Package ‘ggseqplot’

November 15, 2025

Title Render Sequence Plots using 'ggplot2'

Version 0.8.9

Description A set of wrapper functions that mainly re-produces most of the sequence plots rendered with TraMineR::seqplot(). Whereas 'TraMineR' uses base R to produce the plots this library draws on 'ggplot2'.
The plots are produced on the basis of a sequence object defined with TraMineR::seqdef(). The package automates the reshaping and plotting of sequence data. Resulting plots are of class 'ggplot', i.e. components can be added and tweaked using '+' and regular 'ggplot2' functions.

URL `https://maraab23.github.io/ggseqplot/`, `https://github.com/maraab23/ggseqplot`

BugReports `https://github.com/maraab23/ggseqplot/issues`

Depends R (>= 4.1.0)

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.3

RdMacros Rdpack

Imports cli, colorspace, dplyr (>= 1.1.0), forcats (>= 1.0.0), ggh4x, ggplot2 (>= 3.4.0), ggrepel, ggtext, glue, haven, patchwork, purrr, Rdpack, rlang, tidyr, TraMineR (>= 2.2-5)

Suggests covr, ggthemes, knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Marcel Raab [aut, cre] (ORCID: <`https://orcid.org/0000-0002-3097-1591`>)

Maintainer Marcel Raab <`marcel.raab@gmail.com`>

Repository CRAN

Date/Publication 2025-11-15 12:50:02 UTC

Contents

ggseqdplot 2
ggseqeplot 5
ggseqfplot 7
ggseqiplot 9
ggseqmsplot 12
ggseqmtplot 14
ggseqrfplot 16
ggseqrplot 20
ggseqtrplot 22

Index **25**

`ggseqdplot` *Sequence Distribution Plot*

Description

Function for rendering state distribution plots with `ggplot2` (Wickham 2016) instead of base R's `plot` function that is used by `TraMineR::seqplot` (Gabadinho et al. 2011).

Usage

```
ggseqdplot(
  seqdata,
  no.n = FALSE,
  group = NULL,
  dissect = NULL,
  weighted = TRUE,
  with.missing = FALSE,
  border = FALSE,
  with.entropy = FALSE,
  linetype = "dashed",
  linecolor = "black",
  linewidth = 1,
  facet_ncol = NULL,
  facet_nrow = NULL,
  ...
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`no.n`	specifies if number of (weighted) sequences is shown (default is `TRUE`)
`group`	A vector of the same length as the sequence data indicating group membership. When not NULL, a distinct plot is generated for each level of group.

`dissect`	if `"row"` or `"col"` are specified separate distribution plots instead of a stacked plot are displayed; `"row"` and `"col"` display the distributions in one row or one column respectively; default is `NULL`
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`with.missing`	Specifies if missing states should be considered when computing the state distributions (default is `FALSE`).
`border`	if `TRUE` bars are plotted with black outline; default is `FALSE` (also accepts `NULL`)
`with.entropy`	add line plot of cross-sectional entropies at each sequence position
`linetype`	The linetype for the entropy subplot (`with.entropy==TRUE`) can be specified with an integer (0-6) or name (0 = blank, 1 = solid, 2 = dashed, 3 = dotted, 4 = dotdash, 5 = longdash, 6 = twodash); ; default is `"dashed"`
`linecolor`	Specifies the color of the entropy line if `with.entropy==TRUE`; default is `"black"`
`linewidth`	Specifies the width of the entropy line if `with.entropy==TRUE`; default is `1`
`facet_ncol`	Number of columns in faceted (i.e. grouped) plot
`facet_nrow`	Number of rows in faceted (i.e. grouped) plot
`...`	if group is specified additional arguments of `ggplot2::facet_wrap` such as `"labeller"` or `"strip.position"` can be used to change the appearance of the plot. Does not work if `dissect` is used

Details

Sequence distribution plots visualize the distribution of all states by rendering a series of stacked bar charts at each position of the sequence. Although this type of plot has been used in the life course studies for several decades (see Blossfeld (1987) for an early application), it should be noted that the size of the different bars in stacked bar charts might be difficult to compare - particularly if the alphabet comprises many states (Wilke 2019). This issue can be addressed by breaking down the aggregated distribution specifying the `dissect` argument. Moreover, it is important to keep in mind that this plot type does not visualize individual trajectories; instead it displays aggregated distributional information (repeated cross-sections). For a more detailed discussion of this type of sequence visualization see, for example, Brzinsky-Fay (2014), Fasang and Liao (2014), and Raab and Struffolino (2022).

The function uses `TraMineR::seqstatd` to obtain state distributions (and entropy values). This requires that the input data (`seqdata`) are stored as state sequence object (class `stslist`) created with the `TraMineR::seqdef` function. The state distributions are reshaped into a a long data format to enable plotting with `ggplot2`. The stacked bars are rendered by calling `geom_bar`; if `entropy = TRUE` entropy values are plotted with `geom_line`. If the `group` or the `dissect` argument are specified the sub-plots are produced by using `facet_wrap`. If both are specified the plots are rendered with `facet_grid`.

The data and specifications used for rendering the plot can be obtained by storing the plot as an object. The appearance of the plot can be adjusted just like with every other ggplot (e.g., by changing the theme or the scale using + and the respective functions).

Value

A sequence distribution plot created by using `ggplot2`. If stored as object the resulting list object (of class gg and ggplot) also contains the data used for rendering the plot.

Author(s)

Marcel Raab

References

Blossfeld H (1987). "Labor-Market Entry and the Sexual Segregation of Careers in the Federal Republic of Germany." *American Journal of Sociology*, **93**(1), 89–118. doi:10.1086/228707.

Brzinsky-Fay C (2014). "Graphical Representation of Transitions and Sequences." In Blanchard P, Bühlmann F, Gauthier J (eds.), *Advances in Sequence Analysis: Theory, Method, Applications*, Life Course Research and Social Policies, 265–284. Springer, Cham. doi:10.1007/9783319049694_14.

Fasang AE, Liao TF (2014). "Visualizing Sequences in the Social Sciences: Relative Frequency Sequence Plots." *Sociological Methods & Research*, **43**(4), 643–676. doi:10.1177/0049124113506563.

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). "Analyzing and Visualizing State Sequences in R with TraMineR." *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Raab M, Struffolino E (2022). *Sequence Analysis*, volume 190 of *Quantitative Applications in the Social Sciences*. SAGE, Thousand Oaks, CA. `https://sa-book.github.io/`.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

Wilke C (2019). *Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures*. O'Reilly Media, Sebastopol, CA. ISBN 978-1-4920-3108-6.

Examples

