Package ‘patientProfilesVis’

November 18, 2022

Type Package
Title Visualization of Patient Profiles
Version 2.0.5
Date 2022-11-18

Description Creation of patient profile visualizations for exploration, diagnostic or monitoring purposes during a clinical trial. These static visualizations display a patient-specific overview of the evolution during the trial time frame of parameters of interest (as laboratory, ECG, vital signs), presence of adverse events, exposure to a treatment; associated with metadata patient information, as demography, concomitant medication. The visualizations can be tailored for specific domain(s) or endpoint(s) of interest. Visualizations are exported into patient profile report(s) or can be embedded in custom report(s).

Imports clinUtils, ggplot2 (>= 3.0.0), plyr, cowplot, tools, reshape2, knitr, grid, stringr, parallel, gridExtra, scales, utils

Suggests pander, shiny, testthat, grDevices, rmarkdown, gtable, pdftools, viridisLite

URL https://github.com/openanalytics/patientProfilesVis

BugReports https://github.com/openanalytics/patientProfilesVis/issues

License MIT + file LICENSE

RooxygenNote 7.2.2

VignetteBuilder knitr

SystemRequirements latex

NeedsCompilation no

Author Laure Cougnaud [aut, cre], Margaux Faes [rev] (tests), Open Analytics [cph]

Maintainer Laure Cougnaud <laure.cougnaud@openanalytics.eu>

Repository CRAN

Date/Publication 2022-11-18 16:30:02 UTC
R topics documented:

- addReferenceLinesProfilePlot ........................................ 3
- checkTimeExpand .................................................. 4
- checkTimeTrans .................................................... 5
- checkVar .......................................................... 5
- combineVerticallyGGplot ........................................... 6
- convertAesVar ..................................................... 7
- countNLines ......................................................... 7
- createSubjectProfileReport ........................................ 8
- defineIndex ........................................................ 11
- filterData .......................................................... 12
- filterMissingInVar .................................................. 14
- formatParamVarTextPlot ............................................ 15
- formatTimeInterval .................................................. 16
- formatTimeLim ....................................................... 19
- getAesScaleManual ................................................. 20
- getColorPalettePatientProfile ..................................... 20
- getMaxNLinesCombinePlot ......................................... 21
- getNLinesLabel ...................................................... 22
- getNLinesLegend ..................................................... 23
- getNLinesSubjectProfile ........................................... 23
- getOptimalColWidth ................................................ 24
- getPageVar .......................................................... 25
- getPathTemplate ..................................................... 26
- getShapePalettePatientProfile ..................................... 26
- getSplitVectorByInt ................................................ 27
- getTimeLimSubjectProfilePlots ................................... 28
- getTimeTrans ........................................................ 29
- getWidthPlot ........................................................ 30
- interactionWithMissing ............................................. 31
- isSubjectProfileTimeVariant ...................................... 31
- patientProfilesVis-common-args ................................... 32
- patientProfilesVis-palette ......................................... 33
- prepareSubjectProfile .............................................. 34
- sortSubjects ........................................................ 35
- subjectProfileCombine ............................................ 36
- subjectProfileEventPlot ............................................ 38
- subjectProfileExport ............................................... 41
- subjectProfileIntervalPlot ........................................ 42
- subjectProfileLinePlot ............................................. 47
- subjectProfileReportFormat ........................................ 50
- subjectProfileTextPlot ............................................. 51
- subjectProfileTheme ............................................... 54

Index 56
addReferenceLinesProfilePlot

Add reference lines to a profile plot

Description
Add reference lines to a profile plot

Usage
addReferenceLinesProfilePlot(
  gg,
  subjectVar = "USUBJID",
  refLines = NULL,
  refLinesData = NULL,
  refLinesTimeVar = NULL,
  refLinesLabelVar = NULL,
  refLinesColor = "black",
  refLinesLinetype = "dotted",
  timeLim = NULL,
  addLabel = FALSE
)

Arguments

<table>
<thead>
<tr>
<th>gg</th>
<th>ggplot2 with a subject profile plot for a specific subject (and page) (subset of the output of the subjectProfile[X]Plot)</th>
</tr>
</thead>
<tbody>
<tr>
<td>subjectVar</td>
<td>String, variable of data with subject ID</td>
</tr>
<tr>
<td>refLines</td>
<td>(optional) nested list with details for reference line(s). Each sublist contains:</td>
</tr>
<tr>
<td></td>
<td>• (required) 'label': string with label for the reference line</td>
</tr>
<tr>
<td></td>
<td>• (required) 'time': unique time (x) coordinate for the reference line</td>
</tr>
<tr>
<td></td>
<td>• (optional) 'color': color for the reference line, 'black' by default</td>
</tr>
<tr>
<td></td>
<td>• (optional) 'linetype': linetype for the reference line, 'dotted' by default</td>
</tr>
<tr>
<td>refLinesData</td>
<td>data.frame with data from which the reference line(s) should be extracted</td>
</tr>
<tr>
<td>refLinesTimeVar</td>
<td>string, variable of refLinesData with time for reference line(s)</td>
</tr>
<tr>
<td>refLinesLabelVar</td>
<td>string, variable of refLinesData with label for reference line(s)</td>
</tr>
<tr>
<td>refLinesColor</td>
<td>vector of length 1 with default color for reference line(s)</td>
</tr>
<tr>
<td>refLinesLinetype</td>
<td>vector of length 1 with default linetype for reference line(s)</td>
</tr>
<tr>
<td>timeLim</td>
<td>vector of length 2 with time limits. This is used to set the limits to the plot containing the reference lines labels (if requested).</td>
</tr>
<tr>
<td>addLabel</td>
<td>logical, if TRUE (FALSE by default) add the label of the reference line(s) at the bottom of the plot</td>
</tr>
</tbody>
</table>
checkTimeExpand

**Value**

If `addLabel` is:

- **TRUE**: list with:
  - `'gg'`: `ggplot2` plot with reference lines
  - `'ggRefLines'`: `ggplot2` plot containing only the labels at the specified position
- **FALSE**: `ggplot2` plot with reference lines

**Author(s)**

Laure Cougnaud

<table>
<thead>
<tr>
<th>checkTimeExpand</th>
<th>Check if some of the modules are time expanded, and extract maximum time expand for each module.</th>
</tr>
</thead>
</table>

**Description**

Check if some of the modules are time expanded, and extract maximum time expand for each module.

**Usage**

```r
checkTimeExpand(listPlots, timeLim = NULL)
```

**Arguments**

- `listPlots`: list of plots
- `timeLim`: time limits

**Value**

List of time expand for each module (named by `listPlots`)

**Author(s)**

Laure Cougnaud
**checkTimeTrans**

*Description*

Check if the subject profiles are time transformed, and if some of the plots to align (with specified timeLim) have compatible time transformation alignments.

*Usage*

```r
checkTimeTrans(listPlots, timeLim = NULL)
```

*Arguments*

- `listPlots` list of plots
- `timeLim` time limits

*Value*

List of time transformation for each module (named by `listPlots`)

*Author(s)*

Laure Cougnaud

---

**checkVar**

*Description*

Check if specified variable(s) are present in the data.

*Usage*

```r
checkVar(var, data)
```

*Arguments*

- `var` Character vector with variable(s) of interest.
- `data` Data.frame with data.

*Value*

No returned value, an error message is triggered if some variable(s) are not available in the data.
**Author(s)**
Laure Cougnaud

---

**combineVerticallyGGplot**

*Combine vertically multiple ggplot.*

**Description**
If the different modules for a subject don’t fit in the page, there are automatically split in multiple pages. The margins are extracted across plots to ensure that plots will be probably aligned.

**Usage**
```r
combineVerticallyGGplot(
  listPlots,
  maxNLines = NULL,
  nCores = 1,
  shiny = FALSE,
  verbose = FALSE,
  reportPerSubject = FALSE
)
```

**Arguments**
- `listPlots` listPlots per subject as created inside the `subjectProfileCombine` function.
- `maxNLines` Maximum number of lines for a combined plot, to fit in the page height. When the different visualizations are combined for each subject, they will be allocated to different pages if the number of lines of the combined visualization is higher than this number.
- `nCores` Integer containing the number of cores used for the computation (1 by default). If more than 1, computation is parallelized, in this case the package `parallel` is required.
- `shiny` logical, set to TRUE (FALSE by default) if the report is generated from a Shiny application. Messages during report creation will be included in the Shiny interface, and it will be mentioned at the end of the report. In this case, the shiny package should be available.
- `verbose` logical, if TRUE print messages during execution
- `reportPerSubject` Logical, if TRUE (FALSE by default) export a subject profile report by subject.

**Value**
a list (by subject) of list (by page) of `ggplot` object
**convertAesVar**

**Author(s)**
Laure Cougnaud

**Description**
This converts the empty values (") to NA. The variable is then converted as a factor. Missing values are also included in the levels of the factor, to ensure that missing values are displayed in the legend of the plot.

**Usage**
convertAesVar(data, var)

**Arguments**
- **data**
  data.frame with data
- **var**
  variable of data with aesthetic

**Value**
updated factor var variable

**Author(s)**
Laure Cougnaud

---

**countNLines**

**Description**
Count number of lines ('\n' character) per character in a vector

**Usage**
countNLines(x)

**Arguments**
- **x**
  character vector
createSubjectProfileReport

Create subject profile report.

