Package 'MONECA'

June 14, 2025

Title Mobility Network Clustering Analysis

Description MONECA creates a weighted network from a mobility table and uses cliques (or similar) to create discrete and nested clusters.

URL None yet

Version 0.1.8

Maintainer Anton Grau Larsen <agl.ioa@cbs.dk>

Author Jonas Touboel <jt@soc.ku.dk>

Depends R (>= 3.5.0)

Imports ggplot2 (>= 3.4.0), ggraph (>= 2.0.0), grid, igraph (>= 1.3.0), toOrdinal, RColorBrewer (>= 1.1-2), scales (>= 1.2.0), stats, grDevices, dplyr (>= 1.0.0), tidygraph (>= 1.2.0)

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

License GPL-3

RoxygenNote 7.3.2

VignetteBuilder knitr

Encoding UTF-8

NeedsCompilation no

Contents

MONECA-package 2
.moneca_env 4
analyze_package_dependencies 4
ego.plot 4
find.segments 6
first.level.summary 7
force.segments 7
generate_dependency_report 8
generate_example_datasets 8
generate_mobility_data 10
gg.moneca 12
layout.matrix 15
modern_plotting 15
moneca 16
moneca.plot 17

occupations 19
plot_ego_ggraph 19
plot_moneca_ggraph 21
plot_stair_ggraph 24
print.descriptive.moneca 25
print.first_level_summary 26
print.moneca 26
segment.colors 28
segment.edges 28
segment.membership 29
segment.quality 30
stair.plot 31
test_igraph_compatibility 33
validate_description_dependencies 34
vertex.mobility 34
weight.matrix 34

Index **36**

`MONECA-package` *MONECA: Mobility Network Clustering Analysis*

Description

MONECA creates weighted networks from mobility tables and uses cliques to identify discrete and nested clusters of social positions with high internal mobility. The package provides comprehensive tools for analyzing social mobility patterns through graph-theoretic approaches with modern visualization capabilities.

Details

The MONECA package implements a sophisticated hierarchical clustering algorithm that detects mobility patterns in social networks. It identifies segments with high internal mobility by finding cliques in weighted network representations of mobility data, creating nested hierarchical structures that reveal the underlying organization of social mobility.

Key Features:

- **Network-based clustering** using relative risk matrices
- **Hierarchical segmentation** with multiple nested levels
- **Modern visualization** with ggraph and ggplot2
- **Synthetic data generation** for testing and demonstrations
- **Quality metrics** for assessing segmentation performance
- **Multiple visualization types** (network, ego, stair plots)
- **Flexible input formats** with automatic data validation

Core Functions:

- `moneca`: Main clustering algorithm
- `find.segments`: Core segmentation identification
- `weight.matrix`: Relative risk matrix calculation

- `segment.membership`: Extract cluster memberships

Modern Visualization (Recommended):

- `plot_moneca_ggraph`: Network visualization with ggraph
- `plot_ego_ggraph`: Ego network analysis
- `plot_stair_ggraph`: Multi-level segmentation plots

Legacy Visualization (Backward Compatibility):

- `gg.moneca`: Original ggplot2-based visualization
- `moneca.plot`: Base R network plotting
- `ego.plot`: Legacy ego network plots
- `stair.plot`: Legacy stair plots

Data Generation:

- `generate_mobility_data`: Create synthetic mobility tables
- `generate_example_datasets`: Predefined example datasets

Package Dependencies

Required packages:

- `igraph` (>= 1.3.0): Network analysis with compatibility layer
- `ggplot2` (>= 3.4.0): Graphics and plotting
- `RColorBrewer`: Color palettes for visualizations
- `scales`: Scale transformations and formatting

Modern visualization packages:

- `ggraph` (>= 2.0.0): Advanced network visualization
- `tidygraph` (>= 1.2.0): Tidy graph manipulation
- `dplyr` (>= 1.0.0): Data manipulation

Additional packages:

- `grid`: Low-level graphics
- `toOrdinal`: Ordinal number formatting

See Also

Useful links:

- `Noneyet`

`.moneca_env` *igraph Compatibility Layer*

Description

This file provides compatibility functions to handle both old and new igraph API ensuring MONECA works with igraph versions from 1.3.0 onwards

Usage

```
.moneca_env
```

Format

An object of class `environment` of length 7.

`analyze_package_dependencies` *Analyze package dependencies usage*

Description

Analyze package dependencies usage

Usage

```
analyze_package_dependencies()
```

Value

Data frame with dependency analysis

`ego.plot` *Legacy Ego Network Visualization*

Description

Creates ego network plots showing mobility patterns from a single focal position using the legacy ggplot2 plotting system. For modern ego network analysis, use `plot_ego_ggraph`.

Usage

```
ego.plot(
  segments,
  mxa.b,
  id = 1,
  lay = layout.matrix(segments),
  edge.size = 0.8,
  border.padding = 1,
  title.line = TRUE,
  vertex.size = "totals",
  small.cell.reduction = 5,
  edge.weight = "discrete",
  color.scheme = "RdPu",
  ...
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`mxa.b`	The original mobility matrix used in the MONECA analysis.
`id`	Integer or character specifying the focal node (ego) for the analysis.
`lay`	Layout matrix for node positioning, typically from `layout.matrix`.
`edge.size`	Numeric value for edge thickness. Default is 0.8.
`border.padding`	Numeric value for segment boundary padding. Default is 1.
`title.line`	Logical indicating whether to add a title line. Default is TRUE.
`vertex.size`	Specification for vertex sizes. If "totals", sizes are derived from row/column totals in the mobility matrix. Otherwise, uses the specified values.
`small.cell.reduction`	Numeric threshold below which vertices receive different shapes to indicate low mobility volumes. Default is 5.
`edge.weight`	Character string specifying edge weight display. "discrete" creates categorical edge weights, otherwise uses continuous weights.
`color.scheme`	Character string specifying the RColorBrewer color scheme for the visualization. Default is "RdPu".
`...`	Additional arguments passed to `gg.moneca`.

Details

This function creates a focused view of mobility patterns from a single position in the social structure. It highlights both incoming and outgoing mobility flows and uses different visual elements to represent:

- Edge colors/weights for relative risk levels
- Node sizes for total mobility volumes
- Node shapes for positions with low mobility
- Node colors for mobility share proportions

Note: This function is maintained for backward compatibility. For new analyses, consider using `plot_ego_ggraph` which offers better performance and more modern styling options.

Value

A ggplot2 object showing the ego network.

See Also

`plot_ego_ggraph` for modern ego network visualization, `gg.moneca` for the underlying plotting function

Examples