```
library(TraMineR)
library(ggplot2)

# Use example data from TraMineR: actcal data set
data(actcal)

# We use only a sample of 300 cases
set.seed(1)
actcal <- actcal[sample(nrow(actcal), 300), ]
actcal.lab <- c("> 37 hours", "19-36 hours", "1-18 hours", "no work")
actcal.seq <- seqdef(actcal, 13:24, labels = actcal.lab)

# state distribution plots; grouped by sex
# with TraMineR::seqplot
seqdplot(actcal.seq, group = actcal$sex)
# with ggseqplot
ggseqdplot(actcal.seq, group = actcal$sex)
# with ggseqplot applying a few additional arguments, e.g. entropy line
ggseqdplot(actcal.seq, group = actcal$sex,
           no.n = TRUE, with.entropy = TRUE, border = TRUE)
```

```
# break down the stacked plot to ease comparisons of distributions
ggseqdplot(actcal.seq, group = actcal$sex, dissect = "row")

# make use of ggplot functions for modifying the plot
ggseqdplot(actcal.seq) +
  scale_x_discrete(labels = month.abb) +
  labs(title = "State distribution plot", x = "Month") +
  guides(fill = guide_legend(title = "Alphabet")) +
  theme_classic() +
  theme(plot.title = element_text(size = 30,
                                  margin = margin(0, 0, 20, 0)),
    plot.title.position = "plot")
```

`ggseqeplot` *Sequence Entropy Plot*

Description

Function for plotting the development of cross-sectional entropies across sequence positions with `ggplot2` (Wickham 2016) instead of base R's `plot` function that is used by `TraMineR::seqplot` (Gabadinho et al. 2011). Other than in `TraMineR::seqHtplot` group-specific entropy lines are displayed in a common plot.

Usage

```
ggseqeplot(
  seqdata,
  group = NULL,
  weighted = TRUE,
  with.missing = FALSE,
  linewidth = 1,
  linecolor = "Okabe-Ito",
  gr.linetype = FALSE
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`group`	If grouping variable is specified plot shows one line for each group
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`with.missing`	Specifies if missing states should be considered when computing the entropy index (default is `FALSE`).
`linewidth`	Specifies the with of the entropy line; default is `1`

`linecolor`	Specifies color palette for line(s); default is `"Okabe-Ito"` which contains up to 9 colors (first is black). if more than 9 lines should be rendered, user has to specify an alternative color palette
`gr.linetype`	Specifies if line type should vary by group; hence only relevant if group argument is specified; default is `FALSE`

Details

The function uses `TraMineR::seqstatd` to compute entropies. This requires that the input data (`seqdata`) are stored as state sequence object (class `stslist`) created with the `TraMineR::seqdef` function.

The entropy values are plotted with `geom_line`. The data and specifications used for rendering the plot can be obtained by storing the plot as an object. The appearance of the plot can be adjusted just like with every other ggplot (e.g., by changing the theme or the scale using + and the respective functions).

Value

A line plot of entropy values at each sequence position. If stored as object the resulting list object also contains the data (long format) used for rendering the plot.

Author(s)

Marcel Raab

References

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). "Analyzing and Visualizing State Sequences in R with TraMineR." *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

Examples

```
library(TraMineR)

# Use example data from TraMineR: actcal data set
data(actcal)

# We use only a sample of 300 cases
set.seed(1)
actcal <- actcal[sample(nrow(actcal), 300), ]
actcal.lab <- c("> 37 hours", "19-36 hours", "1-18 hours", "no work")
actcal.seq <- seqdef(actcal, 13:24, labels = actcal.lab)

# sequences sorted by age in 2000 and grouped by sex
# with TraMineR::seqplot (entropies shown in two separate plots)
seqHtplot(actcal.seq, group = actcal$sex)
# with ggseqplot (entropies shown in one plot)
```

```
ggseqeplot(actcal.seq, group = actcal$sex)
ggseqeplot(actcal.seq, group = actcal$sex, gr.linetype = TRUE)

# manual color specification
ggseqeplot(actcal.seq, linecolor = "darkgreen")
ggseqeplot(actcal.seq, group = actcal$sex,
          linecolor = c("#3D98D3FF", "#FF363CFF"))
```

`ggseqfplot` *Sequence Frequency Plot*

Description

Function for rendering sequence index plot of the most frequent sequences of a state sequence object using `ggplot2` (Wickham 2016) instead of base R's `plot` function that is used by `TraMineR::seqplot` / `TraMineR::plot.stslist.freq` (Gabadinho et al. 2011).

