Package: gglyph (via r-universe)

October 24, 2026

Title Network-Style Visualization of Directed Pairwise Relationships

Version 0.2.0

Description Create network-style visualizations of pairwise relationships using custom edge glyphs built on top of 'ggplot2'. The package supports both statistical and non-statistical data and allows users to represent directed relationships. This enables clear, publication-ready graphics for exploring and communicating relational structures in a wide range of domains. The method was first used in Abu-Akel et al. (2021) <doi:10.1371/journal.pone.0245100>. Code is released under the MIT License; included datasets are licensed under the Creative Commons Attribution 4.0 International (CC BY 4.0).

License MIT + file LICENSE

URL https://valentinsvelev.github.io/gglyph/

BugReports https://github.com/valentinsvelev/gglyph/issues/

Depends R (>= 4.1.0)

Imports dplyr, ggplot2, ggtext, grid, magrittr, rlang, stats, tibble, tidyr, utils

Suggests ggthemes, haven, kableExtra, knitr, patchwork, psych, purrr, readr, rmarkdown, rstatix, spelling, svglite, testthat (>= 3.0.0), tidyverse, viridis, viridisLite

VignetteBuilder knitr

Config/testthat/edition 3

Encoding UTF-8

Language en-US

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Config/pak/sysreqs libicu-dev libjpeg-dev libpng-dev libxml2-dev libssl-dev

Repository https://valentinsvelev.r-universe.dev

Date/Publication 2025-10-03 08:02:37 UTC

RemoteUrl https://github.com/valentinsvelev/gglyph

RemoteRef HEAD

RemoteSha 2d3adae8ede04b9b22e55fc53274045fbd7f402f

Contents

generate_mock_data 2
geom_glyph 3
pisa_2022 5
process_data_general 6
process_data_statistical 7
sipri_milex_1995_2023 8

Index **10**

`generate_mock_data` *Generate mock data for gglyph::geom_glyph()*

Description

Generates custom mock data to be passed to gglyph::geom_glyph().

Usage

```
generate_mock_data(
  n_nodes = 5,
  n_edges = 7,
  n_groups = 1,
  statistical = FALSE,
  p_threshold = 0.05
)
```

Arguments

`n_nodes`	Number of nodes in the graph. Default is 5.
`n_edges`	Number of edges to generate. Default is 7.
`n_groups`	Number of groups (for faceting). Default is 1 (ungrouped).
`statistical`	If TRUE, generates mock p-values for edges. Default is FALSE.
`p_threshold`	The significance threshold for filtering edges. Default is 0.05.

Value

A data frame with mock data for nodes and edges.

Examples

```
# For non-grouped data
mock_data <- generate_mock_data(
  n_nodes = 5,
  n_edges = 7,
  n_groups = 1,
  statistical = FALSE,
  p_threshold = 0.05
)

# For grouped data
mock_data <- generate_mock_data(
  n_nodes = 5,
  n_edges = 7,
  n_groups = 3,
  statistical = TRUE,
  p_threshold = 0.05
)
```

---

geom_glyph *Create a directed network-style graph*

Description

Create a network-style graph that illustrates directed pairwise relationships using custom edges.

Usage

```
geom_glyph(
  mapping = NULL,
  data = NULL,
  edge_size = 1,
  edge_colour = "grey",
  edge_fill = NULL,
  edge_alpha = 1,
  node_size = 1,
  node_colour = "black",
  node_fill = NULL,
  node_alpha = 1,
  node_shape = 21,
  node_spacing = 1,
  label_size = 12,
  group_label_size = 13,
  legend_title = NULL,
  legend_subtitle = NULL,
  ...,
  stat = "identity",
  position = "identity",
```

```
  na.rm = FALSE,
  show.legend = TRUE,
  inherit.aes = TRUE
)
```

Arguments

mapping	Set of aesthetic mappings created by aes(). You must supply mapping if there is no plot mapping.
data	A DataFrame with preprocessed data from either gglyph::preprocess_data_general() or gglyph::preprocess_data_statistical(). To be passed to ggplot2::ggplot().
edge_size	A numeric scaling factor indicating the size/width of the edges. Default is 1.
edge_colour	Color(s) of the edge outlines. Can be a single string (for non-grouped data) or a vector of strings or a function (for grouped data). Default is "grey".
edge_fill	Color(s) for the edge fill. Can be a single string, a vector of strings, or a color function. If NULL, defaults to edge_colour.
edge_alpha	A numeric value indicating the transparency of the edges. Default is 1.
node_size	A numeric value indicating the size of the nodes. Default is 8.
node_colour	Color(s) of the node outlines. Can be a single string (for non-grouped data) or a vector of strings or a function (for grouped data). Default is "black".
node_fill	Color for the node fill. If NULL, defaults to node_colour.
node_alpha	A numeric value indicating the transparency of the nodes. Default is 1.
node_shape	A numeric value specifying the shape of the nodes, following ggplot2's shape specifications. Default is 21 (a circle with a border).
node_spacing	A numeric scaling factor for the distance between nodes. Values > 1 will push nodes further apart, while values < 1 will bring them closer. Default is 1.
label_size	A numeric value indicating the size of the node labels. Default is 12.
group_label_size	A numeric value indicating the size of group label. Default is 13.
legend_title	Title for the legend as a string.
legend_subtitle	Subtitle for the legend as a string.
...	Additional arguments passed to ggplot2 layer.
stat	The statistical transformation to use on the data for this layer.
position	A position adjustment to use on the data for this layer.
na.rm	If FALSE, the default, missing values are removed with a warning. If TRUE, missing values are silently removed.
show.legend	Should this layer be included in the legends? Default is TRUE.
inherit.aes	If FALSE, overrides the default aesthetics, rather than combining with them. Default is FALSE.