```
## Not run:
# Requires legacy data and eliter package
data(occupations)
ego.plot(mob.seg, mob.mat, id = 2)

# Customized ego plot
ego.plot(mob.seg, mob.mat,
         id = 3,
         edge.size = 1.2,
         color.scheme = "Blues",
         small.cell.reduction = 10)

## End(Not run)
```

`find.segments` *Find Segments in Mobility Networks*

Description

Identifies discrete groups or segments based on a weighted network matrix using a clique-based algorithm. This function implements the core segmentation algorithm that iteratively assigns nodes to segments based on network ties.

Usage

```
find.segments(
  mat,
  cliques,
  cut.off = 1,
  mode = "symmetric",
  delete.upper.tri = TRUE
)
```

Arguments

`mat`	A weighted adjacency matrix representing mobility flows or relationships. Should include row and column names representing the categories/classes.
`cliques`	A list of cliques (complete subgraphs) in the network, typically obtained from `igraph::cliques()`.
`cut.off`	Numeric threshold for minimum weight or relative risk to be considered a network tie. Default is 1.

`mode`	Character string specifying how to handle asymmetric relationships: • "symmetric" (default): Standard symmetric treatment • "Mutual": Only mutual ties (bidirectional) are considered • "Unmutual": Unidirectional ties are allowed
`delete.upper.tri`	Logical indicating whether to process only the lower triangle of the matrix for efficiency. Default is TRUE.

Details

The algorithm works by iteratively examining the strongest remaining ties in the network and assigning nodes to segments based on clique membership. It uses a greedy approach that prioritizes stronger connections and larger existing segments.

Value

A list with two components:

membership A factor indicating segment membership for each node

cliques A list where each element contains the indices of nodes belonging to that segment

See Also

`moneca` for the main analysis function

`first.level.summary` *First level summary*

Description

First level summary

Usage

```
first.level.summary(
  segments,
  small.cell.reduction = segments$small.cell.reduction
)
```

`force.segments` *force.segments*

Description

Create a two-level segment-object with a forced solutionz

Usage

```
force.segments(segments, variable)
```

`generate_dependency_report`

Generate comprehensive dependency report

Description

Generate comprehensive dependency report

Usage

```
generate_dependency_report(output_file = NULL)
```

Arguments

`output_file`	Optional file path to save the report

Value

Invisible NULL

`generate_example_datasets`

Generate Predefined Example Datasets

Description

Creates a collection of predefined mobility datasets with different structural characteristics. These examples are useful for tutorials, testing different MONECA parameters, and demonstrating various mobility regime types.

Usage

```
generate_example_datasets()
```

Details

This function provides ready-to-use examples that demonstrate different types of social mobility structures:

- **Simple**: Good for learning MONECA basics
- **Complex**: Tests algorithm performance on larger structures
- **Rigid**: Shows segmentation in low-mobility societies
- **Fluid**: Tests segmentation in high-mobility contexts
- **Polarized**: Demonstrates class divide patterns
- **Gradual**: Shows continuous mobility gradients

These examples are particularly useful for:

- Tutorial and teaching materials
- Comparing MONECA parameters across mobility types
- Testing visualization functions
- Benchmarking algorithm performance

Value

A named list containing mobility matrices with different characteristics:

simple Small 4-class example with clear segmentation

complex Larger 8-class example with hierarchical structure

rigid High immobility example showing strong class boundaries

fluid Low immobility example with extensive mobility

polarized Example with strong upper/lower class divide

gradual Example with smooth class transitions

Each matrix is in standard MONECA format with row/column totals.

See Also

`generate_mobility_data` for custom synthetic data, `moneca` for analysis, `plot_moneca_ggraph` for visualization

Examples

```
# Load all example datasets
examples <- generate_example_datasets()
names(examples)

# Examine the simple example
print(examples$simple)

# Compare different mobility regimes
rigid_seg <- moneca(examples$rigid, segment.levels = 3)
fluid_seg <- moneca(examples$fluid, segment.levels = 3)

# Visualize different structures
## Not run:
plot_moneca_ggraph(rigid_seg, title = "Rigid Class Structure")
plot_moneca_ggraph(fluid_seg, title = "Fluid Mobility Regime")

# Create comparative stair plots
stair_rigid <- plot_stair_ggraph(rigid_seg)
stair_fluid <- plot_stair_ggraph(fluid_seg)

## End(Not run)

# Analyze segmentation quality across examples
for (name in names(examples)) {
  seg <- moneca(examples[[name]], segment.levels = 2)
  cat("Dataset:", name, "- Segments:", length(seg$segment.list[[2]]), "\n")
}
```

`generate_mobility_data` *Generate Realistic Synthetic Mobility Data*

Description

Creates synthetic social mobility tables with configurable patterns that mimic real-world mobility structures. This function is essential for testing MONECA algorithms, creating examples, and understanding how different mobility patterns affect segmentation results.

Usage