Usage

```
ggseqfplot(
  seqdata,
  group = NULL,
  ranks = 1:10,
  weighted = TRUE,
  border = FALSE,
  proportional = TRUE,
  ylabs = "total",
  no.coverage = FALSE,
  facet_ncol = NULL,
  facet_nrow = NULL
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`group`	A vector of the same length as the sequence data indicating group membership. When not NULL, a distinct plot is generated for each level of group.
`ranks`	specifies which of the most frequent sequences should be plotted; default is the first ten (`1:10`); if set to 0 all sequences are displayed
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`border`	if `TRUE` bars are plotted with black outline; default is `FALSE` (also accepts `NULL`)
`proportional`	if `TRUE` (default), the sequence heights are displayed proportional to their frequencies

`ylabs`	defines appearance of y-axis labels; default (`"total"`) only labels min and max (i.e. cumulative relative frequency); if `"share"` labels indicate relative frequency of each displayed sequence (note: overlapping labels are removed)
`no.coverage`	specifies if information on total coverage is shown as caption or as part of the group/facet label if `ylabs == "share"` (default is `TRUE`)
`facet_ncol`	Number of columns in faceted (i.e. grouped) plot
`facet_nrow`	Number of rows in faceted (i.e. grouped) plot

Details

The subset of displayed sequences is obtained by an internal call of `TraMineR::seqtab`. The extracted sequences are plotted by a call of `ggseqiplot` which uses `ggplot2::geom_rect` to render the sequences. The data and specifications used for rendering the plot can be obtained by storing the plot as an object. The appearance of the plot can be adjusted just like with every other ggplot (e.g., by changing the theme or the scale using `+` and the respective functions).

Experienced ggplot2 users might notice the customized labeling of the y-axes in the faceted plots (i.e. plots with specified `group` argument). This has been achieved by utilizing the very helpful `ggh4x` library.

Value

A sequence frequency plot created by using `ggplot2`. If stored as object the resulting list object (of class gg and ggplot) also contains the data used for rendering the plot.

Author(s)

Marcel Raab

References

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). "Analyzing and Visualizing State Sequences in R with TraMineR." *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

See Also

`ggseqiplot`

Examples

```
library(TraMineR)
library(ggplot2)

# Use example data from TraMineR: actcal data set
data(actcal)

# We use only a sample of 300 cases
```

```
set.seed(1)
actcal <- actcal[sample(nrow(actcal), 300), ]
actcal.lab <- c("> 37 hours", "19-36 hours", "1-18 hours", "no work")
actcal.seq <- seqdef(actcal, 13:24, labels = actcal.lab)

# sequence frequency plot
# with TraMineR::seqplot
seqfplot(actcal.seq)
# with ggseqplot
ggseqfplot(actcal.seq)
# with ggseqplot applying additional arguments and some layout changes
ggseqfplot(actcal.seq,
           group = actcal$sex,
           ranks = 1:5,
           ylabs = "share") +
  scale_x_discrete(breaks = 1:12,
                   labels = month.abb,
                   expand = expansion(add = c(0.2, 0)))
```

`ggseqiplot` *Sequence Index Plot*

Description

Function for rendering sequence index plots with `ggplot2` (Wickham 2016) instead of base R's `plot` function that is used by `TraMineR::seqplot` (Gabadinho et al. 2011).

Usage

```
ggseqiplot(
  seqdata,
  no.n = FALSE,
  group = NULL,
  sortv = NULL,
  weighted = TRUE,
  border = FALSE,
  ytlab = NULL,
  facet_scale = "free_y",
  facet_ncol = NULL,
  facet_nrow = NULL,
  ...
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`no.n`	specifies if number of (weighted) sequences is shown as part of the y-axis title or group/facet title (default is `TRUE`)

`group`	A vector of the same length as the sequence data indicating group membership. When not NULL, a distinct plot is generated for each level of group.
`sortv`	Vector of numerical values sorting the sequences or a sorting method (either `"from.start"` or `"from.end"`). See details.
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`border`	if `TRUE` bars are plotted with black outline; default is `FALSE` (also accepts `NULL`)
`ytlab`	Specifies the type of y-axis labels. Options are: • `NULL` (default): uses pretty breaks for sequential numbering • `"all"`: displays all sequences with sequential numbers (1, 2, 3, ...) • `"id"`: displays sequence IDs (rownames of `seqdata`) using pretty breaks • `"id-all"`: displays all sequences with their sequence IDs (rownames) When using `"id"` or `"id-all"`, if `seqdata` has no rownames, sequential numbers are used as identifiers.
`facet_scale`	Specifies if y-scale in faceted plot should be free (`"free_y"` is default) or `"fixed"`
`facet_ncol`	Number of columns in faceted (i.e. grouped) plot
`facet_nrow`	Number of rows in faceted (i.e. grouped) plot
`...`	if group is specified additional arguments of `ggplot2::facet_wrap` such as `"labeller"` or `"strip.position"` can be used to change the appearance of the plot

Details

Sequence index plots have been introduced by Scherer (2001) and display each sequence as horizontally stacked bar or line. For a more detailed discussion of this type of sequence visualization see, for example, Brzinsky-Fay (2014), Fasang and Liao (2014), and Raab and Struffolino (2022).

The function uses `TraMineR::seqformat` to reshape `seqdata` stored in wide format into a spell/episode format. Then the data are further reshaped into the long format, i.e. for every sequence each row in the data represents one specific sequence position. For example, if we have 5 sequences of length 10, the long file will have 50 rows. In the case of sequences of unequal length not every sequence will contribute the same number of rows to the long data.

The reshaped data are used as input for rendering the index plot using ggplot2's `geom_rect`. `ggseqiplot` uses `geom_rect` instead of `geom_tile` because this allows for a straight forward implementation of weights. If weights are specified for `seqdata` and `weighted=TRUE` the sequence height corresponds to its weight.