Description

By default all subjects available in at least one module of listPlots are considered. If only a set of subjects are of interest, these are specified either:

- directly with the subject IDs of interest via subjectSubset
- by extracting subjects with a specific value (subjectSubsetValue) in a variable (subjectSubsetVar) in a specific dataset subjectSubsetData

Usage

createSubjectProfileReport(
  listPlots,
  timeLim = NULL,
  timeAlign = "all",
  timeAlignPerSubject = "none",
  refLines = NULL,
  refLinesData = NULL,
  refLinesTimeVar = NULL,
  refLinesLabelVar = NULL,
  bookmarkData = NULL,
  bookmarkVar = NULL,
  subjectSortData = bookmarkData,
  subjectSortVar = bookmarkVar,
  subjectSortDecreasing = FALSE,
  subjectVar = "USUBJID",
  subjectSubset = NULL,
  subjectSubsetData = NULL,
  subjectSubsetVar = NULL,
  subjectSubsetValue = NULL,
  subjectSample = NULL,
  seed = 123,
  subset = NULL,
  outputFile = "subjectProfile.pdf",
  exportFigures = FALSE,
createSubjectProfileReport

reportPerSubject = FALSE,
exportBatchSize = NULL,
labelVars = NULL,
maxNLines = NULL,
shiny = FALSE,
formatReport = subjectProfileReportFormat(),
verbose = FALSE,
nCores = 1
)

Arguments

listPlots nested list of plots, as returned by the subjectProfileTextPlot, subjectProfileEventPlot, subjectProfileIntervalPlot or subjectProfileLinePlot functions.

timeLim Time limits, as a numeric vector of length 2, or a list with time limits for each module, or nested list with time limits for each module and subject. If not specified, these are set to the time limits specified when creating each module (stored in attributes(x)$metaData$timeLim) otherwise to the range defined by timeAlign and timeAlignPerSubject. Note that this doesn’t modify the geoms of the plots, it only extends the axis range. So for interval module(s) if the specified timeLim is smaller than the time limits in the input plot, no arrows are created in case than the time goes above/below specified timeLim (the segment is cut).

timeAlign Character vector with time alignment across modules/subjects, either:
• ‘all’ (by default): all plots have the same time limits
• ‘none’: each of the plot (module*subject) has its own time limits
• character vector with names of the modules which should have the same time limits (should correspond to the names of listPlots)

timeAlignPerSubject Character vector, specifying if the plots should be aligned (or not) across subjects
• ‘none’ (by default): all modules to align have the same time limit across subjects
• ‘all’: all modules to align have different time limits per subject
• character vector with subset of the modules to align per subject (should correspond to the names of listPlots)

Only the modules already specified in timeAlign can be aligned by subject.

refLines (optional) nested list with details for reference line(s). Each sublist contains:
• (required) ‘label’: string with label for the reference line
• (required) ‘time’: unique time (x) coordinate for the reference line
• (optional) ‘color’: color for the reference line, ‘black’ by default
• (optional) ‘linetype’: linetype for the reference line, ‘dotted’ by default

refLinesData data.frame with data from which the reference line(s) should be extracted
refLinesTimeVar string, variable of refLinesData with time for reference line(s)
refLinesLabelVar
string, variable of refLinesData with label for reference line(s)

bookmarkData, bookmarkVar
Data.frame with data containing information for the index, and character vector with corresponding variable(s) of interest. An index will be created at the end of the subject profile report. The index contains a section per variable, referencing the pages of the report containing subject profiles for each category/variable.

subjectSortData
Data.frame with data containing information on how the subjects should be sorted (by default same as bookmarkData):
- in the report, in case one single report is created for all subjects
- for the export, in case reportPerSubject is TRUE

This data should contain subjectSortVar and subjectVar.

subjectSortVar
Character vector, variable(s) of subjectSortData indicating the order for the subjects in the report, (by default same as bookmarkVar).

subjectSortDecreasing
Logical, if TRUE (FALSE by default) subjects are sorted based on decreasing order of subjectSortVar.

subjectVar
String, variable of data with subject ID

subjectSubset
subjectSubset (optional) Character vector with subjects of interest (available in subjectVar), NULL by default.

subjectSubsetData
Data.frame used to select subset of subjects of interest.

subjectSubsetVar
String with variable of subjectSubsetData that should be considered to filter subjects. If not specified, all subjects available in subjectSubsetData are considered.

subjectSubsetValue
Character vector with value(s) of subjectSubsetVar of interest to filter subjects on.

subjectSample
(optional) Integer of length 1 with number of random subject(s) that should be considered in the specified subset dataset. By default, all specified subjects are considered (set to NULL).

seed
(optional) Integer of length 1 with seed used to select random subjects if subjectSample is specified (123 by default).

subset
Character vector with subjects of interest (among names of each list in listPlots).

outputFile
string, path to the output report

exportFigures
Logical, if TRUE (FALSE by default) the subject profile figures are also exported in pdf format in a ‘figures’ folder. Figures are named as [subjectID]-[page].pdf

reportPerSubject
Logical, if TRUE (FALSE by default) export a subject profile report by subject.
**defineIndex**

---

**Value**

The path(s) of the report(s) is returned invisibly, and the report is created at the location specified by `outputFile`.

If the report is created by subject, the name of the exported subject profile is built as: `[filename]_[subjectID].pdf`, with `[filename]` extracted from `outputFile`. Space and platform-specific file separator are replaced by a dash in the filename.

If no patient profiles are available in the input, nothing is returned and a warning is triggered.

**Author(s)**

Laure Cougnaud

---

**defineIndex**

*Define LaTeX index based on specified variable(s) of the dataset*

**Description**

Define LaTeX index based on specified variable(s) of the dataset

**Usage**

```r
defineIndex(subjects, data, var, subjectVar = "USUBJID", labelVars = NULL)
```
Arguments

- **subjects** vector with subject IDs (based on the subjectVar variable)
- **data** data.frame with data containing information on which the index should be based
- **var** variable(s) of data of interest for the index
- **subjectVar** String, variable of data with subject ID
- **labelVars** Named character vector with variable labels (names are the variable code)

Value

list with elements:

- ‘indexDef’: string with LaTeX code for creation of index, to be included directly with \texttt{cat} in a knitr document (two backslashes)
- ‘indexInfo’: character vector, named with subject ID, containing LaTeX code for index for each subject specified in subjects parameter, to be passed to the \texttt{knit} function as text (four backslashes)
- ‘indexPrint’: string with LaTeX code for printing/inclusion of index, to be included directly with \texttt{cat} in a knitr document (two backslashes)

Author(s)

Laure Cougnaud

---

**filterData** 

*Filter a dataset for records of interest, for use in the patient profiles.*

Description

Data is filtered based on the following workflow:

1. The subset dataset (of data if not specified) is filtered based on subject variable and value (if specified).
2. If a external subset dataset is specified, only the subject IDs of this filtered dataset are considered.
3. The data is filtered based on the selected subjects, from subjectSubset (if specified) or from step 2.
4. The data is filtered based on a random selection of subjects, if subjectSample is specified.

This filtering workflow is used for all subject profile visualization functions of the package.
Usage

```r
filterData(
  data,
  subsetData = NULL,
  subsetVar = NULL,
  subsetValue = NULL,
  subjectVar = "USUBJID",
  subjectSubset = NULL,
  subjectSample = NULL,
  seed = 123
)
```

Arguments

data (required) Data.frame with data

subsetData (optional) Data.frame with extra dataset to filter on. This dataset is filtered, and only records from data with common subject IDs will be retained. If not specified, data is used.

subsetVar (optional) String with variable of subset data to filter on. subsetValue should be specified too. If not specified, all records from the subset data are retained.

subsetValue (optional) Character vector with value(s) of interest to retain in the filtered data. These values should be available in subsetVar. Missing values in the subject variable are not retained in the filtered data.

subjectVar (required) String, variable of data (and subset data) with subject ID.

subjectSubset (optional) Character vector with subjects of interest (available in subjectVar), NULL by default.

subjectSample (optional) Integer of length 1 with number of random subject(s) that should be considered, e.g. to check the created patient profiles for a subset of the data. By default, all specified subjects are considered (set to NULL).

seed (optional) Integer of length 1 with seed used to select random subjects if subjectSample is specified (123 by default).

Value

possibly filtered dataset

Author(s)

Laure Cougnaud

Examples

```r
library(clinUtils)

data(dataSDTMCDISCP01)
dataAll <- dataSDTMCDISCP01
```
# keep only a subset of subjects
# (e.g. to visualize specified patient profiles
# before creating them for all subject)
filterData(
  data = dataAll$AE,
  subjectSample = 2
)

# filter based on specified variable/value:
# only adverse events possibly related
filterData(
  data = dataAll$AE,
  subsetVar = "AEREL",
  subsetValue = "POSSIBLE"
)

# filter based on a different dataset:
# keep only adverse events for subjects in a specific treatment arm
filterData(
  data = dataAll$AE,
  subsetData = dataAll$DM,
  subsetVar = "ACTARM",
  subsetValue = "Placebo"
)

# filter based on subjects of interest
filterData(
  data = dataAll$AE,
  subjectSubset = c("01-701-1148", "01-701-1211")
)

---

**filterMissingInVar**

*Filter missing records in data in the time and y variables, with informative message.*

**Description**

Filter missing records in data in the time and y variables, with informative message.

**Usage**

```
filterMissingInVar(
  data,
  var,
  varLab = getLabelVar(var, labelVars = labelVars),
  labelVars = NULL
)
```
**formatParamVarTextPlot**

**Arguments**

- `data`  
  Data.frame with data.
- `var`  
  String with variable of interest.
- `varLab`  
  String, label for var.
- `labelVars`  
  Named character vector with variable labels (names are the variable code).

**Value**

Update data with filtered records + message in the console.

**Author(s)**

Laure Cougnaud

---

**formatParamVarTextPlot**

*Format text variables for the subject profile text plotting function.*

**Description**

Text variables are wrapped across multiple lines if needed, and optionally sorted according to the levels of a grouping variable.