```
generate_mobility_data(
  n_classes = 10,
  n_total = 10000,
  immobility_strength = 0.6,
  class_clustering = 0.3,
  noise_level = 0.1,
  class_names = NULL,
  seed = NULL
)
```

Arguments

`n_classes`	Integer specifying the number of social classes or occupational categories. Must be at least 2. Default is 10.
`n_total`	Integer specifying the total sample size (number of individuals). Must be at least as large as n_classes. Default is 10000.
`immobility_strength`	Numeric value (0-1) controlling the strength of diagonal immobility (tendency to remain in origin class). Higher values create more rigid class structures. Default is 0.6.
`class_clustering`	Numeric value (0-1) controlling the tendency for mobility between adjacent classes in the social hierarchy. Higher values create more gradual, short-distance mobility. Default is 0.3.
`noise_level`	Numeric value (0-1) controlling the amount of random mobility between non-adjacent classes. Higher values create more chaotic mobility patterns. Default is 0.1.
`class_names`	Character vector of names for the social classes. If NULL, classes are named "Class 1", "Class 2", etc. Length must equal n_classes.
`seed`	Integer for random seed to ensure reproducible results. Default is NULL (no seed set).

Details

This function generates mobility data using a hierarchical model that reflects common patterns in social mobility research:

1. **Diagonal Dominance**: Most people remain in their origin class

2. **Distance Effects**: Mobility is more likely between adjacent classes
3. **Marginal Heterogeneity**: Class sizes vary realistically
4. **Random Component**: Some unpredictable mobility occurs

The generated data can be used to test how different mobility regimes affect MONECA segmentation and to create realistic examples for demonstrations.

Parameter combinations:

- High immobility + low clustering = rigid class boundaries
- Low immobility + high clustering = fluid but structured mobility
- High noise = chaotic mobility patterns
- Balanced parameters = realistic social mobility

Value

A square matrix in MONECA format with:

Core matrix Upper-left (n_classes x n_classes) contains mobility flows

Row margins Last column contains origin class totals

Column margins Last row contains destination class totals

Grand total Bottom-right cell contains total sample size

The matrix includes row and column names for easy interpretation.

See Also

`moneca` for analyzing the generated data, `generate_example_datasets` for predefined examples, `plot_moneca_ggraph` for visualizing results

Examples

```
# Basic synthetic data with default parameters
basic_data <- generate_mobility_data()
print(basic_data[1:6, 1:6])  # Show first 5 classes plus totals

# Small example for quick testing
test_data <- generate_mobility_data(
  n_classes = 5,
  n_total = 1000,
  immobility_strength = 0.7,
  class_clustering = 0.2,
  seed = 42
)

# Professional class structure with custom names
professional_data <- generate_mobility_data(
  n_classes = 6,
  n_total = 5000,
  class_names = c("Upper", "Upper-Middle", "Middle",
                  "Lower-Middle", "Working", "Lower"),
  immobility_strength = 0.5,
  class_clustering = 0.4,
  seed = 123
)

# Highly fluid mobility regime
fluid_data <- generate_mobility_data(
  n_classes = 8,
  immobility_strength = 0.3,
  class_clustering = 0.1,
  noise_level = 0.3
)

# Use in MONECA analysis
seg <- moneca(test_data, segment.levels = 3)
plot_moneca_ggraph(seg, title = "Synthetic Mobility Analysis")
```

gg.moneca *Legacy ggplot2 Visualization for MONECA Objects*

Description

Creates network visualizations of MONECA clustering results using ggplot2. This function provides extensive customization options but has been superseded by `plot_moneca_ggraph` for most use cases.

Usage

```
gg.moneca(
  segments,
  level = seq(segments$segment.list),
  layout = layout.matrix(segments),
  edges = log(segment.edges(segments) + 1),
  mode = "directed",
  vertex.size = "total",
  vertex.fill = "segment",
  vertex.alpha = 1,
  vertex.color = "black",
  vertex.shape = 21,
  show.edges = TRUE,
  edge.size = 1,
  edge.alpha = "weight",
  edge.color = "weight",
  edge.line = "solid",
  show.text = TRUE,
  text.size = 3,
  text.color = "black",
  text.alpha = 1,
  text.vjust = 1.5,
  show.borders = TRUE,
  border.size = 1,
  border.fill = NA,
  border.color = "black",
  border.alpha = 1,
  border.padding = 0.7,
  border.text = TRUE,
  border.labels = "segments",
  border.text.size = 4,
  border.text.color = "black",
  border.text.vjust = -0.2,
  border.text.hjust = 1,
  midpoints = TRUE,
 midpoint.arrow = arrow(angle = 20, length = unit(0.33, "cm"), ends = "last", type =
    "closed"),
  edge.text = FALSE,
  edge.text.size = 3,
  edge.text.alpha = 0.9,
  legend = "side"
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`level`	Integer vector specifying which hierarchical levels to display.
`layout`	Matrix of node coordinates or layout function result.
`edges`	Edge matrix or transformed edge weights for visualization.
`mode`	Character string specifying graph mode ("directed" or "undirected").
`vertex.size`	Aesthetic for vertex size. Can be "total", "mobility", or numeric.
`vertex.fill`	Aesthetic for vertex fill color. Can be "segment" or color specification.
`vertex.alpha`	Numeric value (0-1) for vertex transparency.
`vertex.color`	Color for vertex borders.
`vertex.shape`	Numeric code for vertex shape (see ggplot2 shapes).
`show.edges`	Logical indicating whether to display edges.
`edge.size`	Size specification for edges.
`edge.alpha`	Transparency for edges. Can be "weight" or numeric.
`edge.color`	Color specification for edges. Can be "weight" or color name.
`edge.line`	Line type for edges ("solid", "dashed", etc.).
`show.text`	Logical indicating whether to show vertex labels.
`text.size`	Numeric size for vertex labels.
`text.color`	Color for vertex labels.
`text.alpha`	Transparency for vertex labels.
`text.vjust`	Vertical adjustment for labels.
`show.borders`	Logical indicating whether to show segment boundaries.
`border.size`	Size for segment borders.
`border.fill`	Fill color for segment boundaries.
`border.color`	Color for segment border lines.
`border.alpha`	Transparency for segment borders.
`border.padding`	Padding around segment boundaries.
`border.text`	Logical indicating whether to show segment labels.

`border.labels`	Character vector of custom segment labels.
`border.text.size`	Size for segment labels.
`border.text.color`	Color for segment labels.
`border.text.vjust`	Vertical adjustment for segment labels.
`border.text.hjust`	Horizontal adjustment for segment labels.
`midpoints`	Logical indicating whether to show edge midpoints.
`midpoint.arrow`	Logical indicating whether to show arrows at midpoints.
`edge.text`	Logical indicating whether to show edge labels.
`edge.text.size`	Size for edge labels.
`edge.text.alpha`	Transparency for edge labels.
`legend`	Position for legend ("side", "bottom", "none", etc.).