When using grouped plots (i.e., when `group` is specified) with `facet_scale = "fixed"`, the function internally uses `scales = "free_y"` in `ggplot2::facet_wrap` but applies `coord_cartesian` with fixed `ylim` to achieve the effect of a fixed y-scale across facets. This approach allows for consistent y-axis ranges while maintaining flexibility in the labeling.

When a `sortv` is specified, the sequences are arranged in the order of its values. With `sortv="from.start"` sequence data are sorted according to the states of the alphabet in ascending order starting with the first sequence position, drawing on succeeding positions in the case of ties. Likewise, `sortv="from.end"` sorts a reversed version of the sequence data, starting with the final sequence position turning to preceding positions in case of ties.

When `ytlab` is set to `"id"`, `"all"`, `"id-all"`, the y-axis labeling behavior changes. With `"all"`, all sequences are labeled with sequential numbers (1, 2, 3, ...) instead of using pretty breaks. With `"id"`, the rownames of the sequence object are used as y-axis labels with pretty breaks. With `"id-all"`, all sequences are labeled with their rownames. If the sequence object has no rownames, sequential numbers (1, 2, 3, ...) are used as identifiers. These features are especially useful when working with sorted sequences or when displaying specific cases with meaningful identifiers. Note that with `"all"` and `"id-all"`, overlapping labels are automatically prevented to maintain readability. When there are many sequences and insufficient space, not all labels may be displayed. In such cases, consider increasing the plot height if you insist on seeing all labels displayed.

Note that the default aspect ratio of `ggseqiplot` is different from `TraMineR::seqIplot`. This is most obvious when `border=TRUE`. You can change the ratio either by adding code to `ggseqiplot` or by specifying the ratio when saving the code with `ggsave`.

Value

A sequence index plot. If stored as object the resulting list object also contains the data (spell format) used for rendering the plot.

Author(s)

Marcel Raab

References

Brzinsky-Fay C (2014). “Graphical Representation of Transitions and Sequences.” In Blanchard P, Bühlmann F, Gauthier J (eds.), *Advances in Sequence Analysis: Theory, Method, Applications*, Life Course Research and Social Policies, 265–284. Springer, Cham. doi:10.1007/9783319049694_14.

Fasang AE, Liao TF (2014). “Visualizing Sequences in the Social Sciences: Relative Frequency Sequence Plots.” *Sociological Methods & Research*, **43**(4), 643–676. doi:10.1177/0049124113506563.

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). “Analyzing and Visualizing State Sequences in R with TraMineR.” *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Raab M, Struffolino E (2022). *Sequence Analysis*, volume 190 of *Quantitative Applications in the Social Sciences*. SAGE, Thousand Oaks, CA. `https://sa-book.github.io/`.

Scherer S (2001). “Early Career Patterns: A Comparison of Great Britain and West Germany.” *European Sociological Review*, **17**(2), 119–144. doi:10.1093/esr/17.2.119.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

Examples

```
library(TraMineR)

# Use example data from TraMineR: actcal data set
data(actcal)
```

```
# We use only a sample of 300 cases
set.seed(1)
actcal <- actcal[sample(nrow(actcal), 300), ]
actcal.lab <- c("> 37 hours", "19-36 hours", "1-18 hours", "no work")
actcal.seq <- seqdef(actcal, 13:24, labels = actcal.lab)

# ex1 using weights
data(ex1)
ex1.seq <- seqdef(ex1, 1:13, weights = ex1$weights)

# sequences sorted by age in 2000 and grouped by sex
# with TraMineR::seqplot
seqIplot(actcal.seq, group = actcal$sex, sortv = actcal$age00)
# with ggseqplot
ggseqiplot(actcal.seq, group = actcal$sex, sortv = actcal$age00)

# sequences of unequal length with missing state, and weights
seqIplot(ex1.seq)
ggseqiplot(ex1.seq)

# ... turn weights off and add border
seqIplot(ex1.seq, weighted = FALSE, border = TRUE)
ggseqiplot(ex1.seq, weighted = FALSE, border = TRUE)

# Use sequence IDs as y-axis labels, and "fixed" y scale
ggseqiplot(ex1.seq,group = c(1, 1, 1, 2, 2, 2, 2),
           weighted = FALSE, border = TRUE, ytlab = "id", facet_scale = "fixed")

# Display all sequences with sequential numbers and with ids
ggseqiplot(actcal.seq[1:20, ], sortv = "from.end", ytlab = "all")
ggseqiplot(actcal.seq[1:20, ], sortv = "from.end", ytlab = "id-all")
```

---

ggseqmsplot *Modal State Sequence Plot*

Description

Function for rendering modal state sequence plot with `ggplot2` (Wickham 2016) instead of base R's `plot` function that is used by `TraMineR::seqplot` (Gabadinho et al. 2011).

Usage

```
ggseqmsplot(
  seqdata,
  no.n = FALSE,
  barwidth = NULL,
  group = NULL,
```

```
    weighted = TRUE,
    with.missing = FALSE,
    border = FALSE,
    facet_ncol = NULL,
    facet_nrow = NULL
  )
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`no.n`	specifies if number of (weighted) sequences is shown (default is `TRUE`)
`barwidth`	specifies width of bars (default is `NULL`); valid range: (0, 1]
`group`	A vector of the same length as the sequence data indicating group membership. When not NULL, a distinct plot is generated for each level of group.
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`with.missing`	Specifies if missing states should be considered when computing the state distributions (default is `FALSE`).
`border`	if `TRUE` bars are plotted with black outline; default is `FALSE` (also accepts `NULL`)
`facet_ncol`	Number of columns in faceted (i.e. grouped) plot
`facet_nrow`	Number of rows in faceted (i.e. grouped) plot

Details

The function uses `TraMineR::seqmodst` to obtain the modal states and their prevalence. This requires that the input data (`seqdata`) are stored as state sequence object (class `stslist`) created with the `TraMineR::seqdef` function.

The data on the modal states and their prevalences are reshaped to be plotted with `ggplot2::geom_bar`. The data and specifications used for rendering the plot can be obtained by storing the plot as an object. The appearance of the plot can be adjusted just like with every other ggplot (e.g., by changing the theme or the scale using `+` and the respective functions).

Value

A modal state sequence plot. If stored as object the resulting list object also contains the data (long format) used for rendering the plot

Author(s)

Marcel Raab

References

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). "Analyzing and Visualizing State Sequences in R with TraMineR." *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

Examples

