Package ‘ggpcp’

July 22, 2025

Type Package

Title Parallel Coordinate Plots in the 'ggplot2' Framework

Version 0.2.0

Date 2022-11-22

Maintainer Heike Hofmann <hofmann@iastate.edu>

Description Modern Parallel Coordinate Plots have been introduced in the 1980s as a way to visualize arbitrarily many numeric variables. This Grammar of Graphics implementation also incorporates categorical variables into the plots in a principled manner. By separating the data managing part from the visual rendering, we give full access to the users while keeping the number of parameters manageably low.

License GPL-3

Imports assertthat (>= 0.2.1), dplyr (>= 1.0.7), ggplot2 (>= 3.3.5), rlang (>= 0.4.11), tibble (>= 3.1.4), tidyselect (>= 1.1.1), tidyr (>= 1.1.3)

Depends R (>= 4.1.0)

Encoding UTF-8

LazyData true

RoxygenNote 7.2.2

Suggests knitr, rmarkdown, purrr, testthat, GGally

URL https://github.com/heike/ggpcp

BugReports https://github.com/heike/ggpcp/issues

NeedsCompilation no

Author Heike Hofmann [aut, cre] (ORCID:
<https://orcid.org/0000-0001-6216-5183>),
Susan VanderPlas [aut] (ORCID: <https://orcid.org/0000-0002-3803-0972>),
Yawei Ge [aut]

Repository CRAN

Date/Publication 2022-11-28 09:30:08 UTC

Contents

---

aes_pcp *Wrapper for aes defaults*

Description

The function provides a mapping from ggpcp internal variable names to the variables' functional purpose in the grammar of graphics framework. Any of the defaults can be overwritten by the user or flexibly expanded by other aesthetic mappings in the usual manner.

Usage

```
aes_pcp(
  x = pcp_x,
  y = pcp_y,
  yend = pcp_yend,
  class = pcp_class,
  group = pcp_id,
  level = pcp_level,
  label = pcp_level,
  ...
)
```

Arguments

`x`	x axis
`y`	y axis
`yend`	end point of line segment
`class`	specifying type of the variable
`group`	identifier
`level`	character string of factor levels

`label` label for factors

`...` other aesthetics are directly passed on to `ggplot2`'s mapping

Value

a list of default mappings for all required aesthetics

See Also

`ggplot2::aes()`

Examples

```
library(ggplot2)
iris |>
  pcp_select(tidyselect::everything()) |>
  pcp_scale() |>
  pcp_arrange() |>
  ggplot(aes_pcp(colour = Species)) +
    geom_pcp() +
    theme_pcp()
```

`Carcinoma` *Data set: Assessment of Carcinoma slides*

Description

A differently formatted data is set available as `carcinoma` in package `poLCA`. Here, pathologists' ratings are recorded

Usage

```
Carcinoma
```

Format

A data frame with 118 rows and 9 variables:

Overall structure

No slide number 1 through 126 (data for slides 14, 20, 21, 50, 75, 97, 109, and 125 are missing)

Average average rating of all eight pathologists.

Pathologist ratings

A scores 1 to 5 of pathologist's A evaluation (1) Negative; (2) Atypical Squamous Hyperplasia; (3) Carcinoma in Situ; (4) Squamous Carcinoma with Early Stromal Invasion; (5) Invasive Carcinoma.

B scores by pathologist B.

C scores by pathologist C.

D scores by pathologist D.

E scores by pathologist E.

F scores by pathologist F.

G scores by pathologist G.

Source

Data published as Table 1 in Landis, J. Richard, and Koch, Gary G. "An Application of Hierarchical Kappa-type Statistics in the Assessment of Majority Agreement among Multiple Observers." Biometrics 33.2 (1977): 363-74, doi:10.2307/2529786.

Study and Design in Holmquist, Nelson D., McMahan C.A., Williams O. Dale. Variability in classification of carcinoma in situ of the uterine cervix. Arch Pathol. 1967 Oct;84(4):334-45. PMID: 6045443, doi:10.1097/00006254196806000000023.

