
  outputFile = NULL,
  outputWidth = 3000,
  outputHeight = 1500,
  includeColNames = TRUE,
  maxLabelLength = 30,
  nodeFontSize = 10,
  edgeFontSize = 8,
  colNameFontSize = nodeFontSize,
  grayscale = FALSE,
  colorTheme = behaviorchange::opts$get("aabbcc"),
  penWidth = 1,
  silent = FALSE,
  returnGraphOnly = FALSE,
  returnSvgOnly = FALSE,
  columnWarning = TRUE,
  graphTheme = list(c("fontname", "Arial", "node")),
  regExReplacements = behaviorchange::opts$get("diagrammerSanitization")
)
```

## S3 method for class 'abcdiagram'

```r
print(
  x,
  width = x$input$width,
  height = x$input$height,
  title = DiagrammeR::get_graph_name(x$output$graph),
  ...
)
```
### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>specs</td>
<td>The specifications: either a google sheets URL, the path to a local file, a character vector with both, or a matrix or data frame.</td>
</tr>
<tr>
<td>specCols</td>
<td>The order of the columns. This character vector specified the order of the elements of an ABCD. In the default order, from left to right, these are (see below for definitions and more details):</td>
</tr>
<tr>
<td></td>
<td>• <code>bcps</code> = <strong>Behavior Change Principles (BCPs)</strong>;</td>
</tr>
<tr>
<td></td>
<td>• <code>cnds</code> = <strong>Conditions for effectiveness</strong>;</td>
</tr>
<tr>
<td></td>
<td>• <code>apps</code> = <strong>Applications</strong>;</td>
</tr>
<tr>
<td></td>
<td>• <code>sdts</code> = <strong>Sub-determinants</strong>;</td>
</tr>
<tr>
<td></td>
<td>• <code>dets</code> = <strong>Determinants</strong>;</td>
</tr>
<tr>
<td></td>
<td>• <code>pobs</code> = <strong>Performance Objectives</strong>;</td>
</tr>
<tr>
<td></td>
<td>• <code>behs</code> = <strong>Behaviors</strong>;</td>
</tr>
<tr>
<td>localBackup</td>
<td>Whether to write the specifications to a local backup.</td>
</tr>
<tr>
<td>title</td>
<td>The title of the diagram.</td>
</tr>
<tr>
<td>outputFile</td>
<td>If specified, the ABCD is written to this file using <strong>DiagrammeR::export_graph</strong>.</td>
</tr>
<tr>
<td>outputWidth, outputHeight</td>
<td>If an <code>outputFile</code> is specified, these determine its width and height (in pixels).</td>
</tr>
<tr>
<td>includeColNames</td>
<td>Whether to include the column names as titles/legend for the entities in each 'column' of the ABCD.</td>
</tr>
<tr>
<td>maxLabelLength</td>
<td>At which width to word wrap the labels.</td>
</tr>
<tr>
<td>nodeFontSize, edgeFontSize, colNameFontSize</td>
<td>Font sizes of the nodes (i.e. the text in boxes), edges (basically the conditions for effectiveness) and the column names (at the bottom).</td>
</tr>
<tr>
<td>grayscale</td>
<td>Whether to use the colorTheme or produce a grayscale ABCD.</td>
</tr>
<tr>
<td>colorTheme</td>
<td>The color theme, a named list containing the colors, each a character vector with three HTML (hex) color values. The list elements have to be named bcp, condition_for_effectiveness, application, sub_determinant, determinant, sub_behavior, and target_behavior, and each must contain a named vector with two elements named fill, stroke, and text, containing the color codes for the fill, stroke, and text, respectively; see behaviorchange::opts$get(&quot;aabbcc&quot;) for an example.</td>
</tr>
<tr>
<td>penWidth</td>
<td>The width of the pen to draw the strokes.</td>
</tr>
<tr>
<td>silent</td>
<td>Whether to suppress (TRUE) or show (FALSE) more detailed information.</td>
</tr>
<tr>
<td>returnGraphOnly, returnSvgOnly</td>
<td>Whether to return the full results object or only either the <strong>DiagrammeR::DiagrammeR</strong> graph or a one-value character vector containing a Scalable Vector Graphic as produced by <strong>DiagrammeRsvg::export_svg</strong>().</td>
</tr>
<tr>
<td>columnWarning</td>
<td>Can be used to suppress the warning if the number of columns is too large.</td>
</tr>
<tr>
<td>graphTheme</td>
<td>Specific settings to apply to the graph using <strong>apply_graph_theme()</strong>; a list of vectors, where each vector has three elements: the setting, the value, and what to apply it to ('node', 'edge', or 'graph').</td>
</tr>
</tbody>
</table>
Details

Specifically, a full ABCD is a model that shows the following elements:

- **Behavior Change Principles (BCPs)**: The specific psychological principles engaged to influence the relevant sub-determinants, usually selected using the determinants to which the sub-determinants ‘belong’. These are also known as methods of behavior change in the Intervention Mapping framework, or behavior change techniques, BCTs, in the Behavior Change Wheel approach. For a list of 99 BCPs, see Kok et al. (2016).

- **Conditions for effectiveness**: The conditions that need to be met for a Behavior Change Principle (BCP) to be effective. These conditions depend on the specific underlying Evolutionary Learning Processes (ELPs) that the BCP engages (Crutzen & Peters, 2018). If the conditions for effectiveness (called *parameters* for effectiveness in the Intervention Mapping framework) are not met, the method will likely not be effective, or at least, not achieve its maximum effectiveness.

- **Applications**: Since BCP’s describe aspects of human psychology in general, they are necessarily formulated on a generic level. Therefore, using them in an intervention requires translating them to the specific target population, culture, available means, and context. The result of this translation is the application of the BCP. Multiple BCPs can be combined into one application; and one BCP can be applied in multiple applications (see Kok, 2014).

- **Sub-determinants**: Behavior change interventions engage specific aspects of the human psychology (ideally, they specifically, target those aspects found most important in predicting the target behavior, as can be established with CIBER plots. These aspects are called sub-determinants (the Intervention Mapping framework references Change Objectives, which are sub-determinants formulated according to specific guidelines). In some theoretical traditions, sub-determinants are called *beliefs*.

- **Determinants**: The overarching psychological constructs that are defined as clusters of specific aspects of the human psychology that explain humans’ behavior (and are targeted by behavior change interventions). Psychological theories contain specific definitions of such determinants, and make statements about how they relate to each other and to human behavior. There are also theories (and exists empirical evidence) on how these determinants can be changed (i.e. BCPs), so although the sub-determinants are what is targeted in an intervention, the selection of feasible BCPs requires knowing to which determinants those sub-determinants belong.

- **Performance objectives**: The specific sub-behaviors that often underlie (or make up) the ultimate target behavior. These are distinguished from the overarching target behavior because the relevant determinants of these sub-behaviors can be different: for example, the reasons why people do or do not buy condoms can be very different from the reasons why they do or do not carry condoms or why they do or do not negotiate condom use with a sexual partner.
• **Behavior**: The ultimate target behavior of the intervention, usually an umbrella that implicitly contains multiple performance objectives.

For details, see Peters et al. (2019).

**Value**

A list consisting of an input, intermediate, and output list, where the ABCD is stored in the output list as a DiagrammeR::DiagrammeR called graph.

**Author(s)**

Gjalt-Jorn Peters, <gjalt-jorn@a-bc.eu>, with contributions from Matti Heino and Sander Eggers.

**References**


**Examples**

```r
### Load one of the ABCD matrices supplied
### with the behaviorchange package
data(abcd_specification_example_xtc);

### Create ABCD matrix (using 'print' to allow pkgdown() to print properly).
print(behaviorchange::abcd(abcd_specification_example_xtc));

### Other examples not executed during testing as creating ABCDs takes long

## Not run:
### Change the appearance; note that many attributes are specified
### for specific elements, and element-level settings always override
### the global settings that can be specified here.
print(
  behaviorchange::abcd(
    abcd_specification_example_xtc,
    graphTheme = list(c("fontname", "Courier New", "node")
  )
)
)
```
Simple example datasets for ABCDs

Description

This are three (nested) datasets illustrating the logic model of change for a simple condom use intervention in a way that can be visualised using the `abcd` function. The full dataset is `abcd_specs_full`, a subset that does not explicitly include the conditions for effectiveness (instead showing letters that can then be explained in, for example, the manuscript text) is called `abcd_specs_without_conditions`, and a version that only contains the information about one sub-behavior (performance objective) is available as `abcd_specs_single_po_without_conditions`. The variables in the full dataset are:

Usage