```
library(TraMineR)

# Use example data from TraMineR: actcal data set
data(actcal)

# We use only a sample of 300 cases
set.seed(1)
actcal <- actcal[sample(nrow(actcal), 300), ]
actcal.lab <- c("> 37 hours", "19-36 hours", "1-18 hours", "no work")
actcal.seq <- seqdef(actcal, 13:24, labels = actcal.lab)

# modal state sequence plot; grouped by sex
# with TraMineR::seqplot
seqmsplot(actcal.seq, group = actcal$sex)
# with ggseqplot
ggseqmsplot(actcal.seq, group = actcal$sex)
# with ggseqplot and some layout changes
ggseqmsplot(actcal.seq, group = actcal$sex, no.n = TRUE, border = FALSE, facet_nrow = 2)
```

`ggseqmtplot` *Mean time plot*

Description

Function for rendering plot displaying the mean time spent in each state of a state sequence object using `ggplot2` (Wickham 2016) instead of base R's `plot` function that is used by `TraMineR::seqplot` (Gabadinho et al. 2011).

Usage

```
ggseqmtplot(
  seqdata,
  no.n = FALSE,
  group = NULL,
  weighted = TRUE,
  with.missing = FALSE,
  border = FALSE,
  error.bar = NULL,
```

```
  error.caption = TRUE,
  facet_scale = "fixed",
  facet_ncol = NULL,
  facet_nrow = NULL
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`no.n`	specifies if number of (weighted) sequences is shown (default is `TRUE`)
`group`	A vector of the same length as the sequence data indicating group membership. When not NULL, a distinct plot is generated for each level of group.
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`with.missing`	Specifies if missing states should be considered when computing the state distributions (default is `FALSE`).
`border`	if `TRUE` bars are plotted with black outline; default is `FALSE` (also accepts `NULL`)
`error.bar`	allows to add error bars either using the standard deviation `"SD"` or the standard error `"SE"`; default plot is without error bars
`error.caption`	a caption is added if error bars are displayed; this default behavior can be turned off by setting the argument to `"FALSE"`
`facet_scale`	Specifies if y-scale in faceted plot should be `"fixed"` (default) or `"free_y"`
`facet_ncol`	Number of columns in faceted (i.e. grouped) plot
`facet_nrow`	Number of rows in faceted (i.e. grouped) plot

Details

The information on time spent in different states is obtained by an internal call of `TraMineR::seqmeant`. This requires that the input data (`seqdata`) are stored as state sequence object (class `stslist`) created with the `TraMineR::seqdef` function. The resulting output then is prepared to be plotted with `ggplot2::geom_bar`. The data and specifications used for rendering the plot can be obtained by storing the plot as an object. The appearance of the plot can be adjusted just like with every other ggplot (e.g., by changing the theme or the scale using `+` and the respective functions).

Value

A mean time plot created by using `ggplot2`. If stored as object the resulting list object (of class gg and ggplot) also contains the data used for rendering the plot

Author(s)

Marcel Raab

References

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). "Analyzing and Visualizing State Sequences in R with TraMineR." *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

Examples

```
library(TraMineR)
library(ggplot2)

# Use example data from TraMineR: actcal data set
data(actcal)

# We use only a sample of 300 cases
set.seed(1)
actcal <- actcal[sample(nrow(actcal), 300), ]
actcal.lab <- c("> 37 hours", "19-36 hours", "1-18 hours", "no work")
actcal.seq <- seqdef(actcal, 13:24, labels = actcal.lab)

# modal state sequence plot; grouped by sex
# with TraMineR::seqplot
seqmtplot(actcal.seq, group = actcal$sex)
# with ggseqplot
ggseqmtplot(actcal.seq, group = actcal$sex)
# with ggseqplot using additional arguments and some adjustments
ggseqmtplot(actcal.seq, no.n = TRUE, error.bar = "SE") +
 coord_flip() +
 theme(axis.text.y=element_blank(),
       axis.ticks.y = element_blank(),
       panel.grid.major.y = element_blank(),
       legend.position = "top")
```

`ggseqrfplot` *Relative Frequency Sequence Plot*

Description

Function for rendering sequence index plots with `ggplot2` instead of base R's `plot` function that is used by `TraMineR::seqrfplot`. Note that ggseqrfplot uses `patchwork` to combine the different components of the plot. The function and the documentation draw heavily from `TraMineR::seqrf`.