Details

This function provides a highly customizable but complex interface for creating MONECA visualizations. It requires the eliter package for some functionality. For most users, `plot_moneca_ggraph` offers a more modern and user-friendly interface with better defaults.

Note: This function is maintained for backward compatibility but is no longer actively developed. New features are added to the ggraph-based plotting functions instead.

Value

A ggplot2 object.

See Also

`plot_moneca_ggraph` for modern ggraph-based plotting, `moneca.plot` for base graphics plotting

Examples

```
## Not run:
# Requires eliter package
data(occupations)
gg.moneca(mob.seg)

# Custom styling
gg.moneca(mob.seg,
         vertex.fill = "red",
         edge.color = "blue",
         show.borders = FALSE)

## End(Not run)
```

`layout.matrix` *Layout matrix*

Description

A matrix with the coordinates of the segments

Usage

```
layout.matrix(
  segments,
  attraction = c(320, 40, 10, 4, 2),
  level = seq(segments$segment.list),
  mode = "directed",
  weight.adjustment = 1,
  start.temp = 20,
  niter = 10000,
  tie.adjustment = 0.4,
  ...
)
```

Arguments

`segments`	a segment object
`attraction`	the distance between the segment points for each level.
`level`	the included levels
`mode`	the mode
`area.size`	the size of the plot area - see layout.fruchterman.reingold

`modern_plotting` *Modern MONECA Plotting with ggraph*

Description

Enhanced plotting functions using ggplot2 and ggraph for MONECA objects. These functions replace the old plotting system with modern, customizable network visualizations.

moneca *MONECA - Mobility Network Clustering Analysis*

Description

Main function for performing hierarchical clustering analysis on mobility tables. MONECA creates weighted networks from mobility data and uses cliques to identify discrete and nested clusters of positions with high internal mobility.

Usage

```
moneca(
  mx = mx,
  segment.levels = 3,
  cut.off = 1,
  mode = "symmetric",
  delete.upper.tri = TRUE,
  small.cell.reduction = 0
)
```

Arguments

`mx`	A mobility table (square matrix) with row and column totals in the last row/column. Row names should identify the categories/classes.
`segment.levels`	Integer specifying the number of hierarchical segmentation levels to compute. Default is 3. The algorithm may return fewer levels if no further meaningful segmentation is possible.
`cut.off`	Numeric threshold for the minimum relative risk to be considered a significant tie. Default is 1 (no mobility above random expectation required).
`mode`	Character string specifying edge mode ("symmetric", "Mutual", or "Unmutual"). Currently not fully implemented - uses symmetric mode.
`delete.upper.tri`	Logical indicating whether to use only lower triangle for efficiency. Default is TRUE.
`small.cell.reduction`	Numeric value to handle small cell counts. Cells with counts below this threshold are set to 0. Default is 0 (no reduction).

Details

MONECA implements an iterative algorithm that:

1. Converts the mobility table to a relative risk matrix
2. Identifies network cliques based on the threshold
3. Groups nodes into segments using the clique structure
4. Aggregates the mobility table by segments
5. Repeats the process for the specified number of levels

The algorithm stops early if no further segmentation is possible (e.g., all nodes collapse into a single segment).

Value

An object of class "moneca" containing:

segment.list A list of segment memberships for each hierarchical level. Each element is a list of vectors containing the original row indices.

mat.list A list of aggregated mobility matrices for each level, where rows/columns represent segments instead of original categories.

small.cell.reduction The small cell reduction parameter used.

References

Toubøl, J., & Larsen, A. G. (2017). Mapping the Social Class Structure: From Occupational Mobility to Social Class Categories Using Network Analysis. Sociology, 51(6), 1257-1276.

See Also

`find.segments` for the core segmentation algorithm, `weight.matrix` for relative risk calculation, `plot_moneca_ggraph` for modern visualization, `segment.membership` for extracting memberships

Examples

```
# Generate synthetic mobility data
mobility_data <- generate_mobility_data(n_classes = 6, seed = 42)

# Run MONECA analysis
seg <- moneca(mobility_data, segment.levels = 3)
print(seg)

# Examine segment membership
membership <- segment.membership(seg)
print(membership)

# Visualize with modern plotting
## Not run:
plot_moneca_ggraph(seg, node_color = "segment", title = "MONECA Clustering")

## End(Not run)
```

`moneca.plot` *Legacy Network Plot for MONECA Results*

Description

Creates a network visualization of MONECA segmentation results using base graphics and igraph. For modern visualizations, use `plot_moneca_ggraph`.

Usage

```
moneca.plot(
  segments,
  layout = layout.matrix(segments),
  edges = NULL,
  mode = "directed",
  level = seq(segments$segment.list),
  vertex.size = 5,
  vertex.frame.color = "black",
  edge.curved = FALSE,
  vertex.color = "grey50",
  vertex.label.color = "black",
  vertex.label.cex = 0.5,
  vertex.label.dist = 0.12,
  edge.arrow.size = 0.1,
  mark.col = NULL,
  mark.expand = 10,
  border.col = "black",
  edge.width = 1,
  edge.color = "black"
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`layout`	A matrix of node coordinates, typically from `layout.matrix`.
`edges`	An adjacency matrix of edges, typically from `segment.edges`.
`mode`	Character string specifying graph mode ("directed" or "undirected").
`level`	Integer vector of hierarchical levels to visualize.
`vertex.size`	Numeric value for vertex size. Default is 5.
`vertex.frame.color`	Color for vertex borders. Default is "black".
`edge.curved`	Logical for curved edges. Default is FALSE.
`vertex.color`	Color for vertices. Default is "grey50".
`vertex.label.color`	Color for vertex labels. Default is "black".
`vertex.label.cex`	Size multiplier for labels. Default is 0.5.
`vertex.label.dist`	Distance of labels from vertices. Default is 0.12.
`edge.arrow.size`	Size of edge arrows. Default is 0.1.
`mark.col`	Color for segment markers. Default is NULL.
`mark.expand`	Expansion factor for segment boundaries. Default is 10.
`border.col`	Color for segment borders. Default is "black".
`edge.width`	Width of edges. Default is 1.
`edge.color`	Color for edges. Can be a color name or matrix. Default is "black".

Details

This function provides backward compatibility with earlier versions of MONECA. For new analyses, consider using `plot_moneca_ggraph` which offers more modern styling and customization options.

Value

NULL (creates a plot as side effect).

See Also

`plot_moneca_ggraph` for modern plotting

`occupations` *Occupational mobility*

Description

Occupational mobility

Examples