```r
data(abcd_specs_complete)
data(abcd_specs_without_conditions)
data(abcd_specs_single_po_without_conditions)
data(abcd_specification_example_xtc)
data(abcd_specs_dutch_xtc)
data(abcd_specification_empty)
```

Format

For `abcd_specs_complete`, a data frame with 7 variables and 7 rows; for `abcd_specs_without_conditions`, a data frame with 6 variables and 7 rows; for `abcd_specs_single_po_without_conditions`, a data frame with 5 variables and 4 rows; for `abcd_specification_example_xtc` and `abcd_specs_dutch_xtc`, a data frame with 7 variables and 5 rows; and for `abcd_specification_empty`, a data frame with 7 variables and 1 row.

Details

- **Behavior Change Principles**: The behavior change principles (BCPs), also known as methods for behavior change or 'behavior change techniques' (BCTs), that describe the psychological principles that are assumed to realise the change in the (sub-)determinants.
- **Conditions for effectiveness (e.g. parameters for use)**: The conditions for effectiveness that describe the constraints and considerations taken into account in the translation of the BCPs to practical applications for the relevant target population, context, culture, etc.
• Applications: The applications of these BCPs. Where the BCPs describe theoretical principles, the applications are more or less tangible intervention elements.

• Sub-determinants (e.g. beliefs; can be formulated as Change Objectives): The specific aspects of the target population’s psychology that are targeted by the BCPs (e.g. beliefs, or in Intervention Mapping vocabulary, Change Objectives).

• Determinants: The determinants, psychological constructs, that the targeted sub-determinants are a part of, and that together predict the Performance Objectives (sub-behaviors).

• Performance Objectives: Explicitly defined sub-behaviors at a level of specificity that distinguishes them from other sub-behaviors, and that together form the target behavior.

• Target Behavior: The ultimate target behavior, usually defined at a relatively general level.

In addition to these three datasets, a Dutch example specification is included named abcd_specs_dutch_xtc, and the same in English as abcd_specification_example_xtc.

Finally, abcd_specification_empty is an empty ‘template’ ABCD matrix.

---

**apply_graph_theme**

*Apply multiple DiagrammeR global graph attributes*

**Description**

Apply multiple DiagrammeR global graph attributes

**Usage**

```r
apply_graph_theme(graph, ...)
```

**Arguments**

- `graph` The DiagrammeR::DiagrammeR graph to apply the attributes to.
- `...` One or more character vectors of length three, where the first element is the attribute, the second the value, and the third, the attribute type (graph, node, or edge).

**Value**

The DiagrammeR::DiagrammeR graph.

**Examples**

```r
abcd_complete <- behaviorchange::abcd(behaviorchange::abcd_specs_complete)$output$graph;
abcd_complete <- apply_graph_theme(abcd_complete,
    c("penwidth", 5, "node"),
    c("penwidth", 15, "edge"));
```
cat0

Concatenate to screen without spaces

Description

The cat0 function is to cat what paste0 is to paste; it simply makes concatenating many strings without a separator easier.

Usage

```r
cat0(..., sep = "")
```

Arguments

- `...` The character vector(s) to print; passed to `cat`.
- `sep` The separator to pass to `cat`, of course, "" by default.

Value

Nothing (invisible NULL, like `cat`).

Examples

```r
cat0("The first variable is "", names(mtcars)[1], ".
```

CIBER

Confidence Interval-Based Estimation of Relevance (CIBER)

Description

This function generates a high-level plot consisting of several diamond plots. This function is useful for estimating the relative relevance of a set of determinants of, for example, behavior. The plot in the left hand panel shows each determinant’s distribution with a diamond representing the confidence interval. The right hand plot shows the determinants’ associations to one or more ‘target’ variables, such as behavior or determinants of behavior.

Usage