Usage

```
ggseqrfplot(
  seqdata = NULL,
  diss = NULL,
```

```
  k = NULL,
  sortv = "mds",
  weighted = TRUE,
  grp.meth = "prop",
  squared = FALSE,
  pow = NULL,
  seqrfobject = NULL,
  border = FALSE,
  ylab = NULL,
  yaxis = TRUE,
  which.plot = "both",
  quality = TRUE,
  box.color = NULL,
  box.fill = NULL,
  box.alpha = NULL,
  outlier.jitter.height = 0,
  outlier.color = NULL,
  outlier.fill = NULL,
  outlier.shape = 19,
  outlier.size = 1.5,
  outlier.stroke = 0.5,
  outlier.alpha = NULL
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function. `seqdata` is ignored if `seqrfobject` is specified.
`diss`	pairwise dissimilarities between sequences in `seqdata` (see `TraMineR::seqdist`). `diss` is ignored if `seqrfobject` is specified.
`k`	integer specifying the number of frequency groups. When `NULL`, `k` is set as the minimum between 100 and the sum of weights over 10. `k` is ignored if `seqrfobject` is specified.
`sortv`	optional sorting vector of length `nrow(diss)` that may be used to compute the frequency groups. If `NULL`, the original data order is used. If `mds` (default), the first MDS factor of `diss` (`diss^2` when `squared=TRUE`) is used. Ties are randomly ordered. Also allows for the usage of the string inputs: `"from.start"` or `"from.end"` (see `ggseqiplot`). `sortv` is ignored if `seqrfobject` is specified.
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used.
`grp.meth`	Character string. One of `"prop"`, `"first"`, and `"random"`. Grouping method. See details. `grp.meth` is ignored if `seqrfobject` is specified.
`squared`	Logical. Should medoids (and computation of `sortv` when applicable) be based on squared dissimilarities? (default is `FALSE`). `squared` is ignored if `seqrfobject` is specified.
`pow`	Dissimilarity power exponent (typically 1 or 2) for computation of pseudo R2 and F. When `NULL`, pow is set as 1 when `squared = FALSE`, and as 2 otherwise.

	`pow` is ignored if `seqrfobject` is specified.
`seqrfobject`	object of class `seqrf` generated with `TraMineR::seqrf`. Default is `NULL`; either `seqrfobject` or `seqdata` and `diss` have to specified
`border`	if `TRUE` bars of index plot are plotted with black outline; default is `FALSE` (also accepts `NULL`)
`ylab`	character string specifying title of y-axis. If `NULL` axis title is "Frequency group"
`yaxis`	Controls if a y-axis is plotted. When set as `TRUE`, index of frequency groups is displayed.
`which.plot`	character string specifying which components of relative frequency sequence plot should be displayed. Default is `"both"`. If set to `"medoids"` only the index plot of medoids is shown. If `"diss.to.med"` only the box plots of the group-specific distances to the medoids are shown.
`quality`	specifies if representation quality is shown as figure caption; default is `TRUE`
`box.color`	specifies color of boxplot borders; default is "black
`box.fill`	specifies fill color of boxplots; default is "white"
`box.alpha`	specifies alpha value of boxplot fill color; default is 1
`outlier.jitter.height`	if greater than 0 outliers are jittered vertically. If greater than .375 height is automatically adjusted to be aligned with the box width.
`outlier.color`, `outlier.fill`, `outlier.shape`, `outlier.size`, `outlier.stroke`, `outlier.alpha`	parameters to change the appearance of the outliers. Uses defaults of `ggplot2::geom_boxplot`

Details

This function renders relative frequency sequence plots using either an internal call of `TraMineR::seqrf` or by using an object of class `"seqrf"` generated with `TraMineR::seqrf`.

For further details on the technicalities we refer to the excellent documentation of `TraMineR::seqrf`. A detailed account of relative frequency index plot can be found in the original contribution by Fasang and Liao (2014).

`ggseqrfplot` renders the medoid sequences extracted by `TraMineR::seqrf` with an internal call of `ggseqiplot`. For the box plot depicting the distances to the medoids `ggseqrfplot` uses `geom_boxplot` and `geom_jitter`. The latter is used for plotting the outliers.

Note that `ggseqrfplot` renders in the box plots analogous to the those produced by `TraMineR::seqrfplot`. Actually, the box plots produced with `TraMineR::seqrfplot` and `ggplot2::geom_boxplot` might slightly differ due to differences in the underlying computations of `grDevices::boxplot.stats` and `ggplot2::stat_boxplot`.

Note that `ggseqrfplot` uses `patchwork` to combine the different components of the plot. If you want to adjust the appearance of the composed plot, for instance by changing the plot theme, you should consult the documentation material of `patchwork`.

At this point `ggseqrfplot` does not support a grouping option. For plotting multiple groups, I recommend to produce group specific seqrfobjects or plots and to arrange them in a common plot using `patchwork`. See Example 6 in the vignette for further details: `vignette("ggseqplot", package = "ggseqplot")`

Value

A relative frequency sequence plot using `ggplot`.

Author(s)

Marcel Raab

References

Fasang AE, Liao TF (2014). “Visualizing Sequences in the Social Sciences: Relative Frequency Sequence Plots.” *Sociological Methods & Research*, **43**(4), 643–676. doi:10.1177/0049124113506563.

Examples

```
library(TraMineR)
library(ggplot2)
library(patchwork)

# From TraMineR::seqprf
# Defining a sequence object with the data in columns 10 to 25
# (family status from age 15 to 30) in the biofam data set
data(biofam)
biofam.lab <- c("Parent", "Left", "Married", "Left+Marr",
  "Child", "Left+Child", "Left+Marr+Child", "Divorced")

# Here, we use only 100 cases selected such that all elements
# of the alphabet be present.
# (More cases and a larger k would be necessary to get a meaningful example.)
biofam.seq <- seqdef(biofam[501:600, 10:25], labels=biofam.lab,
                     weights=biofam[501:600,"wp00tbgs"])
diss <- seqdist(biofam.seq, method = "LCS")