**Usage**

```r
formatParamVarTextPlot(
  data, 
  paramVar = NULL, 
  paramValueVar = NULL, 
  paramValueLab = NULL, 
  paramGroupVar = NULL, 
  revert = FALSE, 
  width = formatReport$yLabelWidth, 
  widthValue = ifelse(formatReport$landscape, 240, 190), 
  formatReport = subjectProfileReportFormat(), 
  table = FALSE, 
  colWidth = NULL
)
```

**Arguments**

- `data`  
  data.frame with data
- `paramVar`  
  string, variable of data with parameter
- `paramValueVar`  
  string, variable of data containing the parameter value.
- `paramValueLab`  
  Character vector with labels for paramValueVar.
formatTimeInterval

**paramGroupVar** (optional) character vector with variable(s) of data with grouping. If specified, the parameters will be grouped by this(these) variable(s) in the y-axis.

**revert** logical, if TRUE revert the order of the levels of the variable

**width** max number of characters in the codeparamVar parameter.

**widthValue** max number of characters in the codeparamValueVar parameter.

**formatReport** list with parameters used to specify the format of the report, e.g. output of the subjectProfileReportFormat function

**table** Logical, if TRUE the paramValueVar variables are displayed as table (so are not concatenated).

**colWidth** Numeric vector with approximate width of each parameter value column for a table layout.
For example in case two parameters are specified: c(0.8, 0.2)) such as the first column takes 80% of plot area, and the second column 20%.
Note: columns can be slightly bigger if their content is larger than the specified width. If not specified, column width is optimized based on the max length of the character in each column.

Value
data with reformatted paramVar and paramValueVar variables, with additional attribute: colWidth.

Author(s)
Laure Cougnaud

See Also
subjectProfileTextPlot

---

**formatTimeInterval** Set missing start/end time variable in the data.

Description
Set missing start/end time variable in the data.

Usage

```r
formatTimeInterval(
  data,
  timeStartVar,
  timeStartLab = getLabelVar(timeStartVar, labelVars = labelVars),
  timeEndVar,
  timeEndLab = getLabelVar(timeEndVar, labelVars = labelVars),
  timeStartShapeVar = NULL,
  timeEndShapeVar = NULL,
)```
subjectVar = "USUBJID",
timeLim = NULL,
timeLimData = NULL,
timeLimStartVar = NULL,
timeLimStartLab = getLabelVar(timeLimStartVar, labelVars = labelVars),
timeLimEndVar = NULL,
timeLimEndLab = getLabelVar(timeLimEndVar, labelVars = labelVars),
timeImpType = c("minimal", "data-based", "none"),
labelVars = NULL
)

Arguments

data Data.frame with data.
timeStartVar String, variable of data with start of time interval.
timeStartLab String, label for timeStartVar, displayed in a message and in the plot caption.
timeEndVar String, variable of data with end of time interval.
timeEndLab String, label for timeEndVar, displayed in a message and in the plot caption.
timeStartShapeVar (optional) String, variable of data used for the shape of the symbol displayed at the start of the time interval.
If not specified, default shape palette is used, see section 'Time interval representation'.
timeEndShapeVar String, variable of data used for the shape of the symbol displayed at the end of the time interval. If not specified, default shape palette is used, see section 'Time interval representation'.
subjectVar String, variable of data with subject ID
timeLim (optional) Vector of length 2 with time limits (x-axis). If not specified, these are extracted from the minimum timeStartVar and maximum timeEndVar per subject.
The time limits are stored as attributes of the plots, used to align the plots in the final report.
timeLimData Data.frame with data used to impute time in case some time records are missing in data, see section: 'Time interval representation'.
timeLimStartVar String, variable of timeLimData with start of the time interval.
timeLimStartLab String, label for timeLimStartVar, displayed in a message and in the plot caption.
timeLimEndVar String, variable of timeLimData with end of the time interval.
timeLimEndLab String, label for timeLimEndVar, displayed in a message and in the plot caption.
timeImpType String with imputation type: 'minimal' (default), 'data-based' or 'none', see section: 'Time interval representation'.
This imputation type is not used if a dataset used to impute time is specified.
labelVars Named character vector with variable labels (names are the variable code)
Value

list with:

- 'data': Data with:
  - imputed `timeStartVar` and `timeEndVar`
  - new column 'timeStartStatus': character vector containing status of `timeStartVar` variable: 'Complete' or 'Missing start' or NA
  - new column 'timeEndStatus': character vector containing status of `timeEndVar` variable: 'Complete' or 'Missing end' or NA
- 'timeLim': vector of length 2 with minimum/maximum time limits across subjects.
- 'timeLimSpecified': vector of length 2 with time limits as specified by the user, either extracted from `timeLim` or from `timeLimData`. If missing value within `timeLim`, the corresponding minimum/maximum value in the (updated) data is used.
- 'timeShapePalette': Named character vector with symbols for the different time status
- 'caption': String with extra explanation concerning imputation that could be included in plot caption.

Time interval representation

In case the start or the end of the time interval contain missing values:

- if a dataset (`timeLimData`), start (`timeLimStartVar`) and end (`timeLimEndVar`) variables are specified:
  1. for each subject:
     - the minimum and maximum time values across these specified time variables are extracted
     - missing start values are replaced by the minimum time
     - missing start values are replaced by the maximum time
  2. if all values are missing for this subject, they are taken across subjects
- otherwise, depending on the imputation type (`timeImpType`):
  - 'minimal' (by default):
    * if the start and the end of the interval are missing: no imputation is done, only the label is displayed
    * if the start time is missing and the end time is not missing: start time is imputed with end time, and status is set to 'Missing start'
    * if the end time is missing and the start time is not missing: end time is imputed with start time, and status is set to 'Missing end'
  - 'data-based' (default in version < 1.0.0): minimum/maximum values in the start/end time variables in the data are considered for the specific subject (if available). If there are missing for a specific subject, they are taken across subjects. If all time are missing, the range is set to 0 and Inf
  - 'none': no imputation is done

The symbols displayed at the start and end of the interval are:

- by default:
formatTimeLim

- a filled square labelled 'Complete' if the time is not missing
- a filled left-directed arrow in case of missing start time
- a filled right-directed arrow in case of missing end time

- if the variable(s) used for the shape of the start or end of the interval are specified (via `timeStartShapeVar`/`timeEndShapeVar`): labels are based on these variables, and a standard shape palette is used

The time limits are the same across subjects, and set to:

- `timeLim` if specified
- maximum time range in `timeLimStartVar` and `timeLimEndVar` in `timeLimData` if specified
- the maximum range on the data obtained after imputation of missing values

Author(s)

Laure Cougnaud

formatTimeLim

Format specified `timeLim`.

Description

In case one of the limits if missing, the corresponding minimum/maximum across subjects is used.

Usage

```r
formatTimeLim(
  data,
  subjectVar = "USUBJID",
  timeStartVar,
  timeEndVar,
  timeLim = NULL
)
```

Arguments

data Data.frame with data.
subjectVar String, variable of data with subject ID
timeStartVar String, variable of data with start of time interval.
timeEndVar String, variable of data with end of time interval.
timeLim (optional) Vector of length 2 with time limits (x-axis). If not specified, these are extracted from the minimum `timeStartVar` and maximum `timeEndVar` per subject.
The time limits are stored as attributes of the plots, used to align the plots in the final report.
Value
Numeric vector of length 2 or list of such element for each subject.

Author(s)
Laure Cougnaud

getAesScaleManual  Get custom 'scale_[type]_manual' function

Description
Get custom 'scale_[type]_manual' function

Usage
getAesScaleManual(lab, palette, type)

Arguments
lab  label for the scale (title of the legend)
palette  named vector with color palette
type  string with type of scale, e.g. 'color'

Value
output of the 'scale_[type]_manual' function

Author(s)
Laure Cougnaud

getColorPalettePatientProfile
Get a color palette for patient profile visualizations.

Description
This is a simple wrapper around getColorPalette, with different defaults:
  • inclusion of missing values by default (includeNA set to TRUE)

Usage
g getColorPalettePatientProfile(..., includeNA = TRUE)
getMaxNLinesCombinePlot

Arguments

... Arguments passed on to `clinUtils::getColorPalette`

n Integer of length 1, number of elements in palette.

x Vector with elements used for palette. If factor, the levels are used, otherwise the unique elements of the vector. Missing values are automatically removed, excepted if `includeNA` is set to TRUE.

palette A vector of custom colors, or a function returning this vector from a specific number of colors.

Default is the the colorblind `viridis` color palette.

includeNA Logical (TRUE by default), should NA elements be retained in the palette in case x is specified?

Value

Vector of shapes, named with the elements in x if x is specified.

Author(s)

Laure Cougnaud

See Also

g getColorPalette

getMaxNLinesCombinePlot

Get maximum number of lines of a 'combined plot' for a specific document

Description

Get maximum number of lines of a 'combined plot' for a specific document

Usage

```r
getMaxNLinesCombinePlot(
  heightLineIn = subjectProfileReportFormat()$heightLineIn,
  margin = subjectProfileReportFormat()$margin,
  landscape = subjectProfileReportFormat()$landscape,
  aspectRatio = subjectProfileReportFormat()$aspectRatio
)
```
Arguments

- **heightLineIn**: Numeric of length 1 with height of a line in inches, 0.2 by default.
- **margin**: Numeric of length 1, with margin in inches.
- **landscape**: Logical, if TRUE the created report is in landscape format. FALSE by default, the report is created in portrait format.
- **aspectRatio**: Numeric of length 1 (0.75 by default) with ratio between size of image in inches (derived from specified margin, landscape and heightLineIn) and real size for exported image.

Value

numeric with maximum height for plot

Author(s)

Laure Cougnaud

---

**getNLinesLabel**

*Get number of lines for specific label either from a ggplot2 object via gg or from the label via value*

Description

Get number of lines for specific label either from a ggplot2 object via gg or from the label via value

Usage