```r
CIBER(
  data,
  determinants,
  targets,
  conf.level = list(means = 0.9999, associations = 0.95),
  subQuestions = NULL,
  leftAnchors = rep("Lo", length(determinants)),
  rightAnchors = rep("Hi", length(determinants)),
```
binaryCIBER(
  data,
  determinants,
  targets,
  conf.level = list(means = 0.9999, associations = 0.95),
  subQuestions = NULL,
  leftAnchors = rep("Lo", length(determinants)),
  rightAnchors = rep("Hi", length(determinants)),
  outputFile = NULL,
  outputWidth = NULL,
  outputHeight = NULL,
  outputUnits = "in",
  outputParams = list(),
  orderBy = NULL,
  decreasing = NULL,
  numberSubQuestions = FALSE,
  comparisonColors = viridis::viridis(2, end = 0.5),
  categoryLabels = NULL,
  ...
generateColors = list(means = c("red", "blue", "green"), associations = c("red", "grey", "green")),
strokeColors = viridis::viridis(length(targets)),
vLines = c(-0.8, 0, 0.8),
vLineColor = "grey",
titlePrefix = "Means and associations (d) with",
titleVarLabels = NULL,
titleSuffix = "",
fullColorRange = NULL,
associationsAlpha = 0.5,
returnPlotOnly = TRUE,
drawPlot = TRUE,
baseSize = 0.8,
dotSize = 2.5 * baseSize,
baseFontSize = 10 * baseSize,
theme = ggplot2::theme_bw(base_size = baseFontSize),
...}

Arguments

data The dataframe containing the variables.
determinants The 'determinants': the predictors (or 'covariates') of the target variables(s) (or
targets
The 'targets' or 'criteria' variables: the variables predicted by the determinants.

conf.level
The confidence levels for the confidence intervals: has to be a named list with two elements: means and associations, specifying the desired confidence levels for the means and associations, respectively. The confidence level for the associations is also used for the intervals for the proportions of explained variance.

subQuestions
The subquestions used to measure each determinants. This can also be used to provide pretty names for the variables if the determinants were not measured by one question each. Must have the same length as determinants.

leftAnchors
The anchors to display on the left side of the left hand panel. If the determinants were measured with one variable each, this can be used to show the anchors that were used for the respective scales. Must have the same length as determinants.

rightAnchors
The anchors to display on the left side of the left hand panel. If the determinants were measured with one variable each, this can be used to show the anchors that were used for the respective scales. Must have the same length as determinants.

outputFile
The file to write the output to (the plot is not stored to disk if NULL). The extension can be specified to change the file type.

outputWidth, outputHeight, outputUnits
The width, height, and units for the output file.

outputParams
More advanced parameters for the output file. This can be used to pass arguments to ggplot2::ggsave(), such as passing outputParams=list(type="cairo-png") to use anti-aliasing when saving a PNG file.

orderBy
Whether to sort the determinants. Set to NULL to not sort at all; specify the name or index of one of the targets to sort by the point estimates of the associations with that target variable. Use decreasing to determine whether to sort in ascending or descending order. For convenience, if orderBy is not NULL, but decreasing is, the determinants are sorted in descending (decreasing) order.

decreasing
Whether to sort the determinants. Specify NULL to not sort at all, TRUE to sort in descending order, and FALSE to sort in ascending order. If decreasing is nor NULL, but orderBy is NULL, the determinants are sorted by their means. For convenience, if orderBy is not NULL, but decreasing is, the determinants are sorted in descending (decreasing) order.

numberSubQuestions
Whether or not to number the subquestions. If they are numbered, they are numbered from the top to the bottom.

generateColors
The colors to use to generate the gradients for coloring the diamonds representing the confidence intervals. Has to be a named list with two elements: means and associations, specifying the desired colors for the means and associations, respectively.

strokeColors
The palette to use to color the stroke of the confidence intervals for the associations between the determinants and the targets. Successive colors from this palette are used for the targets.
vLines, vLineColors

In the association plot, vertical lines can be plotted to facilitate interpretation. Specify their locations and colors here, or set one or both to NULL to eliminate them.

titlePrefix

Text to add before the list of target names and the proportions of explained variance for each target. This plot title also serves as legend to indicate which target ‘gets’ which each color.

titleVarLabels

Optionally, variable labels to use in the plot title. Has to be the exact same length as targets.

titleSuffix

Text to add after the list of target names and the proportions of explained variance for each target.

fullColorRange

If colors are specified, this can be used to specify which values, for the determinant confidence intervals in the left hand panel, are the minimum and maximum. This is useful if those scores are not actually in the data (e.g. for extremely skewed distributions). If NULL, the range of all individual scores on the determinants is used. For the associations, c(-1, 1) is always used as fullColorRange.

associationsAlpha

The alpha level (transparency) of the confidence interval diamonds in the right hand plot. Value between 0 and 1, where 0 signifies complete transparency (i.e. invisibility) and 1 signifies complete ‘opaqueness’.

returnPlotOnly

Whether to return the entire object that is generated (including all intermediate objects) or only the plot.

drawPlot

Whether the draw the plot, or only return it.

jitterWidth

How much to jitter the data points in the left hand plot.

baseSize

This can be used to efficiently change the size of most plot elements.

dotSize

This is the size of the points used to show the individual data points in the left hand plot.

baseFontSize

This can be used to set the font size separately from the baseSize.

theme

This is the theme that is used for the plots.

xbreaks

Which breaks to use on the X axis (can be useful to override ggplot2’s defaults).

rsq

Whether to compute the R squared values.

...

These arguments are passed on to biAxisDiamondPlot (for the left panel) and diamondPlot (for the right panel). Note that all argument are passed to both those functions.

colors

Colors to use for the two groups in a binary CIBER plot with one (dichotomous) target.

categoryLabels

Labels for the two values of the target.

determinantStructure

When using detStructCIBER, the determinant structure as generated by determinantStructure is included here. determinants, targets, subQuestions, leftAnchors, and rightAnchors are then read from the determinantStructure object. In other words: once a determinantStructure has been generated, only dat and determinantStructure have to be provided as argument to generate a CIBER diamond plot.
Details

Details are explained in Crutzen & Peters (2017).

Value

Depending on the value of returnPlotOnly, either the plot only (a gtable object) or an object containing most objects created along the way (in which case the plot is stored in $output$plot).

The plot has width and height attributes which can be used when saving the plot.

References


See Also
determinantStructure

Examples

### This example uses the determinant study Party Panel 17.1; see ?behaviorchange::BBC_data for more information.

data(BBC_pp17.1);
behaviorchange::CIBER(data=BBC_pp17.1,
    determinants=c('epw_AttExpect_hearingDamage',
                    'epw_AttExpect_highTone',
                    'epw_AttExpect_musicVolume',
                    'epw_AttExpect_musicFidelity',
                    'epw_AttExpect_loudConversation',
                    'epw_AttExpect_musicFocus',
                    'epw_AttExpect_musicEnjoy'),
    targets=c('epw_attitude'));

### With a binary target

data(BBC_pp17.1);
behaviorchange::binaryCIBER(data=BBC_pp17.1,
    determinants=c('epGeneralBeliefs_loudnessPreference',
                   'epGeneralBeliefs_loudnessGenre',
                   'epGeneralBeliefs_loudnessTooMuch',
                   'epGeneralBeliefs_priceFoam',
                   'epGeneralBeliefs_priceSilicon',
                   'epGeneralBeliefs_priceCustom'),
    targets=c('epPossession'),
    categoryLabels = c('no',
                       'yes'));
Description

CIBERlite plots can be used to quickly get an idea of means and correlations of a small number of determinants. They were developed to facilitate conducting and interpreting determinant studies by prevention professionals.

Usage

CIBERlite(
  data,
  determinants,
  targets,
  determinantOrder = NULL,
  determinantLabels = NULL,
  subDeterminantLabels = NULL,
  title = NULL,
  conf.level = 0.95,
  scaleRange = NULL,
  determinantAesthetics = list(fill = "black", color = NA, alpha = 0.5),
  subDeterminantAesthetics = list(fill = "black", color = NA, alpha = 0.5),
  rDiamondAesthetics = list(fill = "#c4c4c4", color = NA, alpha = 0.75)
)

Arguments

data  The dataframe containing the variables.
determinants  Either a character vector with the names of the determinants, or a list of named character vectors, where each vector contains a number of subdeterminants, and each vector's name is the name of the more proximal determinant (i.e. that 'contains' those subdeterminants).
targets  A character vector with the names of the targets (i.e. more proximal determinants, behavior, etc).
determinantOrder  The order in which to display the determinants (if this needs to be different from the order as provided in determinants).
determinantLabels  The labels to use for the determinants.
subDeterminantLabels  The labels to use for the subdeterminants.
title  The title of the plot.
conf.level  The confidence levels: a list with two named values; the confidence level for the means, named means, and the confidence level for the associations, named associations.
scaleRange  The full range of the scale of the determinants/subdeterminants; the minimum and maximum values are used if this is not provided.
determinantAesthetics, subDeterminantAesthetics, rDiamondAesthetics
The aesthetics for the determinants, subdeterminants, and correlation diamonds, each a list containing three named values: fill, color, and alpha.

Details
More details will be provided in a forthcoming paper; until then, see https://CIBERlite.com

Value
A ggplot.

Examples
### This example uses the determinant study Party Panel 15.1; see ?behaviorchange::BBC_data for more information.
data(BBC_pp15.1);
CIBERlite(data=BBC_pp15.1,
determinants=c('highDose_attitude', 'highDose_perceivedNorm', 'highDose_pbc'),
targets=c('highDose_intention'));

Description
COMPLECS was developed to help make sense of complex systems. It reads data from a number of worksheets in a spreadsheet and generates a diagram according to those specifications. Originally, COMPLECS was developed to visualise a problem during the needs assessment phase of intervention development.

Usage
complecs(
  input,
  title = "COMPLECS overview",
  layout = "fdp",
  graph_styling = list(c("outputorder", "edgesfirst", "graph"), c("overlap", "false", "graph"), c("fixedsize", "false", "node"), c("fontname", "Arial", "graph"), c("fontname", "Arial", "node"), c("fontname", "Arial", "edge"), c("headclip", "true", "edge"), c("tailclip", "false", "edge")),
directed = TRUE,
  outputFile = NULL,
outputWidth = 1600,
outputHeight = NULL,
returnDotOnly = FALSE,
returnSvgOnly = FALSE,
returnGraphOnly = TRUE,
maxLabelLength = 20,
regExReplacements = opts$get("diagrammerSanitization"),
silent = opts$get("silent")
)

## S3 method for class 'complecs'
print(
  x,
  width = x$input$width,
  height = x$input$height,
  title = DiagrammeR::get_graph_name(x$output$graph),
  ...
)

## S3 method for class 'complecs'
print(
  x,
  width = x$input$width,
  height = x$input$height,
  title = DiagrammeR::get_graph_name(x$output$graph),
  ...
)

Arguments

input Either a link to a Google Sheet, or a path to an Excel file.
title The title of the COMPLECS graph.
layout The layout to use; has to be one of the DiagrammeR layout types (dot, neato, circo and twopi).
graph_styling Additional styling to apply; a list with three-element vectors, where the three elements correspond to, respectively, the attr, value, and attr_type arguments for DiagrammeR::add_global_graph_attrs(). Note that these attributes may override attributes specified in the COMPLECS specification.
directed Whether to draw directed arrows or not.
outputFile A character vector where each element is one path (including filename) to write the graph to.
outputWidth, outputHeight
If not NULL, a way to override the width and height when calling complecs to generate a COMPLECS overview.
returnDotOnly Whether to only return the produced DOT code.
returnSvgOnly Whether to only return the SVG in a character vector.
returnGraphOnly  Whether to only return the produced graph.
maxLabelLength The number of characters where to wrap the labels.
regExReplacements
   A list of pairs of regular expressions that will be applied to the specifications
   before generating the ABCD. This can be used to sanitize problematic characters
   (e.g. ',', " and ").
silent  Whether to be chatty or silent.
x The object to print (i.e. a result of a call to complecs).
width, height If not NULL, a way to override the width and height when calling print to print
   a COMPLECS overview.
... Any additional arguments for the print() method are passed to DiagrammeR::render_graph().

Details
COMPLECS is a recursive acronym for COMPLECS Organises Multiple Players & Linked Environments using Connected Specifications.

Value
A complecs object that includes the graph and the graph in SVG in output$graph and output$graphSvg.

Examples

```r
## Not run:
### Path in the package with example COMPLECS
exampleCOMPLECS <-
system.file(
   "extdata",
   "COMPLECS-spec-example.xlsx",
   package = "behaviorchange"
);

behaviorchange::complecs(
   exampleCOMPLECS
);

### Loading that COMPLECS from a google sheet - but note that
### this requires an internet connection!
behaviorchange::complecs(
   paste0(
      "https://docs.google.com/spreadsheets/d/",
      "1WMO15xroy4a0Rfpu28GhT-NFDoxxw534w9PrWp8rGjjk"
   )
);