# Using 12 groups and default MDS sorting
# and original method by Fasang and Liao (2014)

# ... with TraMineR::seqrfplot (weights have to be turned off)
seqrfplot(biofam.seq, weighted = FALSE, diss = diss, k = 12,
          grp.meth="first", which.plot = "both")

# ... with ggseqrfplot
ggseqrfplot(biofam.seq, weighted = FALSE, diss = diss, k = 12, grp.meth="first")

# Arrange sequences by a user specified sorting variable:
# time spent in parental home; has ties
parentTime <- seqistatd(biofam.seq)[, 1]
b.srf <- seqrf(biofam.seq, diss=diss, k=12, sortv=parentTime)
# ... with ggseqrfplot (and some extra annotation using patchwork)
ggseqrfplot(seqrfobject = b.srf) +
  plot_annotation(title = "Sorted by time spent in parental home",
                  theme = theme(plot.title = element_text(hjust = 0.5, size = 18)))
```

`ggseqrplot` *Representative Sequence plot*

Description

Function for rendering representative sequence plots with `ggplot2` (Wickham 2016) instead of base R's `plot` function that is used by `TraMineR::seqplot` (Gabadinho et al. 2011).

Usage

```
ggseqrplot(
  seqdata,
  diss,
  group = NULL,
  criterion = "density",
  coverage = 0.25,
  nrep = NULL,
  pradius = 0.1,
  dmax = NULL,
  border = FALSE,
  proportional = TRUE,
  weighted = TRUE,
  stats = TRUE,
  colored.stats = NULL,
  facet_ncol = NULL
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`diss`	pairwise dissimilarities between sequences in `seqdata` (see `TraMineR::seqdist`)
`group`	A vector of the same length as the sequence data indicating group membership. When not NULL, a distinct plot is generated for each level of group.
`criterion`	the representativeness criterion for sorting the candidate list. One of `"freq"` (sequence frequency), `"density"` (neighborhood density), `"mscore"` (mean state frequency), `"dist"` (centrality) and `"prob"` (sequence likelihood). See details.
`coverage`	coverage threshold, i.e., minimum proportion of sequences that should have a representative in their neighborhood (neighborhood radius is defined by `pradius`).
`nrep`	number of representative sequences. If `NULL` (default), the size of the representative set is controlled by `coverage`.
`pradius`	neighborhood radius as a percentage of the maximum (theoretical) distance `dmax`. Defaults to 0.1 (10%). Sequence y is redundant to sequence x when it is in the neighborhood of x, i.e., within a distance `pradius*dmax` from x.

`dmax`	maximum theoretical distance. The `dmax` value is used to derive the neighborhood radius as `pradius*dmax`. If `NULL`, the value of `dmax` is derived from the dissimilarity matrix.
`border`	if `TRUE` bars are plotted with black outline; default is `FALSE` (also accepts `NULL`)
`proportional`	if `TRUE` (default), the sequence heights are displayed proportional to the number of represented sequences
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`stats`	if `TRUE` (default), mean discrepancy in each subset defined by all sequences attributed to one representative sequence and the mean distance to this representative sequence are displayed.
`colored.stats`	specifies if representatives in stats plot should be color coded; only recommended if number of representatives is small; if set to `NULL` (default) colors are used if n rep. <= 10; use `TRUE` or `FALSE` to change manually
`facet_ncol`	specifies the number of columns in the plot (relevant if !is.null(group))

Details

The representative sequence plot displays a set of distinct sequences as sequence index plot. The set of representative sequences is extracted from the sequence data by an internal call of `TraMineR::seqrep` according to the criteria listed in the arguments section above.

The extracted sequences are plotted by a call of `ggseqiplot` which uses `ggplot2::geom_rect` to render the sequences. If `stats = TRUE` the index plots are complemented by information on the "quality" of the representative sequences. For further details on representative sequence plots see Gabadinho et al. (2011) and the documentation of `TraMineR::plot.stslist.rep`, `TraMineR::seqplot`, and `TraMineR::seqrep`.

Note that `ggseqrplot` uses `patchwork` to combine the different components of the plot. If you want to adjust the appearance of the composed plot, for instance by changing the plot theme, you should consult the documentation material of `patchwork`.

Value

A representative sequence plot using `ggplot`.

Author(s)

Marcel Raab

References

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). “Analyzing and Visualizing State Sequences in R with TraMineR.” *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Gabadinho A, Ritschard G, Studer M, Müller NS (2011). “Extracting and Rendering Representative Sequences.” In Fred A, Dietz JLG, Liu K, Filipe J (eds.), *Knowledge Discovery, Knowlege Engineering and Knowledge Management*, volume 128, 94–106. Springer, Berlin, Heidelberg. doi:10.1007/9783642190322_7.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

Examples

```
library(TraMineR)

# Defining a sequence object with the data in columns 10 to 25
# (family status from age 15 to 30) in the biofam data set
data(biofam)
# Use sample of 300 cases
set.seed(123)
biofam <- biofam[sample(nrow(biofam),150),]
biofam.lab <- c("Parent", "Left", "Married", "Left+Marr",
"Child", "Left+Child", "Left+Marr+Child", "Divorced")
biofam.seq <- seqdef(biofam, 10:25, labels=biofam.lab)

# Computing the distance matrix
biofam.dhd <- seqdist(biofam.seq, method="DHD")

# Representative sequence  plot (using defaults)
# ... with TraMineR::seqplot
seqrplot(biofam.seq, diss = biofam.dhd)

# ... with ggseqrplot
ggseqrplot(biofam.seq, diss = biofam.dhd)
```

`ggseqtrplot` *Sequence Transition Rate Plot*

Description

Function for plotting transition rate matrix of sequence states internally computed by `TraMineR::seqtrate` (Gabadinho et al. 2011). Plot is generated using `ggplot2` (Wickham 2016).