```r
getNLinesLabel(
  gg,
  value,
  elName = c("x", "y", "title", "caption"),
  elNLines = NULL
)
```

Arguments

- **gg**: ggplot2 object
- **value**: String with label value.
- **elName**: string with name of label to extract, among 'x', 'y' and 'title'
- **elNLines**: (optional) Named integer with number of lines, by default 2 for 'x'/‘y’, 3 for ‘title’ and 1 for caption.

Value

integer with (approximated) number of lines
getNLinesLegend

Author(s)
Laure Cougnaud

Description
Get number of lines in the legend, either from directly the `ggplot2` object, or from the values of the legend (legendValues) and title (legendTitle).

Usage
getNLinesLegend(gg, values, title)

Arguments
- `gg` ggplot2 object
- `values` Vector with unique legend values
- `title` String, title for the plot.

Value
integer with (approximated) number of lines

Author(s)
Laure Cougnaud

getNLinesSubjectProfile

Description
Get approximately the number of 'lines' in the vertical direction of a subject profile.

Usage
getNLinesSubjectProfile(gg)

Description
This is extracted from the presence of labels in the y-axis, labels and title in the x-axis, general title and number of lines in the legend. Can be used to specify plot-specific height during the export.

Usage
getNLinesSubjectProfile(gg)
getOptimalColWidth

Arguments

gg  
ggplot2 object, subset of the output of the subjectProfile[X]Plot function, for a particular subject/module/page.

Value

integer with (approximated) number of lines

Author(s)

Laure Cougnaud

gOptimalColWidth  
Get optimal column widths, based on the minimum word size and median number of characters in each column.

Description

Get optimal column widths, based on the minimum word size and median number of characters in each column.

Usage

gOptimalColWidth(  
data,  
widthValue = ifelse(formatReport$landscape, 240, 190),  
labels = NULL,  
formatReport = subjectProfileReportFormat()  
)

Arguments

data  
Data.frame with columns for which optimal width should be extracted.

widthValue  
max number of characters in the codeparamValueVar parameter.

labels  
(optional) Character vector with column labels for data.

formatReport  
list with parameters used to specify the format of the report, e.g. output of the subjectProfileReportFormat function

Value

Numeric vector of length(ncol(data)) with optimal widths.

Author(s)

Laure Cougnaud
getPageVar

Get variable with page of the plot, used for automatic paging of a plot

Description

Get variable with page of the plot, used for automatic paging of a plot

Usage

getPageVar(
  data,
  var,
  typeVar = c("y", "panel"),
  formatReport = subjectProfileReportFormat(),
  title = TRUE,
  xLab = TRUE,
  caption = TRUE,
  paging = TRUE,
  table = FALSE
)

Arguments

data data.frame with data
var string, variable of data with variable for the y-axis
typeVar string, type of the variable, either: 'y': the variable is displayed in the x-axis or 'panel': the variable is displayed as separated facets. This is used to compute height for each line of the plot.
formatReport list with parameters used to specify the format of the report, e.g. output of the subjectProfileReportFormat function
title logical, has the plot a title?
xLab logical, has the plot a label for the x-axis?
caption logical, has the plot a caption?
paging Logical, if TRUE (by default), automatic paging is enabled, so patient profiles module too big to fit in one page will span multiple pages. Please note that the size of the graphic window (or report page) may need to be re-sized in order that the plot fits.
If FALSE, the entire plot is included in one single page.
table Logical, if TRUE the paramValueVar variables are displayed as table (so are not concatenated).

Value

input data with additional column 'pagePlot' containing the page for the plot
getShapePalettePatientProfile

Author(s)
Laure Cougnaud

getPathTemplate
Get path of the report template in the patientProfilesVis package

Description
Get path of the report template in the patientProfilesVis package

Usage
getPathTemplate(file)

Arguments
file file name (with extension)

Value
String with path to the template in the installed patientProfilesVis package

getShapePalettePatientProfile

Get a shape palette for patient profile visualizations.

Description
This is a simple wrapper around getShapePalette, with different defaults:

• inclusion of missing values by default (includeNA set to TRUE)
• the extraction of shapes as text by default (asText set to TRUE)

Usage
getShapePalettePatientProfile(..., includeNA = TRUE, asText = TRUE)
getSplitVectorByInt

Arguments

... Arguments passed on to \texttt{clinUtils::getShapePalette}

\textit{n} \hspace{1cm} \text{Integer of length 1, number of elements in palette.}

\textit{x} \hspace{1cm} \text{Vector with elements used for palette. If factor, the levels are used, otherwise the unique elements of the vector. Missing values are automatically removed, excepted if \texttt{includeNA} is set to TRUE.}

\textit{palette} \hspace{1cm} \text{A vector of custom shapes, or a function returning this vector from a specific number of shapes. The vector should be a character if \texttt{asText} is set to TRUE. Default is the \texttt{clinShapes} shape palette, or \texttt{clinShapesText} if \texttt{asText} is set to TRUE.}

\textit{includeNA} \hspace{1cm} \text{Logical (TRUE by default), should NA elements be retained in the palette in case \textit{x} is specified?}

\textit{asText} \hspace{1cm} \text{Logical (TRUE by default), should the palette be expressed as integer (base R plot and ggplot2 compatible) or in text format (e.g. required if combined with unicode symbols in ggplot2)?}

Value

Vector of shapes, named with the elements in \textit{x} if \textit{x} is specified.

Author(s)

Laure Cougnaud

See Also

getShapePalette

\begin{verbatim}
getSplitVectorByInt \hspace{1cm} Split/combine a vector of size(s) to have a fixed combined size
\end{verbatim}

Description

Split/combine a vector of size(s) to have a fixed combined size

Usage

\texttt{getSplitVectorByInt(sizes, max = NULL)}

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{sizes}</td>
<td>vector with size</td>
</tr>
<tr>
<td>\textit{max}</td>
<td>integer with maximum combined size in output, Inf by default.</td>
</tr>
</tbody>
</table>
getTimeLimSubjectProfilePlots

Get the limits to set for the subject profile plots, depending on the alignment policy set.

Description

These limits are extracted from specified timeLim for each module (stored in the attributes()$metaData$timeLim), and if empty for all modules: from the maximal range of the x-coordinates across all plots.

Usage

gTimeLimSubjectProfilePlots(
  listPlots,
  timeAlign = "all",
  timeAlignPerSubject = "none"
)

Arguments

listPlots list of list of subjectProfile[X]Plot plots

timeAlign Character vector with time alignment across modules/subjects, either:
  • 'all' (by default): all plots have the same time limits
  • 'none': each of the plot (module*subject) has its own time limits
  • character vector with names of the modules which should have the same time limits (should correspond to the names of listPlots)

timeAlignPerSubject Character vector, specifying if the plots should be aligned (or not) across subjects
  • 'none' (by default): all modules to align have the same time limit across subjects
  • 'all': all modules to align have different time limits per subject
  • character vector with subset of the modules to align per subject (should correspond to the names of listPlots)

Only the modules already specified in timeAlign can be aligned by subject.
**getTimeTrans**

**Value**

Time limits, as a numeric vector of length 2. If time limits should be set by module, named list with time limits by module. If time limits should be set by module and subject, nested list with time limits 1) by module 2) by subject.

The names of the list contains the module/subject name extracted from the names of listPlots. The time limits are only returned if they will need to be explicitly set for a plot. Otherwise, NULL is returned.

**Author(s)**

Laure Cougnaud

---

**getTimeTrans**  
*Get useful transformation for the time variable in patient profiles.*

**Description**

Get useful transformation for the time variable in patient profiles.

**Usage**

```r
getTimeTrans(
  type = c("asinh", "asinh-neg"),
  scale = 1,
  formatFct = prettyNum,
  n = 10
)
```

**Arguments**

- `type` String with transformation type, either:
  - 'asinh': hyperbolic arc-sine (`asinh`) transformation
  - 'asinh-neg': hyperbolic arc-sine transformation only for the negative values, otherwise linear scale

- `scale` Numeric vector of length 1 (1 by default) with size of the linear region around 0, only used if in case `type` is: 'asinh'.
  If specified, the time variable is first scaled with: \( x / \text{scale} \), then transformed.

- `formatFct` function formatting the time axis breaks, (`prettyNum` by default), see format parameter of the `trans_new`.

- `n` Integer of length 1 with number of breaks, 10 by default.

**Value**

`ggplot2` transformation (see `trans_new`)
**GetWidthPlot**

Get width for a plot for a certain page layout

**Author(s)**

Pieter-Jan Stiers, Laure Cougnaud

**Description**

Get width for a plot for a certain page layout

**Usage**