```
data(occupations)
```

`plot_ego_ggraph` *Ego Network Visualization with ggraph*

Description

Creates focused visualizations of mobility patterns from a single focal node (ego). This function shows all incoming and outgoing mobility flows for a specific category, making it ideal for understanding individual position dynamics.

Usage

```
plot_ego_ggraph(
  segments,
  mobility_matrix,
  ego_id,
  min_weight = 0,
  layout = "stress",
  highlight_color = "red",
  flow_color = "viridis",
  node_size_range = c(2, 8),
  edge_width_range = c(0.2, 3),
  title = NULL,
  ...
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`mobility_matrix`	The original mobility matrix used in the MONECA analysis. Should include row and column totals.
`ego_id`	Integer or character specifying the focal node. Can be: • Integer: Row/column index in the mobility matrix • Character: Row/column name from the mobility matrix
`min_weight`	Numeric threshold for minimum edge weight to include nodes. Only nodes connected to the ego with edge weights >= min_weight will be shown. Default is 0 (show all connected nodes). Use higher values to focus on stronger mobility flows.
`layout`	Character string specifying the layout algorithm. Default is "stress" which often works well for ego networks. Other options include "fr", "kk", "dh".
`highlight_color`	Color for the ego (focal) node. Default is "red".
`flow_color`	Character string specifying the color scheme for mobility flows. Default is "viridis". Can be any viridis variant ("viridis", "plasma", "inferno", etc.).
`node_size_range`	Numeric vector of length 2 specifying the range for node sizes. Default is c(3, 12).
`edge_width_range`	Numeric vector of length 2 specifying the range for edge widths. Default is c(0.5, 3).
`title`	Character string for plot title. Default is NULL.
`...`	Additional arguments passed to the ggraph layout function.

Details

Ego networks are particularly useful for understanding the mobility patterns of specific social positions. The visualization highlights:

- The focal position (ego) in a distinct color
- Incoming mobility flows (edges pointing to ego)
- Outgoing mobility flows (edges from ego)
- The relative strength of different flows through edge width and color

Only non-zero mobility flows are displayed to reduce visual clutter. Edge colors and widths are scaled to represent the volume of mobility flows.

Value

A ggplot2 object showing the ego network visualization.

See Also

`plot_moneca_ggraph` for full network visualization, `plot_stair_ggraph` for multi-level visualization, `moneca` for the main analysis function

Examples

```
# Generate synthetic data and run MONECA
mobility_data <- generate_mobility_data(n_classes = 6, seed = 123)
seg <- moneca(mobility_data, segment.levels = 3)

# Ego network for the middle category (index 3)
plot_ego_ggraph(seg, mobility_data, ego_id = 3,
                title = "Mobility from Middle Class")

# Ego network using category name (if available)
if (!is.null(rownames(mobility_data))) {
  plot_ego_ggraph(seg, mobility_data, ego_id = rownames(mobility_data)[1])
}

# Focus on strong mobility flows only (weight >= 10)
plot_ego_ggraph(seg, mobility_data,
                ego_id = 2,
                min_weight = 10,
                title = "Strong Mobility Flows")

# Customized ego plot
plot_ego_ggraph(seg, mobility_data,
                ego_id = 2,
                layout = "fr",
                highlight_color = "orange",
                flow_color = "plasma",
                edge_width_range = c(1, 5),
                title = "Professional Class Mobility")
```

`plot_moneca_ggraph` *Modern Network Visualization for MONECA Results*

Description

Creates sophisticated network visualizations of MONECA clustering results using ggraph and ggplot2. This function provides modern, highly customizable plots with support for multiple layout algorithms, node aesthetics, and segment highlighting.

Usage

```
plot_moneca_ggraph(
  segments,
  level = seq(segments$segment.list)[-1],
  layout = "fr",
  edges = "auto",
  node_size = "total",
  node_color = "segment",
  node_alpha = 0.8,
  edge_width = "weight",
  edge_color = "grey50",
  edge_alpha = 0.6,
  show_labels = TRUE,
  label_size = 3,
  show_segments = TRUE,
  segment_alpha = 0.3,
  color_palette = "Set3",
  theme_style = "void",
  title = NULL,
  ...
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`level`	Integer vector specifying which hierarchical levels to visualize. Default displays all levels except the first (which represents individual categories).
`layout`	Character string or matrix specifying the layout algorithm: • "fr" (default): Fruchterman-Reingold force-directed layout • "kk": Kamada-Kawai layout • "dh": Davidson-Harel layout • "mds": Multidimensional scaling • "stress": Stress majorization • Matrix: Custom coordinate matrix (n_nodes x 2)
`edges`	Edge matrix or "auto" to automatically generate using `segment.edges`. Default is "auto".
`node_size`	Aesthetic mapping for node size: • "total": Size by total mobility volume (default) • "mobility": Size by off-diagonal mobility rate • Numeric vector: Custom sizes for each node • Single numeric: Fixed size for all nodes
`node_color`	Aesthetic mapping for node color: • "segment" (default): Color by segment membership • "mobility": Color by mobility rate • Character vector: Custom colors for each node • Single color: Fixed color for all nodes
`node_alpha`	Numeric value (0-1) for node transparency. Default is 0.8.
`edge_width`	Aesthetic for edge width: • "weight" (default): Width proportional to edge weight • Numeric: Fixed width for all edges
`edge_color`	Color for edges. Default is "grey50".
`edge_alpha`	Numeric value (0-1) for edge transparency. Default is 0.6.
`show_labels`	Logical indicating whether to display node labels. Default is TRUE.
`label_size`	Numeric size for node labels. Default is 3.
`show_segments`	Logical indicating whether to highlight segment boundaries. Default is TRUE.
`segment_alpha`	Numeric value (0-1) for segment boundary transparency. Default is 0.3.
`color_palette`	Character string specifying the color palette for segments. Can be any RColorBrewer palette name. Default is "Set3".

`theme_style`	Character string specifying the plot theme: • "void" (default): Clean background with no axes • "minimal": Minimal theme with subtle gridlines • "classic": Traditional ggplot2 theme
`title`	Character string for plot title. Default is NULL (no title).
`...`	Additional arguments passed to ggraph layout functions.

Details

This function creates publication-quality network visualizations with extensive customization options. It automatically handles node positioning, edge rendering, and segment highlighting. The resulting plot can be further modified using standard ggplot2 syntax.

For interactive exploration, different layout algorithms may work better with different network structures. Force-directed layouts ("fr") work well for most cases, while "stress" layouts often produce cleaner results for dense networks.

Value

A ggplot2 object that can be further customized or displayed.

See Also

`moneca` for the main analysis function, `plot_ego_ggraph` for ego network visualization, `plot_stair_ggraph` for multi-level visualization, `segment.edges` for edge matrix generation

Examples