## End(Not run)
```
complecs_to_precede

Represent a COMPLECS specification as a PRECEDE model

Description

This function reads in a complecs specification and draw a PRECEDE model, with a number of assumptions (see Details section).

Usage

complecs_to_precede(
  input,
  title = "PRECEDE diagram",
  layout = "fdp",
  graph_styling = list(c("outputorder", "edgesfirst", "graph"), c("rankdir", "LR", "graph"), c("overlap", "false", "graph"), c("fixedsize", "false", "node"), c("fontname", "Arial", "graph"), c("fontname", "Arial", "node"), c("fillcolor", "White", "node"), c("shape", "box", "node"), c("style", "filled", "node"), c("fontname", "Arial", "edge"), c("headclip", "true", "edge"), c("tailclip", "false", "edge")),
  directed = TRUE,
  outputFile = NULL,
  outputWidth = 1600,
  outputHeight = NULL,
  returnDotOnly = FALSE,
  returnSvgOnly = FALSE,
  returnGraphOnly = TRUE,
  maxLabelLength = 60,
  regExReplacements = opts$g("diagrammerSanitization"),
  silent = opts$g("silent")
)

Arguments

input Either a link to a Google Sheet, or a path to an Excel file.
title The title of the COMPLECS graph.
layout The layout to use; has to be one of the DiagrammeR layout types (dot, neato, circo and two pi).
graph_styling Additional styling to apply; a list with three-element vectors, where the three elements correspond to, respectively, the attr, value, and attr_type arguments for DiagrammeR::add_global_graphattrs(). Note that these attributes may override attributes specified in the COMPLECS specification.
directed Whether to draw directed arrows or not.
outputFile A character vector where each element is one path (including filename) to write the graph to.
complecs_to_precede

outputWidth, outputHeight
If not NULL, a way to override the width and height when calling complecs to
generate a COMPLECS overview.

returnDotOnly  Whether to only return the produced DOT code.
returnSvgOnly   Whether to only return the SVG in a character vector.
returnGraphOnly Whether to only return the produced graph.

maxLabelLength  The number of characters where to wrap the labels.
regExReplacements  A list of pairs of regular expressions that will be applied to the specifications
before generating the ABCD. This can be used to sanitize problematic characters
(e.g. ', " and \).

silent  Whether to be chatty or silent.

Details

Only entities with the following entity types are used from the COMPLECS specification:

• person
• organization
• environmental_condition
• behavior
• determinant
• outcome

Furthermore, it will be assumed that the only direct connections from behavior entities to outcome
entities belong to the focal population; therefore, if behaviors of environmental actors are important
for an outcome, those behaviors’ effects must be represented as environmental_condition enti-
ties - otherwise the relevant persons or organizations will be erroneously considered as focal
population members.

Value

A complecs object that includes the graph and the graph in SVG in output$graph and output$graphSvg.

Examples

## Not run:
### Path in the package with example COMPLECS
exampleCOMPLECS <-
  system.file(
    "extdata",
    "COMPLECS-spec-example.xlsx",
    package = "behaviorchange"
  );

behaviorchange::complecs_to_precede(
  exampleCOMPLECS
### Loading that COMPLECS from a google sheet - but note that
### this requires an internet connection!
behaviorchange::complecs_to_precede(
  paste0(
    "https://docs.google.com/spreadsheets/d/",
    "1WMO15xroy4aQRfpuZ8GhT-NFDOxw534w9PrWp8rGjjk"
  )
);

## End(Not run)

---

**convert.threshold.to.er**

*Visualising Numbers Needed for Change*

---

**Description**

These functions can be used to visualise Numbers Needed for Change (or Numbers Needed to Treat). `erDataSeq` is a helper function to generate an Event Rate Data Sequence, and it uses `convert.threshold.to.er` and `convert.er.to.threshold` to convert thresholds to event rates and vice versa.

**Usage**

```
convert.threshold.to.er(
  threshold,
  mean,
  sd,
  eventIfHigher = TRUE,
  pdist = stats::pnorm
)

convert.er.to.threshold(
  er,
  mean,
  sd,
  eventIfHigher = TRUE,
  qdist = stats::qnorm
)

erDataSeq(
  er = NULL,
  threshold = NULL,
  mean = NULL,
  sd = NULL,
  eventIfHigher = TRUE,
```
pRange = c(1e-06, 0.99999),
   xStep = 0.01
)

ggNNC(
   cerDataSeq,
   d = NULL,
   eventDesirable = TRUE,
   r = 1,
   xlab = "Continuous outcome",
   plotTitle = c("Numbers Needed for Change = ", ""),
   theme = ggplot2::theme_bw(),
   lineSize = 1,
   cerColor = "#EBF2F8",
   eerColor = "#172F47",
   cerLineColor = "#888888",
   eerLineColor = "#000000",
   dArrowColor = "#000000",
   cerAlpha = 0.66,
   eerAlpha = 0.66,
   xLim = NULL,
   xLimAutoDensityTolerance = 0.001,
   showLegend = TRUE,
   verticalLineColor = "#172F47",
   desirableColor = "#00FF00",
   desirableAlpha = 0.2,
   undesirableColor = "#FF0000",
   undesirableAlpha = 0.2,
   desirableTextColor = "#009900",
   undesirableTextColor = "#990000",
   dArrowDistance = 0.04 * max(cerDataSeq$density),
   dLabelDistance = 0.08 * max(cerDataSeq$density)
)

Arguments

threshold If the event rate is not available, a threshold value can be specified instead, which
   is then used in conjunction with the mean (mean) and standard deviation (sd) and
   assuming a normal distribution to compute the event rate.
mean The mean of the control group distribution.
sd The standard deviation (of the control distribution, but assumed to be the same
   for both distributions).
eventIfHigher Whether scores above or below the threshold are considered 'an event'.
pdist, qdist Distributions to use when converting thresholds to event rates and vice versa;
   defaults to the normal distribution.
er Event rate to visualise (or convert).
pRange The range of probabilities for which to so the distribution.
xStep

cerDataSeq

d

eventDesirable

r

xlab

plotTitle

theme

lineSize

cerColor

eerColor

cerLineColor

eerLineColor

dArrowColor

cerAlpha

eerAlpha

xLim

xLimAutoDensityTolerance

showLegend

verticalLineColor

desirableColor

desirableAlpha

undesirableColor

undesirableAlpha

desirableTextColor

undesirableTextColor

dArrowDistance

dLabelDistance

Details

These functions are used by `nnc()` to show the distributions, and event rates. They probably won’t be used much on their own.
Value

`erDataSeq` returns a data sequence: `ggNNC` a `ggplot2::ggplot()`.

Author(s)

Gjalt-Jorn Peters & Stefan Gruijters
Maintainer: Gjalt-Jorn Peters gjalt-jorn@userfriendlyscience.com

References


See Also

`nnc()`

Examples

```r
### Show distribution for an event rate value of 125
behaviorchange::ggNNC(behaviorchange::erDataSeq(threshold=125, mean=90, sd=30));

### If the event occurs under the threshold instead of above it
behaviorchange::ggNNC(behaviorchange::erDataSeq(threshold=125, mean=90, sd=30,
   eventIfHigher = FALSE));

### ... And for undesirable events (note how desirability is an argument for ggNNC, whereas whether an event occurs 'above' or 'below' the threshold is an argument for erDataSeq):
behaviorchange::ggNNC(behaviorchange::erDataSeq(threshold=125, mean=90, sd=30,
   eventIfHigher = FALSE),
   eventDesirable = FALSE);

### Show event rate for both experimental and control conditions, and show the numbers needed for change
behaviorchange::ggNNC(behaviorchange::erDataSeq(threshold=125, mean=90, sd=30),
   d=.5);