Usage

```
ggseqtrplot(
  seqdata,
  dss = TRUE,
  group = NULL,
  no.n = FALSE,
  weighted = TRUE,
  with.missing = FALSE,
  labsize = NULL,
  axislabs = "labels",
  x_n.dodge = 1,
  facet_ncol = NULL,
```

```
  facet_nrow = NULL
)
```

Arguments

`seqdata`	State sequence object (class `stslist`) created with the `TraMineR::seqdef` function.
`dss`	specifies if transition rates are computed for STS or DSS (default) sequences
`group`	A vector of the same length as the sequence data indicating group membership. When not NULL, a distinct plot is generated for each level of group.
`no.n`	specifies if number of (weighted) sequences is shown in grouped (faceted) graph
`weighted`	Controls if weights (specified in `TraMineR::seqdef`) should be used. Default is `TRUE`, i.e. if available weights are used
`with.missing`	Specifies if missing state should be considered when computing the transition rates (default is `FALSE`).
`labsize`	Specifies the font size of the labels within the tiles (if not specified ggplot2's default is used)
`axislabs`	specifies if sequence object's long "labels" (default) or the state names from its "alphabet" attribute should be used.
`x_n.dodge`	allows to print the labels of the x-axis in multiple rows to avoid overlapping.
`facet_ncol`	Number of columns in faceted (i.e. grouped) plot
`facet_nrow`	Number of rows in faceted (i.e. grouped) plot

Details

The transition rates are obtained by an internal call of `TraMineR::seqtrate`. This requires that the input data (`seqdata`) are stored as state sequence object (class `stslist`) created with the `TraMineR::seqdef` function. As STS based transition rates tend to be dominated by high values on the diagonal, it might be worthwhile to examine DSS sequences instead (`dss = TRUE`)). In this case the resulting plot shows the transition rates between episodes of distinct states.

In any case (DSS or STS) the transitions rates are reshaped into a a long data format to enable plotting with `ggplot2`. The resulting output then is prepared to be plotted with `ggplot2::geom_tile`. The data and specifications used for rendering the plot can be obtained by storing the plot as an object. The appearance of the plot can be adjusted just like with every other ggplot (e.g., by changing the theme or the scale using + and the respective functions).

Value

A tile plot of transition rates.

Author(s)

Marcel Raab

References

Gabadinho A, Ritschard G, Müller NS, Studer M (2011). "Analyzing and Visualizing State Sequences in R with TraMineR." *Journal of Statistical Software*, **40**(4), 1–37. doi:10.18637/jss.v040.i04.

Wickham H (2016). *ggplot2: Elegant Graphics for Data Analysis*, Use R!, 2nd ed. edition. Springer, Cham. doi:10.1007/9783319242774.

Examples

```
library(TraMineR)

# Use example data from TraMineR: biofam data set
data(biofam)

# We use only a sample of 300 cases
set.seed(10)
biofam <- biofam[sample(nrow(biofam),300),]
biofam.lab <- c("Parent", "Left", "Married", "Left+Marr",
                "Child", "Left+Child", "Left+Marr+Child", "Divorced")
biofam.seq <- seqdef(biofam, 10:25, labels=biofam.lab, weights = biofam$wp00tbgs)

# Basic transition rate plot (with adjusted x-axis labels)
ggseqtrplot(biofam.seq, x_n.dodge = 2)

# Transition rate with group variable (with and without weights)
ggseqtrplot(biofam.seq, group=biofam$sex, x_n.dodge = 2)
ggseqtrplot(biofam.seq, group=biofam$sex, x_n.dodge = 2, weighted = FALSE)
```

Index

coord_cartesian, *10*

facet_grid, *3*
facet_wrap, *3*

geom_bar, *3*
geom_boxplot, *18*
geom_jitter, *18*
geom_line, *3*, *6*
geom_rect, *10*
geom_tile, *10*
ggh4x, *8*
ggplot, *19*, *21*
ggplot2, *2, 3*, *5*, *7–9*, *12*, *14–16*, *20*, *22, 23*
ggplot2::facet_wrap, *3*, *10*
ggplot2::geom_bar, *13*, *15*
ggplot2::geom_boxplot, *18*
ggplot2::geom_rect, *8*, *21*
ggplot2::geom_tile, *23*
ggplot2::stat_boxplot, *18*
ggsave, *11*
ggseqdplot, 2
ggseqeplot, 5
ggseqfplot, 7
ggseqiplot, *8*, 9, *17*, *21*
ggseqmsplot, 12
ggseqmtplot, 14
ggseqrfplot, 16
ggseqrplot, 20
ggseqtrplot, 22
grDevices::boxplot.stats, *18*

patchwork, *16*, *18*, *21*
plot, *2*, *5*, *7*, *9*, *12*, *14*, *16*, *20*

TraMineR::plot.stslist.freq, *7*
TraMineR::plot.stslist.rep, *21*
TraMineR::seqdef, *2, 3*, *5–7*, *9, 10*, *13*, *15*, *17*, *20, 21*, *23*
TraMineR::seqdist, *17*
TraMineR::seqformat, *10*
TraMineR::seqHtplot, *5*
TraMineR::seqIplot, *11*
TraMineR::seqmeant, *15*
TraMineR::seqmodst, *13*
TraMineR::seqplot, *2*, *5*, *7*, *9*, *12*, *14*, *20, 21*
TraMineR::seqrep, *21*
TraMineR::seqrf, *16*, *18*
TraMineR::seqrfplot, *16*, *18*
TraMineR::seqstatd, *3*, *6*
TraMineR::seqtab, *8*
TraMineR::seqtrate, *22, 23*