```
# Generate synthetic data and run MONECA
mobility_data <- generate_mobility_data(n_classes = 6, seed = 123)
seg <- moneca(mobility_data, segment.levels = 3)

# Basic network plot
plot_moneca_ggraph(seg)

# Customized plot with different aesthetics
plot_moneca_ggraph(seg,
  layout = "stress",
  node_color = "mobility",
  edge_width = "weight",
  color_palette = "Spectral",
  title = "Social Mobility Network",
  show_segments = FALSE
)

# Plot with custom node sizes and colors
custom_plot <- plot_moneca_ggraph(seg,
  node_size = c(8, 6, 10, 4, 7, 5),
  node_color = "red",
  edge_color = "darkblue",
  theme_style = "minimal"
)

# Further customize with ggplot2
custom_plot +
  ggplot2::labs(subtitle = "Custom subtitle") +
  ggplot2::theme(plot.title = ggplot2::element_text(size = 16))
```

`plot_stair_ggraph` *Multi-Level Segmentation Visualization (Stair Plot)*

Description

Creates a series of network plots showing how segmentation evolves across hierarchical levels in a MONECA analysis. This "stair plot" provides insight into the progressive clustering of social positions.

Usage

```
plot_stair_ggraph(
  segments,
  levels = seq_along(segments$segment.list)[-1],
  layout = NULL,
  ncol = 2,
  ...
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`levels`	Integer vector specifying which hierarchical levels to visualize. Default includes all levels except the first (individual categories).
`layout`	Layout specification for consistency across plots. Can be: • NULL (default): Use layout.matrix() for consistent positioning • Character string: Layout algorithm name ("fr", "kk", "stress", etc.) • Matrix: Custom coordinate matrix for node positions
`ncol`	Integer specifying the number of columns in the plot grid. Default is 2. Set to 1 for vertical arrangement.
`...`	Additional arguments passed to `plot_moneca_ggraph`.
`combine_plots`	Logical indicating whether to combine plots into a single grid (TRUE) or return a list of individual plots (FALSE). Default is TRUE.

Details

The stair plot helps visualize the hierarchical nature of MONECA segmentation by showing how larger segments at higher levels break down into smaller, more specific segments at lower levels. This is particularly useful for:

- Understanding the segmentation process
- Identifying optimal levels of analysis
- Presenting results to different audiences
- Comparing segmentation stability across levels

When using a consistent layout across all plots, the relative positions of nodes remain the same, making it easier to track how segments evolve.

Value

If combine_plots = TRUE, returns a combined plot grid object. If combine_plots = FALSE, returns a list of ggplot objects, one for each level.

See Also

plot_moneca_ggraph for single-level visualization, plot_ego_ggraph for ego network analysis, layout.matrix for consistent layouts, moneca for the main analysis function

Examples

```
# Generate synthetic data and run MONECA
mobility_data <- generate_mobility_data(n_classes = 6, seed = 123)
seg <- moneca(mobility_data, segment.levels = 4)

# Basic stair plot
stair_plots <- plot_stair_ggraph(seg)

# Customized stair plot with specific levels
custom_stair <- plot_stair_ggraph(seg,
                                  levels = c(2, 3),
                                  layout = "stress",
                                  ncol = 1,
                                  node_color = "mobility")

# Return individual plots for further customization
plot_list <- plot_stair_ggraph(seg, combine_plots = FALSE)
# Modify individual plots
plot_list[[1]] <- plot_list[[1]] + ggplot2::labs(subtitle = "Level 2")

## Not run:
# Display the combined plot
print(stair_plots)

# Save individual plots
for (i in seq_along(plot_list)) {
  ggplot2::ggsave(paste0("level_", i, ".png"), plot_list[[i]])
}

## End(Not run)
```

---

print.descriptive.moneca *Print Method for Descriptive MONECA Statistics*

Description

Internal function that formats and displays the detailed statistics calculated by the print.moneca method. This function creates formatted tables showing network properties and mobility statistics across hierarchical levels.

Usage

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

Arguments

`x`	A descriptive.moneca object containing calculated statistics.
`...`	Additional arguments (currently unused).

Details

This internal function is responsible for the formatted display of MONECA analysis results. It creates several summary tables:

- Degree distributions (all, in, out) for each level
- Edge weight distributions for each level
- Diagonal mobility percentages
- Network density and connectivity statistics

Value

NULL (called for side effects - printing formatted output).

`print.first_level_summary` *Print first level summary*

Description

Print first level summary

Usage

```
## S3 method for class 'first_level_summary'
print(out)
```

`print.moneca` *Print Method for MONECA Objects*

Description

Provides a comprehensive summary of MONECA analysis results including mobility statistics, network properties, and segmentation quality measures across all hierarchical levels.