```r
GetWidthPlot(
  margin = subjectProfileReportFormat()$margin,
  landscape = subjectProfileReportFormat()$landscape,
  aspectRatio = subjectProfileReportFormat()$aspectRatio
)
```

**Arguments**

- **margin**: Numeric of length 1, with margin in inches.
- **landscape**: Logical, if TRUE the created report is in landscape format. FALSE by default, the report is created in portrait format.
- **aspectRatio**: Numeric of length 1 (0.75 by default) with ratio between size of image in inches (derived from specified margin, landscape and heightLineIn) and real size for exported image.

**Value**

width for the plot in inches

**Author(s)**

Laure Cougnaud
interactionWithMissing

Get interaction variable between different variables.

Description

This ensures that missing values in one of the variable(s) don’t propagate, so the combined result will be: 'NA - a', and that the levels of the combined vector are sorted as the levels of the specified variables (levels of the first variable varying first).

Usage

interactionWithMissing(data, vars, varSep = " - ")

Arguments

data Data.frame with data.
vars Character vector with variable(s) of interest.
varSep String with separator to which the variable(s) should be combined.

Value

Vector of length: nrow(data), with interaction vector.

Author(s)

Laure Cougnaud

isSubjectProfileTimeVariant

Check if the all profile(s) is/are 'time-variant', so not a subject profile 'text' module or empty plot

Description

Check if the all profile(s) is/are 'time-variant', so not a subject profile 'text' module or empty plot

Usage

isSubjectProfileTimeVariant(gg, empty = TRUE)

Arguments

gg object of class subjectProfileX (and ggplot) or potentially nested list of such objects.
empty Logical, should empty subject profile be considered as time-variant?
patientProfilesVis-common-args

Arguments used across the functions of the patientProfilesVis package.

Description

Arguments used across the functions of the patientProfilesVis package.

Arguments

data  Data.frame with data.
colorVar  String, variable of data with color.
colorLab  String, label for colorVar.
colorPalette  Named vector with color palette. The variable should be named with the corresponding element in colorVar. Colors can also be defined for the entire session, by setting options(patientProfilesVis.colors = X) with X either:
  • a vector with colors
  • a function returning a vector of colors for a specified number of elements (viridis by default)
shapeVar  String, variable of data for shape of the points. By default, same as colorVar.
shapeLab  String, label for shapeVar. Set by default to colorLab if colorVar but not shapeVar is not specified.
shapePalette  Named character vector with shape palette for shapeVar. The variable should be named with the corresponding element in shapeVar. Shapes can also be defined for the entire session, by setting options(patientProfilesVis.shapes = X) with X either:
  • a vector with shapes
  • a function returning a vector of shapes for a specified number of elements
Note it is advised to specify the shapes as character, e.g. 'cross' instead of 4, in case Unicode symbols should also be used.
paramGroupVar  (optional) Character vector with variable(s) of data based on which the data will be grouped and sorted (in the y-axis) in the plot.
xLab  String, label for the x-axis.
yLab  String, label for the y-axis.
label  
String, label for the visualization. This label is stored as attributes of the output from the subjectProfile[]Plot function. This label is displayed in the final profile report, in case no data is available for a specific patient, as: 'No [label] available.'

title  
String, title for the plot.

timeVar  
String, variable of data with time, displayed in the x axis. Records with missing time are not displayed in the plot.

timeLab  
String, label for timeVar. This is used in the message indicating missing values for timeVar, and for the default label of the x-axis.

paramVar  
Character vector with variable(s) of data with parameters. Variable content is displayed in the y-axis.

paramLab  
Named character vector, with label for the parameter variable(s) (paramVar). This is used to set the default title.

timeLim  
(optional) Vector of length 2 with time limits (x-axis). If not specified, these are extracted from the minimum timeStartVar and maximum timeEndVar per subject. The time limits are stored as attributes of the plots, used to align the plots in the final report.

timeTrans  
transformation for the time variable, (see trans parameter in scale_x_continuous, and trans_new). For example, produced by the getTimeTrans function.

timeExpand  
Vector of range expansion constants for the time axis (see expand parameter in scale_x_continuous).

listPlots  
Named list of subject profiles. Each sublist contains subject profiles as returned by the subjectProfile[X]Plot function, so nested by subject and page. The names of the list should be unique, and are used

alpha  
Numeric with transparency, 1 by default.

labelVars  
Named character vector with variable labels (names are the variable code)

subjectVar  
String, variable of data with subject ID

formatReport  
list with parameters used to specify the format of the report, e.g. output of the subjectProfileReportFormat function

paramVarSep  
string with character(s) used to concatenate multiple paramVar, ' - ' by default.

Value

No return value, used for the documentation of the functions of the package.
prepareSubjectProfile

Arguments

includeNA Logical (TRUE by default), should NA elements be retained in the palette in case x is specified?

Value

No return value, used for the documentation of the palette functions of the package.

prepareSubjectProfile prepare list of subject profile (s) to be combined with the combineVerticallyGGplot

Description

prepare list of subject profile (s) to be combined with the combineVerticallyGGplot

Usage

prepareSubjectProfile(
  ..., labels,
  timeLim = NULL,
  refLines = NULL,
  refLinesData = NULL,
  refLinesTimeVar = NULL,
  refLinesLabelVar = NULL,
  subjectVar = "USUBJID",
  timeTrans = NULL,
  timeExpand = NULL
)

Arguments

... list of subject profiles (across modules)
labels string with labels for the plots
timeLim time limits, as returned by the getTimeLimSubjectProfilePlots function.
refLines (optional) nested list with details for reference line(s). Each sublist contains:
  • (required) 'label': string with label for the reference line
  • (required) 'time': unique time (x) coordinate for the reference line
  • (optional) 'color': color for the reference line, 'black' by default
  • (optional) 'linetype': linetype for the reference line, 'dotted' by default
refLinesData data.frame with data from which the reference line(s) should be extracted
refLinesTimeVar string, variable of refLinesData with time for reference line(s)
sortSubjects

refLinesLabelVar  string, variable of refLinesData with label for reference line(s)
subjectVar       String, variable of data with subject ID
timeTrans        Time transformation, or list of such transformation named by module. If NULL, no transformation are done.
timeExpand       Vector of range expansion constants for the time axis (see expand parameter in scale_x_continuous).

Value

subjectProfilePlot object, containing the combined profile plots

Author(s)

Laure Cougnaud

---

sortSubjects  Sort subjects based on a specified dataset/variable.

Description

Sort subjects based on a specified dataset/variable.

Usage

sortSubjects(
subjects,
subjectVar = "USUBJID",
subjectSortData = NULL,
subjectSortVar = NULL,
subjectSortDecreasing = FALSE,
verbose = FALSE
)

Arguments

subjects         Character vector with subjects of interest
subjectVar       String, variable of data with subject ID
subjectSortData  Data.frame with data containing information on how the subjects should be sorted.
subjectSortVar   Variable(s) of subjectSortData used to order the subjects
subjectSortDecreasing Logical, if TRUE (FALSE by default) subjects are sorted based on inverse order of subjectSortVar.
verbose          logical, if TRUE print messages during execution
Value

Updated subjects

Author(s)

Laure Cougnaud

subjectProfileCombine  Combine subject profile plots.

Description

Visualizations of different modules are combined by subject. The plots are aligned in the time axis (if requested). If the plots should be aligned:

- the same time limits are set for all plots
- the time axis is transformed if any of the plot was created with a time transformation
- the time axis is expanded for all plots if any of the plot was created with a time axis expanded. The expansion object for the combined plot is created from the max of each expansion element across modules.

If some plots are missing for a specific subject, an empty plot is created, containing information as a text based on the label with which the plot was created.

Usage

subjectProfileCombine(
  listPlots,
  timeLim = NULL,
  timeAlign = "all",
  timeAlignPerSubject = "none",
  subjectVar = "USUBJID",
  maxNLines = NULL,
  refLines = NULL,
  refLinesData = NULL,
  refLinesTimeVar = NULL,
  refLinesLabelVar = NULL,
  shiny = FALSE,
  verbose = FALSE,
  nCores = 1,
  reportPerSubject = FALSE
)
Arguments

listPlots listPlots per subject as created inside the `subjectProfileCombine` function.
timeLim time limits, as returned by the `getTimeLimSubjectProfilePlots` function.
timeAlign Character vector with time alignment across modules/subjects, either:
  • 'all' (by default): all plots have the same time limits
  • 'none': each of the plot (module*subject) has its own time limits
  • character vector with names of the modules which should have the same
time limits (should correspond to the names of listPlots)
timeAlignPerSubject Character vector, specifying if the plots should be aligned (or not) across subjects
  • 'none' (by default): all modules to align have the same time limit across subjects
  • 'all': all modules to align have different time limits per subject
  • character vector with subset of the modules to align per subject (should correspond to the names of listPlots)
subjectVar String, variable of data with subject ID
maxNLines Maximum number of lines for a combined plot, to fit in the page height.
refLines (optional) nested list with details for reference line(s). Each sublist contains:
  • (required) 'label': string with label for the reference line
  • (required) 'time': unique time (x) coordinate for the reference line
  • (optional) 'color': color for the reference line, 'black' by default
  • (optional) 'linetype': linetype for the reference line, 'dotted' by default
refLinesData data.frame with data from which the reference line(s) should be extracted
refLinesTimeVar string, variable of refLinesData with time for reference line(s)
refLinesLabelVar string, variable of refLinesData with label for reference line(s)
shiny logical, set to TRUE (FALSE by default) if the report is generated from a Shiny application. Messages during report creation will be included in the Shiny interface, and it will be mentioned at the end of the report. In this case, the shiny package should be available.
verbose logical, if TRUE print messages during execution
nCores Integer containing the number of cores used for the computation (1 by default). If more than 1, computation is parallelized, in this case the package parallel is required.
reportPerSubject Logical, if TRUE (FALSE by default) export a subject profile report by subject.
Value

a nested list of `ggplot` object, containing the combined profile plots across modules for each subject/page. Each plot object contains in the associated attribute: `metaData` containing: `nLines`: an estimation of the number of 'lines' each plot occupies (e.g. to set height of the exported figure).

Author(s)

Laure Cougnaud

subjectProfileEventPlot

Visualize events in subject profiles, so event with a single time.

Description

Visualize events in subject profiles, so event with a single time.

Usage