### Illustration of how even with very large effect sizes, if the control event rate is very high, you'll still need a high number of NNC
behaviorchange::ggNNC(behaviorchange::erDataSeq(er=.9),
   d=1);
```
determinantStructure  Determinant Structure specification

Description

These functions can be used to specify a determinant structure: a hierarchical structure of determinants that can then be conveniently plotted and analysed, for example using detStructCIBER. These functions are made to be used together; see the example and the forthcoming article for more information.

Usage

determinantStructure(name, selection = NULL, ...)
determinantVar(name, selection = NULL, ...)
subdeterminants(name, selection = NULL, ...)
subdeterminantProducts(name, selection = NULL, ...)

## S3 method for class 'determinantStructure'
plot(x, useDiagrammeR = FALSE, ...)

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

Arguments

name  The name of the variable that is specified.
selection  A regular expression to use to select the variables in a dataframe that are considered items that together form this variable. For determinantStructure, a list can be provided that also contains a named regular expression with the name 'behaviorRegEx', which specifies the name of the behavior to which this determinant structure pertains.
...
...  Any additional arguments are other determinant structure building functions. These are used to construct the determinant structure 'tree'.
x  The determinantStructure object to print or plot.
useDiagrammeR  Whether to simply use print(plot(x)) (if FALSE) or whether to use data.tree::ToDiagrammeRGraph, tweak it a bit, by setting global graph attributes, and then using DiagrammeR::render_graph (if TRUE).

Details

This family of functions will be explained more in detail in a forthcoming paper. plot and print methods plot and print a determinantStructure object.
Value

A `determinantStructure` object, which is a `data.tree` object.

Author(s)

Gjalt-Jorn Peters, <gjalt-jorn@a-bc.eu>

See Also

`detStructAddVarLabels`, `detStructAddVarNames`, `detStructComputeProducts`, `detStructComputeScales`, `detStructCIBER`

Examples

determinantStructure('using R',
  list('using R',
     behaviorRegEx = 'some RegEx'),
  determinantVar("Intention",
     "another RegEx",
     determinantVar("Attitude",
        "third RegEX",
        subdeterminants("Likelihood",
          "4th RegEx"),
        subdeterminants("Evaluation",
          "5th RegEx"),
        subdeterminantProducts("attProduct",
          c("4th RegEx",
          "5th RegEx"))),
     determinantVar("perceivedNorm",
        "6th RegEx",
        subdeterminants("Approval",
          "7th RegEx"),
        subdeterminants("Motivation to comply",
          "8th RegEx"),
        subdeterminantProducts("normProduct",
          c("7th RegEx",
          "8th RegEx"))),
     determinantVar("pbc",
        "9th RegEx",
        subdeterminants("Control beliefs",
          "10th RegEx"))));
**determinant_selection_table**

**Description**

These functions compute the Potential for Change Index for one or multiple (sub-)determinants, the room for improvement (an intermediate estimate), and produce a convenient table with an overview of all (sub-)determinants. Note that for determinant selection purposes, quantitative estimates such as the Potential for Change Index should never be used without also thoroughly inspecting the visualisations of the univariate distributions and the confidence intervals for the associations to the ultimate intervention targets (usually the target behavior or a proxy measure). For this purpose, the Confidence Interval-Based Estimation of Relevance plots can be used (see `CIBER()`).

**Usage**

determinant_selection_table(
  data,
  determinants,
  target,
  determinantLabels = NULL,
  targetLabel = NULL,
  sortBy = NULL,
  sortByAbs = TRUE,
  decreasing = TRUE,
  digits = 3,
  increasesAreImprovements = TRUE,
  minimum = base::min,
  maximum = base::max,
  center = base::mean,
  weight = stats::cor,
  type = NULL,
  minimumArgs = list(na.rm = TRUE),
  maximumArgs = list(na.rm = TRUE),
  centerArgs = list(na.rm = TRUE),
  weightArgs = list(use = "complete.obs"),
  potentialScale = NULL,
  headingLevel = 3,
  output = behaviorchange::opts$get("tableOutput")
)

determinantSelectionTable_partial(
  x,
  digits = attr(x, "digits"),
  headingLevel = attr(x, "headingLevel"),
  echoPartial = FALSE,
  partialFile = NULL,
  quiet = TRUE,
  ...
)

## S3 method for class 'determinantSelectionTable'
knit_print(}
x,
digits = attr(x, "digits"),
headingLevel = attr(x, "headingLevel"),
echoPartial = FALSE,
partialFile = NULL,
quiet = TRUE,
...
)

## S3 method for class 'determinantSelectionTable'
print(
    x,
digits = attr(x, "digits"),
headingLevel = attr(x, "headingLevel"),
output = attr(x, "output"),
forceKnitrOutput = FALSE,
...
)

potential_for_change_index(
data,
determinants,
target,
increasesAreImprovements = TRUE,
sampleLevel = FALSE,
minimum = base::min,
maximum = base::max,
center = base::mean,
weight = stats::cor,
type = NULL,
minimumArgs = list(na.rm = TRUE),
maximumArgs = list(na.rm = TRUE),
centerArgs = list(na.rm = TRUE),
weightArgs = list(use = "complete.obs")
)

room_for_improvement(
    x,
increasesAreImprovements = TRUE,
sampleLevel = FALSE,
minimum = base::min,
maximum = base::max,
center = base::mean,
minimumArgs = list(na.rm = TRUE),
maximumArgs = list(na.rm = TRUE),
centerArgs = list(na.rm = TRUE),
varName = NULL
)
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>The dataframe containing the variables.</td>
</tr>
<tr>
<td>determinants</td>
<td>The name(s) of the determinant(s).</td>
</tr>
<tr>
<td>target</td>
<td>The target (e.g. behavior or intention).</td>
</tr>
<tr>
<td>determinantLabels, targetLabel</td>
<td>Optionally, labels to use for the (sub-)determinants and the target. The determinantLabels must have the same order as the determinants vector.</td>
</tr>
<tr>
<td>sortBy</td>
<td>The column to sort the results by; if not NULL, a number from 1-6 that corresponds to the six columns of the Determinant Selection Table.</td>
</tr>
<tr>
<td>sortByAbs</td>
<td>Whether to sort by raw values (FALSE) or their absolute value (TRUE).</td>
</tr>
<tr>
<td>decreasing</td>
<td>Whether to sort in decreasing (TRUE) or increasing (FALSE) order.</td>
</tr>
<tr>
<td>digits</td>
<td>The number of digits to round to.</td>
</tr>
<tr>
<td>increasesAreImprovements</td>
<td>Whether increases are improvements (TRUE) or decreases are improvements (FALSE).</td>
</tr>
<tr>
<td>minimum, maximum</td>
<td>The minimum and maximum, as functions that take a vector and return the minimum and maximum scores, as numbers, or as vectors of numbers specifying the minimum and maximum to use for each column in x or determinants, or a lists of functions specifying the functions to use for each column in x or determinants.</td>
</tr>
<tr>
<td>center</td>
<td>For the sample-level version, a function that takes a vector and returns the center (e.g. mean, median, etc), or a list of functions specifying the function to use for each column in x or determinants.</td>
</tr>
<tr>
<td>weight</td>
<td>The function to return the weight/multiplier to use in the computation.</td>
</tr>
<tr>
<td>type</td>
<td>The type of potential for change index. Currently implemented are type '1' and type '2' - see details for more information.</td>
</tr>
<tr>
<td>minimumArgs, maximumArgs, centerArgs, weightArgs</td>
<td>lists with arguments to pass to the corresponding functions. Note that these are not vectorized.</td>
</tr>
<tr>
<td>potentialScale</td>
<td>The scale with minimum and maximum possible values for the Potential for Change index. If NULL, the minimum is set to 0 and the maximum is set to the highest observed value.</td>
</tr>
<tr>
<td>headingLevel</td>
<td>The number of hashes to print in front of the headings when printing while knitting.</td>
</tr>
<tr>
<td>output</td>
<td>Whether to only output to the viewer (if possible; output='viewer'), or only to the console (output='console'), or to both (output=c('viewer', 'console')). Note that displaying in the viewer requires the htmltools package.</td>
</tr>
<tr>
<td>x</td>
<td>For room for improvement, either a numeric vector with scores on a (sub-)determinant, or a data frame with multiple such vectors. For the Determinant Selection Table functions, the object to print/knit.</td>
</tr>
<tr>
<td>echoPartial</td>
<td>Whether to show the executed code in the R Markdown partial (TRUE) or not (FALSE).</td>
</tr>
</tbody>
</table>
partialFile This can be used to specify a custom partial file. The file will have object \( x \) available.

quiet Passed on to `knitr::knit()` whether it should be chatty (FALSE) or quiet (TRUE).

... Any additional arguments are passed to the default print method by the print method, and to `rmdpartials::partial()` when knitting a RMarkdown partial.

forceKnitrOutput Force knit output.

campleLevel Whether to return sample-level estimates (TRUE) or individual-level estimates (FALSE).

varName For internal use.

Details

The Potential for Change index was developed by Keegan et al. and is a numerical representation of a number of important features in `CIBER()` plots (for more details, please see the references below). It turned out a similar measure, the Intervention Potential, was developed by Huber & Mosler (2013). The latter uses regression coefficients as weights, which is problematic for a number of reasons (see Crutzen, Peters & Noijen, 2017), and has therefore not been implemented as a default, but it is possible to use regression coefficients by specifying a custom weight function.

The original Potential for Change Index was conceptualized to optimize intervention tailoring and improve the prediction of individual-level intervention effectiveness. A second conceptualization of the Potential for Change Index can facilitate sub-determinant selection.

In addition to using the minimum, maximum, center, and weight functions to specify custom functions, specific types have also been implemented to quickly use a prespecified combination of functions.

The first (type = '1') is computed as follows:

- For sub-determinants with a positive zero-order correlation with behavior, the sample mean is subtracted from the observed maximum score, and the result is multiplied by the zero-order correlation;
- For sub-determinants with a negative zero-order correlation with behavior, the sample mean is subtracted from the observed minimum score, and the result is multiplied by the zero-order correlation.

The second (type = '2') is computed as follows:

- For sub-determinants with a positive zero-order correlation with behavior, the sample mean is subtracted from the .95 quantile of the scores, and the result is multiplied by the squared zero-order correlation (i.e. the proportion of explained variance);
- For sub-determinants with a negative zero-order correlation with behavior, the sample mean is subtracted from the .05 quantile of the scores, and the result is multiplied by the squared zero-order correlation (i.e. the proportion of explained variance);

The second variant effectively takes the 5% trimmed maximum and minimum, rendering it less sensitive to outliers, penalizes weak associations with behavior more severely, and decreases sensitivity to differences between correlations. These differences should render the second variant a bit more robust over different samples.
The room for improvement is one of the ingredients of the Potential for Change Index or P_delta, a generalized version of the Intervention Potential. The Determinant Selection Table efficiently presents the Potential for Change Indices for a set of (sub-)determinants.

Value

For the individual-level version, a vector or data frame with the same dimensions as provided; for the sample-level version, if a vector is provided, a single number, and if a data frame is provided, a vector with as many values as the data frame has columns. For Determinant Selection Table, a data frame.

References


Examples