Usage

```
## S3 method for class 'moneca'
print(segments, small.cell.reduction = segments$small.cell.reduction, ...)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`small.cell.reduction`	Numeric threshold for small cell handling. If NULL, uses the value from the MONECA object.
`...`	Additional arguments (currently unused).

Details

The print method calculates and displays several key statistics:

Mobility Statistics:

- Total mobility rate (proportion of off-diagonal movement)
- Diagonal mobility (immobility) by level
- Mobility captured by significant edges

Network Properties:

- Node degrees (in, out, total) by level
- Network density by level
- Number of isolated nodes
- Edge weight distributions

Segmentation Quality:

- Number of segments per level
- Proportion of mobility within vs. between segments
- Network coherence measures

Value

Invisibly returns a list of descriptive statistics, but primarily called for its side effect of printing a formatted summary.

See Also

`moneca`, `summary.moneca`

Examples

```
# Generate data and run analysis
mobility_data <- generate_mobility_data(n_classes = 6, seed = 42)
seg <- moneca(mobility_data, segment.levels = 3)

# Print comprehensive summary
print(seg)

# The summary includes mobility rates, network statistics, and
# segmentation quality measures for each hierarchical level
```

`segment.colors` *Generate Colors for Segments*

Description

Creates a grayscale color scheme for MONECA segments based on internal mobility rates. Darker colors indicate higher immobility (lower internal mobility).

Usage

```
segment.colors(segments)
```

Arguments

`segments`	A MONECA object returned by `moneca`.

Details

This function calculates grayscale colors where the intensity reflects the immobility rate within each segment. Segments with higher immobility (more stable positions) receive darker colors.

Value

A list of color vectors, one for each hierarchical level.

`segment.edges` *Extract Segment Edge Matrix*

Description

Creates an adjacency matrix representing edges between segments based on mobility flows. This function is used for network visualization and analysis.

Usage

```
segment.edges(
  segments,
  cut.off = 1,
  mode = "directed",
  level = seq(segments$segment.list),
  segment.reduction = seq(segments$segment.list),
  method = "all",
  top = 3,
  diagonal = NULL,
  small.cell.reduction = 0
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`cut.off`	Numeric threshold for minimum relative risk to include an edge. Default is 1.
`mode`	Character string specifying the graph mode ("directed" or "undirected"). Default is "directed".
`level`	Integer vector specifying which hierarchical levels to include. Default includes all levels.
`segment.reduction`	Integer vector specifying levels for which to remove internal segment edges. Default includes all levels.
`method`	Character string specifying edge filtering method: • "all" (default): Include all edges above threshold • "top.out": Keep only the top outgoing edges per node • "top.in": Keep only the top incoming edges per node
`top`	Integer specifying how many top edges to keep when using "top.out" or "top.in" methods. Default is 3.
`diagonal`	Controls diagonal values. If NULL (default), diagonal is zeroed.
`small.cell.reduction`	Numeric threshold for small cell handling.

Value

A square matrix representing edge weights between nodes/segments.

See Also

`plot_moneca_ggraph`, `moneca.plot`

`segment.membership` *Extract Segment Membership Information*

Description

Returns a data frame showing which segment each original category belongs to across the specified hierarchical levels of a MONECA analysis.

Usage

```
segment.membership(segments, level = seq(segments$segment.list))
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`level`	Integer vector specifying which hierarchical levels to include. Default includes all available levels.

Details

The membership strings indicate both the hierarchical level and the specific segment within that level. For example, "2.3" means the category belongs to segment 3 at level 2 of the hierarchy.

Value

A data frame with two columns:

name Character vector of original category names

membership Character vector indicating segment membership, formatted as "level.segment" (e.g., "2.1" for level 2, segment 1)

See Also

`moneca`, `plot_moneca_ggraph`

Examples

```
# Generate data and run analysis
mob_data <- generate_mobility_data(n_classes = 5, seed = 42)
seg <- moneca(mob_data, segment.levels = 3)

# Get membership information
membership <- segment.membership(seg)
print(membership)

# Get membership for specific levels only
membership_level2 <- segment.membership(seg, level = 2)
```

`segment.quality` *Segment quality*

Description

Segment quality

Usage

```
segment.quality(segments, final.solution = FALSE)
```

stair.plot *Legacy Multi-Level Stair Plot*

Description

Creates a series of plots showing how segmentation evolves across hierarchical levels using the legacy ggplot2 system. For modern stair plots, use plot_stair_ggraph.

Usage

```
stair.plot(
  segments,
  level = seq(segments$segment.list),
  layout = layout.matrix(segments),
 edges = segment.edges(segments, cut.off = 1, method = "all", segment.reduction = 0,
    level = 1),
  mode = "directed",
  vertex.size = "total",
  vertex.alpha = 1,
  vertex.color = "black",
  vertex.shape = 21,
  show.edges = TRUE,
  edge.size = 0.5,
  edge.alpha = "weight",
  edge.color = "black",
  edge.line = "solid",
  show.text = FALSE,
  text.size = 3,
  text.color = "black",
  text.alpha = 1,
  text.vjust = 1.5,
  show.borders = TRUE,
  border.size = 1,
  border.fill = NA,
  border.color = "black",
  border.alpha = 1,
  border.padding = 1,
  border.text = TRUE,
  border.labels = "segments",
  border.text.size = 4,
  border.text.color = "black",
  border.text.vjust = -0.2,
  border.text.hjust = 1,
  midpoints = TRUE,
 midpoint.arrow = arrow(angle = 20, length = unit(0.33, "cm"), ends = "last", type =
    "closed"),
  edge.text = FALSE,
  edge.text.size = 3,
  edge.text.alpha = 0.9,
  legend = "side",
  level.title = "Level"
)
```

Arguments

`segments`	A MONECA object returned by `moneca`.
`level`	Integer vector specifying which levels to include in the stair plot.
`layout`	Layout matrix for consistent node positioning across plots.
`edges`	Edge matrix or specification for network edges.
`mode`	Character string specifying graph mode ("directed" or "undirected").
`vertex.size`	Specification for vertex sizes ("total" or numeric).
`vertex.alpha`	Numeric transparency for vertices (0-1).
`vertex.color`	Color specification for vertex borders.
`vertex.shape`	Numeric shape code for vertices.
`show.edges`	Logical indicating whether to display edges.
`edge.size`	Numeric size for edges.
`edge.alpha`	Transparency for edges ("weight" or numeric).
`edge.color`	Color specification for edges.
`edge.line`	Line type for edges ("solid", "dashed", etc.).
`show.text`	Logical indicating whether to show vertex labels.
`text.size`	Numeric size for text labels.
`text.color`	Color for text labels.
`text.alpha`	Transparency for text labels.
`text.vjust`	Vertical adjustment for text labels.
`show.borders`	Logical indicating whether to show segment boundaries.
`border.size`	Size for segment borders.
`border.fill`	Fill color for segment boundaries.
`border.color`	Color for segment border lines.
`border.alpha`	Transparency for segment borders.
`border.padding`	Padding around segment boundaries.
`border.text`	Logical indicating whether to show segment labels.
`border.labels`	Specification for segment labels.
`border.text.size`	Size for segment labels.
`border.text.color`	Color for segment labels.
`border.text.vjust`	Vertical adjustment for segment labels.
`border.text.hjust`	Horizontal adjustment for segment labels.
`midpoints`	Logical indicating whether to show edge midpoints.
`midpoint.arrow`	Logical indicating whether to show arrows at midpoints.
`edge.text`	Logical indicating whether to show edge labels.
`edge.text.size`	Size for edge labels.
`edge.text.alpha`	Transparency for edge labels.
`legend`	Position specification for legend.
`level.title`	Specification for level titles.

Details

This function creates multiple plots showing the progression of segmentation across hierarchical levels. Each plot uses the same layout to maintain consistency, making it easy to see how segments merge or split across levels.

Note: This function is maintained for backward compatibility and requires the eliter package. For new analyses, use `plot_stair_ggraph` which offers better performance and modern styling.

Value

A list of ggplot2 objects, one for each segmentation level.

See Also

`plot_stair_ggraph` for modern stair plots, `gg.moneca` for the underlying plotting function

Examples