```r
subjectProfileEventPlot(
  data,
  paramVar,
  paramLab = getLabelVar(paramVar, labelVars = labelVars),
  paramVarSep = " - ",
  paramGroupVar = NULL,
  colorVar = NULL,
  colorLab = getLabelVar(colorVar, labelVars = labelVars),
  colorPalette = NULL,
  shapeVar = colorVar,
  shapeLab = if (isTRUE(colorVar == shapeVar)) {
    colorLab
  } else
    getLabelVar(shapeVar, labelVars = labelVars),
  shapePalette = NULL,
  alpha = 1,
  timeVar,
  timeLab = getLabelVar(timeVar, labelVars = labelVars),
  timeTrans = NULL,
  timeExpand = NULL,
  subjectVar = "USUBJID",
  subjectSubset = NULL,
  subjectSample = NULL,
  seed = 123,
  subsetData = NULL,
  subsetVar = NULL,
  subsetValue = NULL,
```

Arguments

data       Data.frame with data.
paramVar   Character vector with variable(s) of data with parameters. Variable content is displayed in the y-axis.
paramLab   Named character vector, with label for the parameter variable(s) (paramVar). This is used to set the default title.
paramVarSep string with character(s) used to concatenate multiple paramVar, ' - ' by default.
paramGroupVar (optional) Character vector with variable(s) of data based on which the data will be grouped and sorted (in the y-axis) in the plot.
colorVar   String, variable of data with color. This variable is used for the colors and the filling of the points.
colorLab   String, label for colorVar.
colorPalette Named vector with color palette. The variable should be named with the corresponding element in colorVar.
Colors can also be defined for the entire session, by setting options(patientProfilesVis.colors = X) with X either:
  • a vector with colors
  • a function returning a vector of colors for a specified number of elements (viridis by default)
shapeVar   String, variable of data for shape of the points. By default, same as colorVar.
shapeLab   String, label for shapeVar. Set by default to colorLab if colorVar but not shapeVar is not specified.
shapePalette Named character vector with shape palette for shapeVar. The variable should be named with the corresponding element in shapeVar.
Shapes can also be defined for the entire session, by setting options(patientProfilesVis.shapes = X) with X either:
  • a vector with shapes
  • a function returning a vector of shapes for a specified number of elements
alpha      Numeric with transparency, 1 by default.
timeVar    String, variable of data with time, displayed in the x axis.
Records with missing time are not displayed in the plot.
timeLab  String, label for timeVar. This is used in the message indicating missing values for timeVar, and for the default label of the x-axis.

timeTrans  transformation for the time variable, (see trans parameter in scale_x_continuous, and trans_new). For example, produced by the getTimeTrans function.

timeExpand  Vector of range expansion constants for the time axis (see expand parameter in scale_x_continuous).

subjectVar  String, variable of data with subject ID

subjectSubset  (optional) Character vector with subjects of interest (available in subjectVar), NULL by default.

subjectSample  (optional) Integer of length 1 with number of random subject(s) that should be considered, e.g. to check the created patient profiles for a subset of the data. By default, all specified subjects are considered (set to NULL).

seed  (optional) Integer of length 1 with seed used to select random subjects if subjectSample is specified (123 by default).

subsetData  (optional) Data.frame with extra dataset to filter on. This dataset is filtered, and only records from data with common subject IDs will be retained. If not specified, data is used.

subsetVar  (optional) String with variable of subset data to filter on.subsetValue should be specified too.

If not specified, all records from the subset data are retained.

subsetValue  (optional) Character vector with value(s) of interest to retain in the filtered data. These values should be available in subsetVar.

Missing values in the subject variable are not retained in the filtered data.

xLab  String, label for the x-axis.

yLab  String, label for the y-axis.

timeLim  (optional) Vector of length 2 with time limits (x-axis). If not specified, these are extracted from the minimum timeStartVar and maximum timeEndVar per subject.

The time limits are stored as attributes of the plots, used to align the plots in the final report.

title  String with title, label of the parameter variable by default.

label  String, label for the visualization. This label is stored as attributes of the output from the subjectProfile[]Plot function.

This label is displayed in the final profile report, in case no data is available for for a specific patient, as: 'No [label] available.'

labelVars  Named character vector with variable labels (names are the variable code)

formatReport  list with parameters used to specify the format of the report, e.g. output of the subjectProfileReportFormat function

tpaging  Logical, if TRUE (by default), automatic paging is enabled, so patient profiles module too big to fit in one page will span multiple pages. Please note that the size of the graphic window (or report page) may need to be re-sized in order that the plot fits.

If FALSE, the entire plot is included in one single page.
subjectProfileExport

Value

list of (across subjects) of list (across modules) of ggplot2 objects, also of class subjectProfileEventPlot, with additional metaData attributes containing 'label', 'timeLim' and 'timeTrans' (if specified).

Author(s)

Laure Cougnaud

See Also

Other patient profiles plotting function: subjectProfileIntervalPlot(), subjectProfileLinePlot(), subjectProfileTextPlot()

Description

Create report

Usage

subjectProfileExport(
  listPlotsSubject,
  outputFile = "subjectProfile.pdf",
  index = NULL,
  formatReport = subjectProfileReportFormat(),
  shiny = FALSE,
  verbose = FALSE,
  nCores = NULL,
  exportFigures = FALSE
)

Arguments

listPlotsSubject
  List of plots for each subject
outputFile
  string, path to the output report
index
  Index, output from defineIndex
formatReport
  list with parameters used to specify the format of the report, e.g. output of the subjectProfileReportFormat function
shiny
  logical, set to TRUE (FALSE by default) if the report is generated from a Shiny application. Messages during report creation will be included in the Shiny interface, and it will be mentioned at the end of the report. In this case, the shiny package should be available.
verbose
  logical, if TRUE print messages during execution
subjectProfileIntervalPlot

**subjectProfileIntervalPlot**

*Visualize time interval in subject profiles, so event with a start and end time.*

**Usage**

```r
subjectProfileIntervalPlot(
  data,
  paramVar,
  paramVarSep = " - ",
  paramLab = getLabelVar(paramVar, labelVars = labelVars),
  paramGroupVar = NULL,
  timeStartVar,
  timeStartLab = getLabelVar(timeStartVar, labelVars = labelVars),
  timeEndVar,
  timeEndLab = getLabelVar(timeEndVar, labelVars = labelVars),
  timeLab = toString(c(timeStartLab, timeEndLab)),
  subjectVar = "USUBJID",
  subjectSubset = NULL,
  subjectSample = NULL,
  seed = 123,
  subsetData = NULL,
  subsetVar = NULL,
  subsetValue = NULL,
  timeImpType = c("minimal", "data-based", "none"),
  timeLim = NULL,
  timeLimData = NULL,
)```

**nCores**

Integer containing the number of cores used for the computation (1 by default). If more than 1, computation is parallelized, in this case the package parallel is required.

**exportFigures**

Logical, if TRUE (FALSE by default) the subject profile figures are also exported in pdf format in a 'figures' folder. Figures are named as [subjectID]-[page].pdf

**Value**

No returned value, the plots are exported to outputDir

**Author(s)**

Laure Cougnaud
subjectProfileIntervalPlot

timeLimStartVar = NULL,
timeLimStartLab = getLabelVar(timeLimStartVar, labelVars = labelVars),
timeLimEndVar = NULL,
timeLimEndLab = getLabelVar(timeLimEndVar, labelVars = labelVars),
timeTrans = NULL,
timeExpand = NULL,
timeAlign = TRUE,
xLab = timeLab,
yLab = "",
colorVar = NULL,
colorLab = getLabelVar(colorVar, labelVars = labelVars),
colorPalette = NULL,
alpha = 1,
timeStartShapeVar = NULL,
timeEndShapeVar = NULL,
shapePalette = NULL,
shapeLab = toString(unique(getLabelVar(c(timeStartShapeVar, timeEndShapeVar), labelVars
= labelVars))),'shapeSize = rel(3),
title = toString(getLabelVar(paramVar, labelVars = labelVars, label = paramLab)),
label = title,
labellVars = NULL,
formatReport = subjectProfileReportFormat(),
paging = TRUE
)

Arguments

data
paramVar
paramVarSep
paramLab
paramGroupVar
( optional ) Character vector with variable(s) of data based on which the data will be grouped and sorted (in the y-axis) in the plot.
timeStartVar
String, variable of data with start of time interval.
timeStartLab
String, label for timeStartVar, displayed in a message and in the plot caption.
timeEndVar
String, variable of data with end of time interval.
timeEndLab
String, label for timeEndVar, displayed in a message and in the plot caption.
timeLab
String, label for timeVar. This is used in the message indicating missing values for timeVar, and for the default label of the x-axis.
subjectVar
String, variable of data with subject ID
subjectSubset
(optional) Character vector with subjects of interest (available in subjectVar), NULL by default.
subjectSample (optional) Integer of length 1 with number of random subject(s) that should be considered, e.g. to check the created patient profiles for a subset of the data. By default, all specified subjects are considered (set to NULL).

seed (optional) Integer of length 1 with seed used to select random subjects if subjectSample is specified (123 by default).

subsetData (optional) Data.frame with extra dataset to filter on. This dataset is filtered, and only records from data with common subject IDs will be retained. If not specified, data is used.

subsetVar (optional) String with variable of subset data to filter on. subsetValue should be specified too. If not specified, all records from the subset data are retained.

subsetValue (optional) Character vector with value(s) of interest to retain in the filtered data. These values should be available in subsetVar. Missing values in the subject variable are not retained in the filtered data.

timeImpType String with imputation type: 'minimal' (default), 'data-based' or 'none', see section: 'Time interval representation'. This imputation type is not used if a dataset used to impute time is specified.

timeLim (optional) Vector of length 2 with time limits (x-axis). If not specified, these are extracted from the minimum timeStartVar and maximum timeEndVar per subject. The time limits are stored as attributes of the plots, used to align the plots in the final report.

timeLimData Data.frame with data used to impute time in case some time records are missing in data, see section: 'Time interval representation'.

timeLimStartVar String, variable of timeLimData with start of the time interval.

timeLimStartLab String, label for timeLimStartVar, displayed in a message and in the plot caption.

timeLimEndVar String, variable of timeLimData with end of the time interval.

timeLimEndLab String, label for timeLimEndVar, displayed in a message and in the plot caption.

timeTrans transformation for the time variable, (see trans parameter in scale_x_continuous, and trans_new). For example, produced by the getTimeTrans function.

timeExpand Vector of range expansion constants for the time axis (see expand parameter in scale_x_continuous).

timeAlign Logical, if TRUE (by default) the different plots are horizontally aligned. If set to FALSE, each plot has its own time-limits. If set to FALSE, this is not compatible with the specification of timeLim.

xLab String, label for the x-axis.

yLab String, label for the y-axis.

colorVar String, variable of data with color, used both for the point(s) and segment(s).

colorLab String, label for colorVar.
subjectProfileIntervalPlot

- **colorPalette** Named vector with color palette. The variable should be named with the corresponding element in `colorVar`. Colors can also be defined for the entire session, by setting `options(patientProfilesVis.colors = X)` with `X` either:
  - a vector with colors
  - a function returning a vector of colors for a specified number of elements (viridis by default)

- **alpha** Numeric with transparency, 1 by default.