```r
### Get example data
dat <- get(data("BBC_pp15.1", package="behaviorchange"));

### Individual-level version, for one sub-determinant
P_delta_example <-
  behaviorchange::potential_for_change_index(
    data=dat,
    determinants=c('highDose_attitude'),
    target='highDose_intention'
 );

head(P_delta_example);
hist(P_delta_example);

### Sample-level version
behaviorchange::potential_for_change_index(
  data=dat,
  determinants=c('highDose_attitude'),
  target='highDose_intention',
  sampleLevel = TRUE
);

### Individual-level for multiple determinants
P_delta_example <-
  behaviorchange::potential_for_change_index(
    data=dat,
    determinants=c('highDose_attitude', 'highDose_perceivedNorm'),
    target='highDose_intention'
  );
```
head(P_delta_example);

### Sample-level version for multiple determinants
behaviorchange::potential_for_change_index(
  data=dat,
  determinants=c('highDose_attitude', 'highDose_perceivedNorm'),
  target='highDose_intention',
  sampleLevel = TRUE
);

### Get the Potential for Change Index Type 2
behaviorchange::potential_for_change_index(
  data=dat,
  determinants=c('highDose_attitude', 'highDose_perceivedNorm'),
  target='highDose_intention',
  type = '2',
  sampleLevel = TRUE
);

### Get a Determinant Selection Table
behaviorchange::determinant_selection_table(
  data=dat,
  determinants = c('highDose_AttBeliefs_long',
                   'highDose_AttBeliefs_intensity',
                   'highDose_AttBeliefs_euphoria'),
  target = 'highDose_intention',
  sortBy = 6
);

### Room for improvement for one variable
head(
  room_for_improvement(
    dat$highDose_AttBeliefs_long
  )
);

room_for_improvement(
  dat$highDose_AttBeliefs_long,
  sampleLevel = TRUE
);
### For multiple (sub-)determinants

```r
head(
  room_for_improvement(
    dat[, c('highDose_AttBeliefs_long',
             'highDose_AttBeliefs_intensity',
             'highDose_AttBeliefs_euphoria')]
  )
);

room_for_improvement(
  dat[, c('highDose_AttBeliefs_long',
             'highDose_AttBeliefs_intensity',
             'highDose_AttBeliefs_euphoria'),
       sampleLevel = TRUE]
);
```

---

**detStructAddVarLabels**  
*Functions to preprocess determinant structures*

**Description**

These functions are used in conjunction with the `determinantStructure` family of functions to conveniently work with determinant structures.

**Usage**

```r
detStructAddVarLabels(
  determinantStructure,
  varLabelDf,
  varNameCol = "varNames.cln",
  leftAnchorCol = "leftAnchors",
  rightAnchorCol = "rightAnchors",
  subQuestionCol = "subQuestions",
  questionTextCol = "questionText"
)

detStructAddVarNames(determinantStructure, names)

detStructComputeProducts(determinantStructure, data, append = TRUE)

detStructComputeScales(
  determinantStructure, data,
  append = TRUE,
  separator = "_"
)
```
detStructAddVarLabels

Arguments

determinantStructure
  The determinantStructure object.
varLabelDf
  The variable label dataframe as generated by the processLSvarLabels in the userfriendlyscience package. It is also possible to specify a 'homemade' dataframe, in which case the column names have to specified (see the next arguments).
varNameCol
  The name of the column of the varLabelDf that contains the variable name. Only needs to be changed from the default value if varLabelDf is not a dataframe as produced by processLSvarLabels.
leftAnchorCol
  The name of the column of the varLabelDf that contains the left anchor. Only needs to be changed from the default value if varLabelDf is not a dataframe as produced by processLSvarLabels.
rightAnchorCol
  The name of the column of the varLabelDf that contains the right anchor. Only needs to be changed from the default value if varLabelDf is not a dataframe as produced by processLSvarLabels.
subQuestionCol
  The name of the column of the varLabelDf that contains the subquestion. Only needs to be changed from the default value if varLabelDf is not a dataframe as produced by processLSvarLabels.
questionTextCol
  The name of the column of the varLabelDf that contains the question text. Only needs to be changed from the default value if varLabelDf is not a dataframe as produced by processLSvarLabels.
names
  A character vector with the variable names. These are matched against the regular expressions as specified in the determinantStructure object, and any matches will be stored in the determinantStructure object.
data
  The dataframe containing the data; the variables names specified in names (when calling detStructAddVarNames) must be present in this dataframe.
append
  Whether to only return the products or scales, or whether to append these to the dataframe and return the entire dataframe.
separator
  The separator to use when constructing the scale variables names.

Details

This family of functions will be explained more in detail in a forthcoming paper.

Value

detStructAddVarLabels and detStructAddVarNames just change the determinantStructure object; detStructComputeProducts and detStructComputeScales return either the dataframe with the new variables appended (if append = TRUE) or just a dataframe with the new variables (if append = FALSE).

References

(Forthcoming)
See Also
determinantStructure, determinantVar, subdeterminants, subdeterminantProducts, detStructCIBER

Examples

```r
### Create some bogus determinant data
detStudy <- mtcars[, c(1, 3:7)];
names(detStudy) <- c('rUse_behav',
                      'rUse_intention',
                      'rUse_attitude1',
                      'rUse_attitude2',
                      'rUse_expAtt1',
                      'rUse_expAtt2');

### Specify the determinant structure

### First a subdeterminant
expAtt <-
  behaviorchange::subdeterminants("Subdeterminants",
                                 "expAtt");

### Then two determinants
attitude <-
  behaviorchange::determinantVar("Determinant",
                                 "attitude",
                                 expAtt);

intention <-
  behaviorchange::determinantVar("ProximalDeterminant",
                                 "intention",
                                 attitude);

### Then the entire determinant structure
detStruct <-
  behaviorchange::determinantStructure('Behavior',
                                        list('behav',
                                             behaviorRegEx = 'rUse'),
                                        intention);

### Add the variable names
behaviorchange::detStructAddVarNames(detStruct,
                                       names(detStudy));

### Add the determinant scale variable to the dataframe
detStudyPlus <-
  behaviorchange::detStructComputeScales(detStruct,
                                        data=detStudy);

### Show its presence
names(detStudyPlus);
mean(detStudyPlus$rUse_Determinant);
```
**dMCD**  
*Estimate Cohen’s d corresponding to a Meaningful Change Definition*

**Description**

This function uses a base rate (Control Event Rate, argument `cer`) and a Meaningful Change Definitions (MCD, argument `mcd`) to compute the corresponding Cohen’s d. See Gruijters & Peters (2019) for details.

**Usage**

```r
dMCD(
  cer, 
  mcd = NULL, 
  eer = NULL, 
  plot = TRUE, 
  mcdOnX = FALSE, 
  plotResultValues = TRUE, 
  resultValueLineColor = "blue", 
  resultValueLineSize = 1, 
  returnLineLayerOnly = FALSE, 
  theme = ggplot2::theme_bw(), 
  highestPossibleEER = 0.999999999, 
  xLab = ifelse(mcdOnX, "Meaningful Change Definition", "Control Event Rate"), 
  yLab = "Cohen's d", 
  dist = "norm", 
  distArgs = list(), 
  distNS = "stats", 
  ...
)
```

## S3 method for class 'dMCD'

print(x, ...)

**Arguments**

- **cer**
  
The Control Event Rate (or base rate): how many people already perform the target behavior in the population (as a proportion)?

- **mcd**
  
The Meaningful Change Definitions: by which percentage (as a proportion) should the event rate increase to render an effect meaningful?

- **eer**
  
Instead of the MCD, it is also possible to specify the Experimental Event Rate (EER), in which case the MCD is computed by taking the difference with the CER.

- **plot**
  
Whether to show a plot.

- **mcdOnX**
  
Whether to plot the Meaningful Change Definition on the X axis (by default, the CER is plotted on the X axis).
lm_rSq_ci

plotResultValues
Whether to plot the result values.

resultValueLineColor, resultValueLineSize
If plotting the result values, lines of this color and size are used.

returnLineLayerOnly
Whether to only return a layer with the plotted line (which can be used to quickly stack lines for different MCDs).

theme
The ggplot2 theme to use.

highestPossibleEER
The highest possible EER to include in the plot.

xLab, yLab
The labels for the X and Y axes.

dist, distArgs, distNS
Used to specify the distribution to use to convert between Cohen’s d and the CER and EER. distArgs can be used to specify additional arguments to the corresponding q and p functions, and distNS to specify the namespace (i.e. package) from where to get the distribution functions.

... Any additional arguments to dMCD are passed on to the ggplot2::geom_line used to draw the line showing the different Cohen’s d values as a function of the base rate (or MCD) on the X axis. Additional arguments for the print method are passed on to the default print method.

x
The object to print (i.e. a result from a call to dMCD).

Value
The Cohen’s d value, optionally with a ggplot2 plot stored in an attribute (which is only a ggplot2 layer if returnLineLayerOnly=TRUE).

References

Examples
dMCD(.2, .05);

---

lm_rSq_ci

*Obtaining an R squared confidence interval estimate for an lm regression*

Description
The lm_rSq_ci function uses the base R lm function to conduct a regression analysis and then computes the confidence interval for R squared.
Usage