```
## Not run:
# Requires legacy data and eliter package
data(occupations)
plots <- stair.plot(mob.seg)
plots[[2]]  # Display second level

# Customized stair plot
custom_stairs <- stair.plot(mob.seg,
                            level = c(2, 3, 4),
                            vertex.size = "total",
                            show.borders = TRUE,
                            border.text.size = 5)

## End(Not run)
```

`test_igraph_compatibility` *Test framework for dependency updates*

Description

This script provides comprehensive testing for the igraph migration and dependency updates to ensure compatibility and functionality

Usage

```
test_igraph_compatibility()
```

Value

List with test results

`validate_description_dependencies` *Validate DESCRIPTION file dependencies*

Description

Validate DESCRIPTION file dependencies

Usage

```
validate_description_dependencies()
```

Value

List with validation results

`vertex.mobility` *Vertex mobility*

Description

Vertex mobility

Usage

```
vertex.mobility(segments)
```

`weight.matrix` *Calculate Relative Risk Weight Matrix*

Description

Converts a mobility table into a relative risk matrix by comparing observed mobility flows to expected flows under independence. This matrix forms the basis for network construction in MONECA analysis.

Usage

```
weight.matrix(
  mx,
  cut.off = 1,
  symmetric = TRUE,
  diagonal = NULL,
  small.cell.reduction = 0
)
```

Arguments

`mx`	A mobility table (square matrix) with row and column totals in the last row/column.
`cut.off`	Numeric threshold for minimum relative risk. Values below this are set to NA. Default is 1.
`symmetric`	Logical indicating whether to force the matrix to be symmetric by adding it to its transpose. Default is TRUE.
`diagonal`	Controls diagonal values. If NULL (default), diagonal is set to NA. Otherwise, diagonal values are preserved.
`small.cell.reduction`	Numeric value for handling small cells. Cells with counts below this threshold are set to 0 before calculating relative risks. Default is 0.

Details

The relative risk for cell (i,j) is calculated as:

$$RR_{ij} = O_{ij}/E_{ij}$$

where O_{ij} is the observed count and E_{ij} is the expected count under independence: $E_{ij} = (n_i * n_j)/N$

Value

A matrix of relative risks where:

- Values > 1 indicate mobility above expected levels
- Values < 1 indicate mobility below expected levels
- Values below cut.off are set to NA

See Also

`moneca` for the main analysis function

Examples

```
# Create a simple mobility table
mob_table <- matrix(c(100, 20, 10, 130,
                      15, 80, 25, 120,
                      5,  10, 50,  65,
                      120, 110, 85, 315),
                    nrow = 4, byrow = TRUE)
rownames(mob_table) <- colnames(mob_table) <- c("A", "B", "C", "Total")

# Calculate relative risk matrix
rr_matrix <- weight.matrix(mob_table, cut.off = 1.5)
```

Index

∗ **Test**
    test_igraph_compatibility, 33
∗ **compatibility**
    test_igraph_compatibility, 33
∗ **igraph**
    test_igraph_compatibility, 33
∗ **internal**
    .moneca_env, 4
    print.descriptive.moneca, 25
    test_igraph_compatibility, 33
∗ **layer**
    test_igraph_compatibility, 33
.moneca_env, 4

analyze_package_dependencies, 4

ego.plot, *3*, 4

find.segments, *2*, 6, *17*
first.level.summary, 7
force.segments, 7

generate_dependency_report, 8
generate_example_datasets, *3*, 8, *11*
generate_mobility_data, *3*, *9*, 10
gg.moneca, *3*, *5*, *6*, 12, *33*

layout.fruchterman.reingold, *15*
layout.matrix, *5*, 15, *18*, *25*

modern_plotting, 15
MONECA *(*MONECA-package*)*, 2
moneca, *2*, *5*, *7*, *9*, *11*, *13*, 16, *18*, *20*, *22–25*, *27–30*, *32*, *35*
MONECA-package, 2
moneca.plot, *3*, *14*, 17, *29*

occupations, 19

plot_ego_ggraph, *3–6*, 19, *23*, *25*
plot_moneca_ggraph, *3*, *9*, *11*, *12*, *14*, *17*, *19*, *20*, 21, *24*, *25*, *29*, *30*
plot_stair_ggraph, *3*, *20*, *23*, 24, *31*, *33*
print.descriptive.moneca, 25
print.first_level_summary, 26
print.moneca, 26

segment.colors, 28
segment.edges, *18*, *22*, *23*, 28
segment.membership, *3*, *17*, 29
segment.quality, 30
stair.plot, *3*, 31
summary.moneca, *27*

test_igraph_compatibility, 33

validate_description_dependencies, 34
vertex.mobility, 34

weight.matrix, *2*, *17*, 34