- **timeStartShapeVar** (optional) String, variable of data used for the shape of the symbol displayed at the start of the time interval. If not specified, default shape palette is used, see section 'Time interval representation'.

- **timeEndShapeVar** String, variable of data used for the shape of the symbol displayed at the end of the time interval. If not specified, default shape palette is used, see section 'Time interval representation'.

- **shapePalette** Named vector with (combined) shape palette for `timeStartShapeVar`/`timeEndShapeVar`.

- **shapeLab** String with label for `timeStartShapeVar`/`timeEndShapeVar`.

- **shapeSize** Size for symbols (only used if `timeStartShapeVar`/`timeEndShapeVar` is specified).

- **title** String with title, label of the parameter variable by default.

- **label** String, label for the visualization. This label is stored as attributes of the output from the `subjectProfile[]Plot` function. This label is displayed in the final profile report, in case no data is available for for a specific patient, as: 'No [label] available.'

- **labelVars** Named character vector with variable labels (names are the variable code)

- **formatReport** list with parameters used to specify the format of the report, e.g. output of the `subjectProfileReportFormat` function

- **paging** Logical, if TRUE (by default), automatic paging is enabled, so patient profiles module too big to fit in one page will span multiple pages. Please note that the size of the graphic window (or report page) may need to be re-sized in order that the plot fits. If FALSE, the entire plot is included in one single page.

**Value**

list of (across subjects) of list (across pages) of ggplot2 objects, also of class subjectProfileIntervalPlot. with additional 'metaData' attributes containing 'label', 'timeLim' `timeTrans` and `timeExpand` (if specified).

**Time interval representation**

In case the start or the end of the time interval contain missing values:
subjectProfileIntervalPlot

• if a dataset \( \text{timeLimData} \), start \( \text{timeLimStartVar} \) and end \( \text{timeLimEndVar} \) variables are specified:
  1. for each subject:
     – the minimum and maximum time values across these specified time variables are extracted
     – missing start values are replaced by the minimum time
     – missing start values are replaced by the maximum time
  2. if all values are missing for this subject, they are taken across subjects
• otherwise, depending on the imputation type \( \text{timeImpType} \):
  – 'minimal' (by default):
    * if the start and the end of the interval are missing: no imputation is done, only the label is displayed
    * if the start time is missing and the end time is not missing: start time is imputed with end time, and status is set to 'Missing start'
    * if the end time is missing and the start time is not missing: end time is imputed with start time, and status is set to 'Missing end'
  – 'data-based' (default in version < 1.0.0): minimum/maximum values in the start/end time variables in the data are considered for the specific subject (if available). If there are missing for a specific subject, they are taken across subjects. If all time are missings, the range is set to 0 and Inf
  – 'none': no imputation is done

The symbols displayed at the start and end of the interval are:

• by default:
  – a filled square labelled 'Complete' if the time is not missing
  – a filled left-directed arrow in case of missing start time
  – a filled right-directed arrow in case of missing end time
• if the variable(s) used for the shape of the start or end of the interval are specified (via \( \text{timeStartShapeVar}/\text{timeEndShapeVar} \)): labels are based on these variables, and a standard shape palette is used

The time limits are the same across subjects, and set to:

• \( \text{timeLim} \) if specified
• maximum time range in \( \text{timeLimStartVar} \) and \( \text{timeLimEndVar} \) in \( \text{timeLimData} \) if specified
• the maximum range on the data obtained after imputation of missing values

Author(s)

Laure Cougnaud

See Also

Other patient profiles plotting function: \texttt{subjectProfileEventPlot()}, \texttt{subjectProfileLinePlot()}, \texttt{subjectProfileTextPlot()}
subjectProfileLinePlot

Visualize subject profiles of the evolution of continuous parameters versus time (spaghetti plot).

Description

Visualize subject profiles of the evolution of continuous parameters versus time (spaghetti plot).

Usage

subjectProfileLinePlot(
  data,
  paramValueVar,
  paramLab = getLabelVar(paramValueVar, labelVars = labelVars),
  paramNameVar = NULL,
  paramVarSep = " - ",
  paramValueRangeVar = NULL,
  colorValueRange = "lightgreen",
  yLimFrom = c("all", "value"),
  colorVar = NULL,
  colorLab = getLabelVar(colorVar, labelVars = labelVars),
  colorPalette = NULL,
  shapeVar = colorVar,
  shapeLab = if (isTRUE(colorVar == shapeVar)) {
    colorLab
  } else {
    getLabelVar(shapeVar, labelVars = labelVars),
  shapePalette = NULL,
  paramGroupVar = NULL,
  timeVar,
  timeLab = getLabelVar(timeVar, labelVars = labelVars),
  timeTrans = NULL,
  timeExpand = NULL,
  subjectVar = "USUBJID",
  subjectSubset = NULL,
  subjectSample = NULL,
  seed = 123,
  subsetData = NULL,
  subsetVar = NULL,
  subsetValue = NULL,
  xLab = timeLab,
  yLab = "",
  timeLim = NULL,
  title = toString(getLabelVar(paramValueVar, labelVars = labelVars, label = paramLab)),
  label = title,
  labelVars = NULL,
formatReport = subjectProfileReportFormat(),
paging = TRUE,
alpha = 1,
shapeSize = rel(1)
)

Arguments

data Data.frame with data.

paramValueVar String, variable of data with parameter value to represent.
Records with missing values are discarded.

paramLab Named character vector, with label for the parameter variable(s) (paramNameVar).
This is used to set the default title.

paramNameVar Character vector with variable(s) of data with parameter name. If multiple, they
are concatenated with paramVarSep.

paramVarSep String with character(s) used to concatenate multiple paramNameVar, ' - ' by
default.

paramValueRangeVar Character vector of length 2 containing variables of data with minimum and
maximum value for paramValueVar, typically reference range indicators.
Range can differ per parameter and even per time point. This range is repre-
sented as a ribbon in the plot background. e.g. to represent the reference range
of the variable.

colorValueRange String with color for the filling of the ribbon represented by paramValueRangeVar.

yLimFrom String with specification on the limits of the y-axis, either:
• 'all' (by default): for each parameter (paramNameVar), the y-axis range
  contains the minimum/maximum value of the reference range (paramValueRangeVar)
or data
• 'value': for each parameter (paramNameVar), the y-axis minimum/maximum
  value is restricted to the data range only. Please note that the ribbon visual-
  izing the reference range is also restricted to the data range if wider.

colorVar String, variable of data with color. This variable is used for the colors and the
filling of the points.

colorLab String, label for colorVar.

colorPalette Named vector with color palette. The variable should be named with the corre-
sponding element in colorVar.
Colors can also be defined for the entire session, by setting options(patientProfilesVis.colors = X) with X either:
• a vector with colors
• a function returning a vector of colors for a specified number of elements
  (viridis by default)

shapeVar String, variable of data for shape of the points. By default, same as colorVar.

shapeLab String, label for shapeVar. Set by default to colorLab if colorVar but not
shapeVar is not specified.
subjectProfileLinePlot

shapePalette Named character vector with shape palette for shapeVar. The variable should be named with the corresponding element in shapeVar. Shapes can also be defined for the entire session, by setting options(patientProfilesVis.shapes = X) with X either:
- a vector with shapes
- a function returning a vector of shapes for a specified number of elements

Note it is advised to specify the shapes as character, e.g. 'cross' instead of 4, in case Unicode symbols should also be used.

paramGroupVar (optional) Character vector with variable(s) of data based on which the data will be grouped and sorted (in the y-axis) in the plot.

timeVar String, variable of data with time, displayed in the x axis. Records with missing time are not displayed in the plot.

timeLab String, label for timeVar. This is used in the message indicating missing values for timeVar, and for the default label of the x-axis.

timeTrans transformation for the time variable, (see trans parameter in scale_x_continuous, and trans_new). For example, produced by the getTimeTrans function.

timeExpand Vector of range expansion constants for the time axis (see expand parameter in scale_x_continuous).

subjectVar String, variable of data with subject ID

subjectSubset (optional) Character vector with subjects of interest (available in subjectVar), NULL by default.

subjectSample (optional) Integer of length 1 with number of random subject(s) that should be considered, e.g. to check the created patient profiles for a subset of the data. By default, all specified subjects are considered (set to NULL).

seed (optional) Integer of length 1 with seed used to select random subjects if subjectSample is specified (123 by default).

subsetData (optional) Data.frame with extra dataset to filter on. This dataset is filtered, and only records from data with common subject IDs will be retained. If not specified, data is used.

subsetVar (optional) String with variable of subset data to filter on. subsetValue should be specified too. If not specified, all records from the subset data are retained.

subsetValue (optional) Character vector with value(s) of interest to retain in the filtered data. These values should be available in subsetVar. Missing values in the subject variable are not retained in the filtered data.

xLab String, label for the x-axis.

yLab String, label for the y-axis.

timeLim (optional) Vector of length 2 with time limits (x-axis). If not specified, these are extracted from the minimum timeStartVar and maximum timeEndVar per subject. The time limits are stored as attributes of the plots, used to align the plots in the final report.

title String with title, label of the parameter value variable by default.
**subjectProfileReportFormat**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>String, label for the visualization. This label is stored as attributes of the output from the subjectProfile function. This label is displayed in the final profile report, in case no data is available for a specific patient, as: 'No [label] available.'</td>
</tr>
<tr>
<td>labelVars</td>
<td>Named character vector with variable labels (names are the variable code)</td>
</tr>
<tr>
<td>formatReport</td>
<td>List with parameters used to specify the format of the report, e.g. output of the subjectProfileReport function</td>
</tr>
<tr>
<td>paging</td>
<td>Logical, if TRUE (by default), automatic paging is enabled, so patient profiles module too big to fit in one page will span multiple pages. Please note that the size of the graphic window (or report page) may need to be re-sized in order that the plot fits. If FALSE, the entire plot is included in one single page.</td>
</tr>
<tr>
<td>alpha</td>
<td>Numeric with transparency, 1 by default.</td>
</tr>
<tr>
<td>shapeSize</td>
<td>Size for the symbols, any integer or object supported by size in geom_point.</td>
</tr>
</tbody>
</table>

**Value**

List of (across subjects) of list (across modules) of `ggplot2` objects, also of class `subjectProfileLinePlot`. Each subject profile contains attributes: 'subjectID' and 'nLines' (estimated number of lines of space the plot will take). The entire list also contains attributes: 'label', 'timeLim' and 'time-Trans' (if specified).