Value

A ggplot2 layer with custom network-based graph.

See Also

ggplot2::ggsave()

Examples

```
# For non-grouped/-facetted plot
data <- gglyph::generate_mock_data(n_groups = 1)

ggplot2::ggplot(data = data) +
  gglyph::geom_glyph()

ggplot2::ggplot(data = data) +
  gglyph::geom_glyph(edge_colour = "purple", node_colour = "blue")

ggplot2::ggplot(data = data) +
  gglyph::geom_glyph(edge_colour = "purple", node_colour = "blue") +
  ggplot2::labs(title = "A beautiful glyph")

# For grouped/facetted plot
data <- gglyph::generate_mock_data(n_groups = 3)

ggplot2::ggplot(data = data) +
  gglyph::geom_glyph() +
  ggplot2::facet_wrap(~ group)

ggplot2::ggplot(data = data) +
  gglyph::geom_glyph(edge_colour = viridis::viridis, node_colour = viridis::viridis) +
  ggplot2::facet_wrap(~ group)

ggplot2::ggplot(data = data) +
  gglyph::geom_glyph(edge_colour = viridis::viridis, node_colour = viridis::viridis) +
  ggplot2::facet_wrap(~ group) +
  ggplot2::labs(title = "Beautiful glyphs")
```

---

pisa_2022 *PISA 2022 data*

Description

Results of pairwise t-tests (with Bonferroni correction) performed on a subset from the PISA 2022 data.

Usage

```
data(pisa_2022)
```

Format

A data frame with 492 rows and 3 variables:

`from` Category A of educational level (ISCED) attained by the parents of the respondent (character).

`to` Category B of educational level (ISCED) attained by the parents of the respondent (character).

`group` Country of the respondent (character).

`sig` p-value of the pairwise t-test (numeric).

Source

Data obtained from:

- OECD (2023). *PISA 2022 Database* [Data Set]. Zenodo. doi:10.5281/zenodo.13382904

Licensed under CC BY 4.0: https://creativecommons.org/licenses/by/4.0/

References

Additional reference(s) for further reading:

- OECD (2024). *PISA 2022 Technical Report*. OECD Publishing, Paris. doi:10.1787/01820d6d-en

Examples

```
data(pisa_2022)
head(pisa_2022)
```

---

process_data_general *Process general/non-statistical data*

Description

Prepare general/non-statistical data for plotting with gglyph::geom_glyph().

Usage

```
process_data_general(data, from = "from", to = "to", group = NULL)
```

Arguments

`data`	A DataFrame or tibble containing the input data to be processed.
`from`	A string indicating the column name for the start nodes.
`to`	A string indicating the column name for the end nodes.
`group`	A string indicating the column name for the grouping variable.

Value

A DataFrame with the preprocessed data that is to be passed to gglyph::geom_glyph().

Examples

```
data(sipri_milex_1995_2023)

# For non-grouped data
processed_data <- process_data_general(
  data = sipri_milex_1995_2023,
  from = "from",
  to = "to"
)

# For grouped data
processed_data <- process_data_general(
  data = sipri_milex_1995_2023,
  from = "from",
  to = "to",
  group = "group"
)
```

---

process_data_statistical

Process statistical data

Description

Prepare statistical data for plotting with gglyph::geom_glyph().

Usage

```
process_data_statistical(
  data,
  from = "from",
  to = "to",
  group = NULL,
  sig = "sig",
  thresh = 0.05
)
```

Arguments

data	A DataFrame or tibble containing the input data to be processed.
from	A string indicating the column name for the start nodes.
to	A string indicating the column name for the end nodes.
group	A string indicating the column name for the grouping variable.
sig	A string indicating the column name for the significance level.
thresh	A single number indicating the significance threshold. Default is 0.05.

Value

A DataFrame with the preprocessed data that is to be passed to gglyph::geom_glyph().

Examples

```
data(pisa_2022)

# For non-grouped data
processed_data <- process_data_statistical(
  data = pisa_2022,
  from = "from",
  to = "to",
  sig = "sig",
  thresh = 0.05
)

# For grouped data
processed_data <- process_data_statistical(
  data = pisa_2022,
  from = "from",
  to = "to",
  sig = "sig",
  group = "group",
  thresh = 0.05
)
```

---

sipri_milex_1995_2023 *SIPRI Military Expenditure 1995-2023 data*

Description

A subset of the SIPRI Military Expenditure 1949-2023 data.

Usage

```
data(sipri_milex_1995_2023)
```

Format

A data frame with 77 rows and 3 variables:

`from` Name of country A (character).

`to` Name of country B (character).

`group` Year (numeric).

Source

Data obtained from:

- SIPRI (2025). *SIPRI Military Expenditure Database* [Data Set]. doi:10.55163/CQGC9685

Licensed under CC BY 4.0: https://creativecommons.org/licenses/by/4.0/

Examples

```
data(sipri_milex_1995_2023)
head(sipri_milex_1995_2023)
```

Index

∗ **dataset**
    pisa_2022, 5
    sipri_milex_1995_2023, 8

generate_mock_data, 2
geom_glyph, 3
ggplot2::ggsave(), *5*

pisa_2022, 5
process_data_general, 6
process_data_statistical, 7

sipri_milex_1995_2023, 8