```r
lm_rSq_ci(
  formula, 
  data = NULL, 
  conf.level = 0.95, 
  ci.method = c("widest", "r.con", "olkinfinn"),
  env = parent.frame()
)
```

Arguments

- **formula**: The formula of the regression analysis, of the form `y ~ x1 + x2`, where `y` is the dependent variable and `x1` and `x2` are the predictors.
- **data**: If the terms in the formula aren't vectors but variable names, this should be the dataframe where those variables are stored.
- **conf.level**: The confidence of the confidence interval around the regression coefficients.
- **ci.method**: Which method to use for the confidence interval around R squared.
- **env**: The environment where to evaluate the formula.

Value

The confidence interval

Author(s)

Gjalt-Jorn Peters
Maintainer: Gjalt-Jorn Peters gjalt-jorn@a-bc.eu

Examples

```r
### Do a simple regression analysis
lm_rSq_ci(age ~ circumference, dat=Orange);
```

---

### nnc

**Numbers Needed for Change**

**Description**

This function computes the Numbers Needed for Change, and shows a visualisation to illustrate them. Numbers Needed for Change is the name for a Numbers Needed to Treat estimate that was computed for a continuous outcome as is common in behavior change research.
Usage

\[
nnc(  
  d = \text{NULL},  
  cer = \text{NULL},  
  r = 1,  
  n = \text{NULL},  
  \text{threshold} = \text{NULL},  
  \text{mean} = 0,  
  \text{sd} = 1,  
  \text{poweredFor} = \text{NULL},  
  \text{thresholdSensitivity} = \text{NULL},  
  \text{eventDesirable} = \text{TRUE},  
  \text{eventIfHigher} = \text{TRUE},  
  \text{conf.level} = 0.95,  
  \text{dReliability} = 1,  
  d.\text{ci} = \text{NULL},  
  cer.\text{ci} = \text{NULL},  
  r.\text{ci} = \text{NULL},  
  d.n = \text{NULL},  
  cer.n = \text{NULL},  
  r.n = \text{NULL},  
  \text{plot} = \text{TRUE},  
  \text{returnPlot} = \text{TRUE},  
  \text{silent} = \text{FALSE}  
)  
\]

## S3 method for class 'nnc'

print(x, digits = 2, ...)

Arguments

d                  The value of Cohen’s \(d\).
cer                The Control Event Rate.
r                  The correlation between the determinant and behavior (for mediated Numbers Needed for Change).
n                  The sample size.
threshold          If the event rate is not available, a threshold value can be specified instead, which is then used in conjunction with the mean (mean) and standard deviation (sd) and assuming a normal distribution to compute the event rate.
mean               The mean value, used to draw the plot, or, if no CER is provided but instead the threshold value, to compute the CER.
sd                 The standard deviation, used to draw the plot (and to compute the CER if a threshold value is supplied instead of the CER).
poweredFor         The Cohen’s \(d\) value for which the study was powered. This expected Cohen’s \(d\) value can be used to compute the threshold, which then in turn is used to compute the CER. To use this approach, also specify the mean and the standard deviation.
thresholdSensitivity
This argument can be used to provide a vector of potential threshold values, each
of which is used to compute an NNC. This enables easy inspection of whether
the value chosen as threshold matters much for the NNC.

eventDesirable
Whether an event is desirable or undesirable.
eventIfHigher
Whether scores above or below the threshold are considered 'an event'.
conf.level
The confidence level of the confidence interval.
dReliability
If Cohen’s d was not measured with perfect reliability, nnc can disattenuate it
to correct for the resulting attenuation using ufs::disattenuate.d() before
computating the Experimental Event Rate. Use this argument to specify the
reliability of the outcome measure. By default, the setting of 1 means that no
disattenuation is applied.
d.ci
Instead of providing a point estimate for Cohen’s d, a confidence interval can be
provided.
cer.ci
Instead of providing a point estimate for the Control Event Rate, a confidence
interval can be provided.
r.ci
Instead of providing a point estimate for the correlation, a confidence interval
can be provided.
d.n
In addition to providing a point estimate for Cohen’s d, a sample size can be
provided; if it is, the confidence interval is computed.
cer.n
In addition to providing a point estimate for the Control Event Rate, a sample
size can be provided; if it is, the confidence interval is computed.
r.n
In addition to providing a point estimate for the correlation, a sample size can
be provided; if it is, the confidence interval is computed.
plot
Whether to generate and show the plot.
returnPlot
Whether to return the plot (as an attribute), or to only display it.
silent
Whether to suppress notifications.
x
The nnc object to print.
digits
The number of digits to round to.
...
Any additional arguments are passed to the print function.

Details
Numbers Needed to Treat is a common and very useful effect size measure in use in the medical
sciences. It is computed based on the Control Event Rate (CER) and the Experimental Event Rate
(EER), and expresses how many people would need to received a treatment to yield a beneficial
result for one person. In behavior change research, a similar measure would be useful, but the
outcome is normally not dichotomous as is common in the medical literature (i.e. whether a par-
ticipants survives or is cured), but continuous. Numbers Needed for Change fills this lacuna: it is
simply the Numbers Needed to Treat, but converted from a Cohen’s d value. nnt is an alias for nnc.
For more details, see Gruijters & Peters (2019) for details.

Value
The Numbers Needed for Change (NNC), potentially with a plot visualising the NNC in an attribute.
Author(s)
Gjalt-Jorn Peters & Stefan Gruijters
Maintainer: Gjalt-Jorn Peters gjalt-jorn@userfriendlyscience.com

References

Examples

```r
### Simple example
behaviorchange::nnc(d=.4, cer=.3);

### Or for a scenario where events are undesirable, and the intervention effective (therefore having a negative value for d):
behaviorchange::nnc(d=-.4, cer=.3, eventDesirable=FALSE);
```

Description
The `behaviorchange::opts` object contains three functions to set, get, and reset options used by the `escalc` package. Use `behaviorchange::opts$set` to set options, `behaviorchange::opts$get` to get options, or `behaviorchange::opts$reset` to reset specific or all options to their default values.

Usage
```r
opts
```

Format
An object of class `list` of length 4.

Details
It is normally not necessary to get or set `behaviorchange` options.

The following arguments can be passed:

```r
... For behaviorchange::opts$set, the dots can be used to specify the options to set, in the format option = value, for example, EFFECTSIZE_POINTESTIMATE_NAME_IN_DF = "\n". For behaviorchange::opts$reset, a list of options to be reset can be passed.

option For behaviorchange::opts$set, the name of the option to set.
```
**default**  For `behaviorchange::opts$get`, the default value to return if the option has not been manually specified.

To see the full list of options and their default values, use `behaviorchange::opts$default()`. Some examples are:

- **aabbcc**  A color theme for `abcd()`.
- **complecs_***  The worksheet and columns names for `complecs()`.
- **silent**  Whether to be chatty or silent.

**Examples**

```r
### Get the default utteranceMarker
behaviorchange::opts$get(complecs_entitySheet);

### Set it to a custom version, so that every line starts with a pipe
behaviorchange::opts$set(complecs_entitySheet = "sheet_with_entities");

### Check that it worked
behaviorchange::opts$get(complecs_entitySheet);

### Reset this option to its default value
behaviorchange::opts$reset(complecs_entitySheet);

### Check that the reset worked, too
behaviorchange::opts$get(complecs_entitySheet);
```

---

**partypanelData**  *Subsets of Party Panel datasets*

**Description**

These are subsets of Party Panel datasets. Party Panel is an annual semi-panel determinant study among Dutch nightlife patrons, where every year, the determinants of another nightlife-related risk behavior are mapped.

**Usage**