**Author(s)**

Laure Cougnaud

**See Also**

Other patient profiles plotting function: `subjectProfileEventPlot()`, `subjectProfileIntervalPlot()`, `subjectProfileTextPlot()`

---

**subjectProfileReportFormat**

*Get list with format specification for subject profile report.*

**Description**

This format is used to set default for the created subject profile report: line height, margin, report in landscape or portrait format, aspect ratio and width for the y-label.
Usage

subjectProfileReportFormat(
  heightLineIn = 0.2,
  margin = 0.75,
  landscape = FALSE,
  aspectRatio = 0.5,
  yLabelWidth = 30
)

Arguments

  heightLineIn     Numeric of length 1 with height of a line in inches, 0.2 by default.
  margin           Numeric of length 1, with margin in inches.
  landscape        Logical, if TRUE the created report is in landscape format. FALSE by default, the report is created in portrait format.
  aspectRatio      Numeric of length 1 (0.75 by default) with ratio between size of image in inches (derived from specified margin, landscape and heightLineIn) and real size for exported image.
  yLabelWidth      Integer of length 1 with approximate maximum number of characters in the y-label of the plot, 30 by default. If the label of the y-axis is longer than this number of character, it will be splitted between words in separated lines.

Value

List with parameters to set format of the subject profile report. If not specified, default are used.

Author(s)

Laure Cougnaud

subjectProfileTextPlot

Visualize text-information in subject profiles.

Description

There are two ways to specify the variables of interest to include:

  • by specifying column(s) of interest containing parameter values, passed to the paramValueVar parameter.
    In this case, variable value is displayed in the plot area, and variable name in the label of the y-axis, as:
    variable 1 | value 1 - value 2 - ...
    variable 2 | value 1 - value 2 - ...
• by specifying column(s) of interest containing parameter values, displayed as a table.
In this case, variable are displayed in columns in the plot area. Variable names are displayed
on top of table, and associated values below, as:
| variable 1 variable 2
| value 1 value 1 | ... 

• by specifying a combination of a variable containing the parameter name (paramNameVar),
coupled with a variable containing the parameter values (paramValueVar).
In this case, parameter values (if multiple) are concatenated and displayed in the plot area for
each parameter name, displayed in the label of the y-axis, as:
variable name 1 | variable value 1 - variable value 2 - ... 
variable name 2 | variable value 1 - ...

Usage

subjectProfileTextPlot(
  data,
  paramValueVar,
  paramValueLab = getLabelVar(paramValueVar, labelVars = labelVars),
  paramNameVar = NULL,
  paramGroupVar = NULL,
  subsetData = NULL,
  subsetVar = NULL,
  subsetValue = NULL,
  subjectVar = "USUBJID",
  subjectSubset = NULL,
  subjectSample = NULL,
  seed = 123,
  xLab = "",
  yLab = "",
  title = "Subject information",
  label = title,
  labelVars = NULL,
  paramVarSep = " - ",
  formatReport = subjectProfileReportFormat(),
  paging = TRUE,
  table = FALSE,
  colWidth = NULL
)

Arguments

data Data.frame with data.

paramValueVar Character vector, either:

  • vector with names of variable(s) (multiple are possible) of data of interest.
The values are displayed in the plot area and variable name in the labels of the y-axis.
Multiple variables can be concatenated in the same line by specifying them,
as an unique string separated by a 'pipe', e.g. 'SEX|AGE'. Variable label(s) are concatenated (with ', ') and displayed in the y-axis.

• if `paramNameVar` is specified:
  – character vector with names of variable(s) (multiple possible) of data with values to represent in the plot area.
  – function taking `data` as input and returning a new variable (of length equal to number of rows in data) with parameter value to represent

`paramValueLab` (optional) Named character vector with labels for `paramValueVar`.

`paramNameVar` (optional) Character vector of length 1 with variable of data with parameter name. This is displayed in the labels of the y-axis.

`paramGroupVar` (optional) Character vector with variable(s) of data based on which the data will be grouped and sorted (in the y-axis) in the plot.

`subsetData` (optional) Data frame with extra dataset to filter on. This dataset is filtered, and only records from data with common subject IDs will be retained. If not specified, `data` is used.

`subsetVar` (optional) String with variable of subset data to filter on. `subsetValue` should be specified too. If not specified, all records from the subset data are retained.

`subsetValue` (optional) Character vector with value(s) of interest to retain in the filtered data. These values should be available in `subsetVar`. Missing values in the subject variable are not retained in the filtered data.

`subjectVar` String, variable of data with subject ID

`subjectSubset` (optional) Character vector with subjects of interest (available in `subjectVar`), NULL by default.

`subjectSample` (optional) Integer of length 1 with number of random subject(s) that should be considered, e.g. to check the created patient profiles for a subset of the data. By default, all specified subjects are considered (set to NULL).

`seed` (optional) Integer of length 1 with seed used to select random subjects if `subjectSample` is specified (123 by default).

`xLab` String, label for the x-axis.

`yLab` String, label for the y-axis.

`title` String with title, 'Subject information' by default.

`label` String, label for the visualization. This label is stored as attributes of the output from the `subjectProfile[]Plot` function. This label is displayed in the final profile report, in case no data is available for a specific patient, as: 'No [label] available.'

`labelVars` Named character vector with variable labels (names are the variable code)

`paramVarSep` String (' - ' by default) with character(s) used to concatenate multiple variables for the same record in the plot area.

`formatReport` list with parameters used to specify the format of the report, e.g. output of the `subjectProfileReportFormat` function.
subjectProfileTheme

- **paging**: Logical, if TRUE (by default), automatic paging is enabled, so patient profiles module too big to fit in one page will span multiple pages. Please note that the size of the graphic window (or report page) may need to be re-sized in order that the plot fits. If FALSE, the entire plot is included in one single page.

- **table**: Logical, if TRUE (FALSE by default) the information contained in the variables: `paramValueVar` is displayed as a table. Otherwise, the values of the different variables are concatenated in the same line.

- **colWidth**: Numeric vector with approximate width of each parameter value column for a table layout. For example in case two parameters are specified: `c(0.8, 0.2)` such as the first column takes 80% of plot area, and the second column 20%. Note: columns can be slightly bigger if their content is larger than the specified width. If not specified, column width is optimized based on the max length of the character in each column.

**Value**

- list of (across subjects) of list (across modules) of `ggplot2` objects, also of class `subjectProfileTextPlot`, with additional `metaData` attributes containing 'label' and 'timeLim'.

**Author(s)**

Laure Cougnaud

**See Also**

Other patient profiles plotting function: `subjectProfileEventPlot()`, `subjectProfileIntervalPlot()`, `subjectProfileLinePlot()`

---

**subjectProfileTheme**: Custom `ggplot2[theme]` for subject profile plot. Currently classic dark-on-light `ggplot2` theme with alternated grey color for vertical grid lines

**Description**

Custom `ggplot2[theme]` for subject profile plot. Currently classic dark-on-light `ggplot2` theme with alternated grey color for vertical grid lines

**Usage**

`subjectProfileTheme()`

**Value**

`ggplot2[theme]` object
Author(s)

Laure Cougnaud
Index

* patient profiles plotting function
  subjectProfileEventPlot, 38
  subjectProfileIntervalPlot, 42
  subjectProfileLinePlot, 47
  subjectProfileTextPlot, 51

addReferenceLinesProfilePlot, 3

asinh, 29

cat, 12
checkTimeExpand, 4
checkTimeTrans, 5
checkVar, 5
clinShapes, 27
clinShapesText, 27
clinUtils::getColorPalette, 21
clinUtils::getShapePalette, 27
combineVerticallyGGplot, 6, 34
convertAesVar, 7
countNLines, 7
createSubjectProfileReport, 8
defineIndex, 11, 41

definition, 36

filterData, 12
filterMissingInVar, 14
formatParamVarTextPlot, 15
formatTimeInterval, 16
formatTimeLim, 19

ggplot, 6, 31, 38
ggplot2, 3, 4, 22–24, 41, 45, 50, 54

interactionWithMissing, 31

isSubjectProfileTimeVariant, 31

knit, 12

patientProfilesVis-common-args, 32
patientProfilesVis-palette, 33
prepareSubjectProfile, 34
prettyNum, 29

scale_x_continuous, 33, 35, 40, 44, 49
sortSubjects, 35
subjectProfileCombine, 6, 36, 37
subjectProfileEventPlot, 9, 38, 46, 50, 54
subjectProfileExport, 41
subjectProfileIntervalPlot, 9, 41, 42, 50, 54
subjectProfileLinePlot, 9, 41, 46, 47, 54
subjectProfileReportFormat, 11, 16, 24, 25, 33, 40, 41, 45, 50, 50, 53
subjectProfileTextPlot, 9, 16, 41, 46, 50, 51
subjectProfileTheme, 54

trans_new, 29, 33, 40, 44, 49

viridis, 21