```r
data(BBC_pp15.1)
data(BBC_pp16.1)
data(BBC_pp17.1)
data(BBC_pp18.1)
```
Format

For BBC_pp15.1, a data.frame with 123 columns and 829 rows. For BBC_pp16.1, a data.frame with 63 columns and 1077 rows. For BBC_pp17.1, a data.frame with 94 columns and 943 rows. For BBC_pp18.1, a data.frame with 84 columns and 880 rows. Note that many rows contain missing values; the columns and rows were taken directly from the original Party Panel datasets, and represent all participants that made it past a given behavior.

Details

The behaviors of the Party Panel waves were:

- 2015: Behaviors related to using highly dosed ecstasy pills
- 2016: Behaviors related to visiting nightlife first-aid facilities
- 2017: Behaviors related to hearing protection
- 2018: Behaviors related to flirting and boundary crossing
- 2019: Behaviors related to sleeping hygiene surrounding nightlife participation

The full datasets are publicly available through the Open Science Framework (https://osf.io/s4fmu/). Also see the GitLab repositories (https://gitlab.com/partypanel) and the website at https://partypanel.eu.

Examples

data('BBC_pp17.1', package='behaviorchange');
behaviorchange::CIBERlite(data=BBC_pp17.1,
determinants=c("epw_attitude",
"epw_perceivedNorm",
"epw_pbc",
"epw_habit"),
targets=c("epw_intention"));

pies

Practically Important Effect Sizes

Description

Practically Important Effect Sizes

Usage

pies(
data = NULL,
controlCol = NULL,
expCol = NULL,
d = NULL,
cer = NULL,
r = 1,
n = NULL,
repeatStr = NULL,
mean = 0,
sd = 1,
bootstrapA = FALSE,
conf.level = 0.95
)

Arguments

data: Optionally, if you want to get A, a data frame.
controlCol, expCol: Optionally, if you want to get A, the names of the columns with control and experimental data.
d: Cohen’s d.
cer: The control even rate (see nnt()).
r, threshold, mean, sd: Arguments for the nnt() function.
n: The sample size.
bootstrapA: Whether to use bootstrapping to compute A.
conf.level: The confidence level of confidence intervals.

Value

A dataframe with all values.

Examples

pies(d = .5, n = 100, cer = .2, threshold = 2);

repeatStr(n = 1, str = " ")

Description

Repeat a string a number of times

Usage

repeatStr(n = 1, str = " ")

Arguments

n, str: Normally, respectively the frequency with which to repeat the string and the string to repeat; but the order of the inputs can be switched as well.
vecTxt

Value

A character vector of length 1.

Examples

#### 10 spaces:
repStr(10);

#### Three euro symbols:
repStr("\u20ac", 3);

---

vecTxt

Easily parse a vector into a character value

Description

Easily parse a vector into a character value

Usage

vecTxt(
  vector,
  delimiter = "",",
  useQuote = "\",
  firstDelimiter = NULL,
  lastDelimiter = " & ",
  firstElements = 0,
  lastElements = 1,
  lastHasPrecedence = TRUE
)

vecTxtQ(vector, useQuote = "", ...)

Arguments

vector

The vector to process.

delimiter, firstDelimiter, lastDelimiter

The delimiters to use for respectively the middle, first firstElements, and last lastElements elements.

useQuote

This character string is pre- and appended to all elements; so use this to quote all elements (useQuote=""""), doublequote all elements (useQuote='"'), or anything else (e.g. useQuote='\'|'). The only difference between vecTxt and vecTxtQ is that the latter by default quotes the elements.

firstElements, lastElements

The number of elements for which to use the first respective last delimiters
lastHasPrecedence

If the vector is very short, it’s possible that the sum of firstElements and lastElements is larger than the vector length. In that case, downwardly adjust the number of elements to separate with the first delimiter (TRUE) or the number of elements to separate with the last delimiter (FALSE)?

... Any addition arguments to vecTxtQ are passed on to vecTxt.

Value

A character vector of length 1.

Examples

 vecTxtQ(names(mtcars));

wrapVector

Wrap all elements in a vector

Description

Wrap all elements in a vector

Usage

wrapVector(x, width = 0.9 * getOption("width"), sep = "\n", ...)

Arguments

x The character vector
width The number of
sep The glue with which to combine the new lines
... Other arguments are passed to strwrap().

Value

A character vector

Examples

 res <- wrapVector( c( "This is a sentence ready for wrapping", "So is this one, although it’s a bit longer" ), width = 10 );

 print(res);
 cat(res, sep="\n");
Index

* datasets
  opts, 41
* data
  abcd_specs_examples, 7
  partypanelData, 42
* hplot
  CIBER, 9
* utilities
  convert.threshold.to.er, 21
  detStructAddVarLabels, 33
  lm_rSq_ci, 37
  nnc, 38
abcd, 3, 7
abcd(), 42
abcd_specification_empty
  (abcd_specs_examples), 7
abcd_specification_example_xtc
  (abcd_specs_examples), 7
abcd_specs_complete
  (abcd_specs_examples), 7
abcd_specs_dutch_xtc
  (abcd_specs_examples), 7
abcd_specs_examples, 7
abcd_specs_single_po_without_conditions
  (abcd_specs_examples), 7
abcd_specs_without_conditions
  (abcd_specs_examples), 7
apply_graph_theme, 8
apply_graph_theme(), 4
BBC_data (partypanelData), 42
BBC_pp15.1 (partypanelData), 42
BBC_pp16.1 (partypanelData), 42
BBC_pp17.1 (partypanelData), 42
BBC_pp18.1 (partypanelData), 42
biAxisDiamondPlot, 13
binaryCIBER (CIBER), 9
cat, 9
cat0, 9
CIBER, 5, 9
CIBER(), 27, 30
CIBERlite, 15
complecs, 16
complecs(), 42
complecs_to_precede, 19
convert.er.to.threshold
  (convert.threshold.to.er), 21
convert.threshold.to.er, 21
data.tree, 26
data.tree::DiagrammeRGraph, 25
determinant_selection_table, 26
determinantSelectionTable_partial
  (determinant_selection_table), 26
determinantStructure, 13, 14, 25, 33–35
determinantVar, 35
determinantVar (determinantStructure), 25
detStructAddVarLabels, 26, 33
detStructAddVarNames, 26
detStructAddVarNames
  (detStructAddVarLabels), 33
detStructCIBER, 25, 26, 35
detStructCIBER (CIBER), 9
detStructComputeProducts, 26
detStructComputeProducts
  (detStructAddVarLabels), 33
detStructComputeScales, 26
detStructComputeScales
  (detStructAddVarLabels), 33
detStructPreprocessing
  (detStructAddVarLabels), 33
DiagrammeR::add_global_graph_attrs(), 17, 19
DiagrammeR::DiagrammeR, 4, 6, 8
DiagrammeR::export_graph, 4
DiagrammeR::render_graph, 25
DiagrammeR::render_graph(), 5, 18
DiagrammeRsvg::export_svg(), 4
diamondPlot, 13
dMCD, 36
erDataSeq(convert.threshold.to.er), 21
g (opts), 41
ggNNC(convert.threshold.to.er), 21
ggplot2, 13
ggplot2::ggplot(), 24
ggplot2::ggsave(), 12
gtable, 14
intervention_potential
  (determinant_selection_table), 26
knit_print.determinantSelectionTable
  (determinant_selection_table), 26
knitr::knit(), 30
lm_rSq.ci, 37
nnc, 38
nnc(), 23, 24
nncvis(convert.threshold.to.er), 21
nnt(nnc), 38
nnt(), 44
opts, 41
P_delta(determinant_selection_table), 26
partyPanelData, 42
pies, 43
plot.determinantStructure
  (determinantStructure), 25
potential_for_change_index
  (determinant_selection_table), 26
print(), 18
print.abcdiagram(abcd), 3
print.complecs(complecs), 16
print.determinantSelectionTable
  (determinant_selection_table), 26
print.determinantStructure
  (determinantStructure), 25
print.dMCD(dMCD), 36
print.nnc(nnc), 38
repeatStr, 44
repStr(repeatStr), 44
reset(opts), 41
rmdpartials::partial(), 30
room_for_improvement
  (determinant_selection_table), 26
set (opts), 41
strwrap(), 46
subdeterminantProducts, 35
subdeterminantProducts
  (determinantStructure), 25
subdeterminants, 35
subdeterminants (determinantStructure), 25
ufs::disattenuate.d(), 40
vecTxt, 45
vecTxtQ(vecTxt), 45
wrapVector, 46