Examples

```
library(ggplot2)
Carcinoma |>
  pcp_select(F, D, C, A, G, E, B, Average) |>
  pcp_scale(method="uniminmax") |>
  pcp_arrange() |>
  ggplot(aes_pcp()) +
    geom_pcp_axes() +
    geom_pcp(aes(colour = Average > 2)) +
    geom_pcp_boxes(colour="black", alpha=0) +
    geom_pcp_labels(aes(label = pcp_level), fill="white", alpha = 1) +
    theme_bw() +
    scale_x_discrete(expand = expansion(add=0.25)) +
   xlab("Pathologist") + ylab("Carcinoma score 1 (Negative) to 5 (Invasive Carcinoma)") +
  theme(axis.text.y=element_blank(), axis.ticks.y=element_blank(), legend.position="none")
```

`GeomPcp_axes` *Proto version of the pcp geoms*

Description

These functions are only exported so that they are visible to the ggplot2 internal functions. User-relevant documentation can be found instead in `geom_pcp()`.

geom_pcp *Generalized Parallel Coordinate plots*

Description

The ggpcp package for generalized parallel coordinate plots is implemented as a ggplot2 extension. In particular, this implementation makes use of ggplot2's layer framework, allowing for a lot of flexibility in the choice and order of showing graphical elements.

command	**graphical element**
geom_pcp	line segments
geom_pcp_axes	vertical lines to represent all axes
geom_pcp_box	boxes for levels on categorical axes
geom_pcp_labels	labels for levels on categorical axes

These ggpcp specific layers can be mixed with ggplot2's regular geoms, such as e.g. ggplot2::geom_point(), ggplot2::geom_boxplot(), ggdensity::geom_hdr(), etc.

Usage

```
geom_pcp(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  axiswidth = c(0, 0.1),
  overplot = "small-on-top",
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

mapping Set of aesthetic mappings created by aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.

data The data to be displayed in this layer. There are three options:

If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot().

A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created.

A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).

`stat` The statistical transformation to use on the data for this layer, either as a `ggproto` `Geom` subclass or as a string naming the stat stripped of the `stat_` prefix (e.g. `"count"` rather than `"stat_count"`)

`position` Position adjustment, either as a string naming the adjustment (e.g. `"jitter"` to use `position_jitter`), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.

`na.rm` If `FALSE` (the default), removes missing values with a warning. If `TRUE` silently removes missing values.

`axiswidth` vector of two values indicating the space numeric and categorical axes are supposed to take. Minimum of 0, maximum of 1.Defaults to 0 for a numeric axis and 0.1 for a categorical axis.

`overplot` character value indicating which method should be used to mitigate overplotting of lines. Defaults to 'small-on-top'. The overplotting strategy 'small-on-top' identifies the number observations for each combination of levels between two categorical variables and plots the lines from highest frequency to smallest (effectively plotting small groups on top). The strategy 'none' gives most flexibility to the user - the plotting order is preserved by the order in which observations are included in the original data.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.

`inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

`...` other arguments passed on to `layer`. These are often aesthetics, used to set an aesthetic to a fixed value, like `color = 'red'` or `size = 3`. They may also be parameters to the paired geom/stat.

Value

a list consisting of a `ggplot2::layer()` object and its associated scales.

About Parallel Coordinate Plots

Parallel coordinate plots are a multivariate visualization that allows several aspects of an observed entity to be shown in a single plot. Each aspect is represented by a vertical axis (giving the plot its name), values are marked on each of these axes. Values corresponding to the same entity are connected by line segments between adjacent axes. This type of visualization was first used by d'Ocagne (1985). Modern re-inventions go back to Inselberg (1985) and Wegman (1990). This implementation takes a more general approach in that it is also able to deal with categorical in the same principled way that allows a tracking of individual observations across multiple dimensions.

Data wrangling

The data pipeline feeding `geom_pcp` is implemented in a three-step modularized form rather than in a `stat_pcp` function more typical for `ggplot2` extensions. The three steps of data pre-processing are:

command	**data processing step**
`pcp_select`	variable selection (and horizontal ordering)
`pcp_scale`	(vertical) scaling of values
`pcp_arrange`	dealing with tie-breaks on categorical axes

Note that these data processing steps are executed before the call to `ggplot2` and the identity function is used by default in all of the `ggpcp` specific layers. Besides the speed-up by only executing the processing steps once for all layers, the separation has the additional benefit, that it provides the users with the possibility to make specific choices at each step in the process. Additionally, separation allows for a cleaner user interface: parameters affecting the data preparation process can be moved to the relevant (set of) function(s) only, thereby reducing the number of arguments without any loss of functionality.

References

M. d'Ocagne. (1885) *Coordonnées parallèles et axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèles.* Gauthier-Villars, page 112, `https://archive.org/details/coordonnesparal00ocaggoog/page/n10`.

Al Inselberg. (1985) *The plane with parallel coordinates.* The Visual Computer, 1(2):69–91, doi:10.1007/BF01898350.

Ed J. Wegman. (1990) *Hyperdimensional data analysis using parallel coordinates.* Journal of the American Statistical Association, 85:664–675, doi:10.2307/2290001.

Examples

```
library(ggplot2)
data(mtcars)
mtcars_pcp <- mtcars |>
  dplyr::mutate(
    cyl = factor(cyl),
    vs = factor(vs),
    am = factor(am),
    gear = factor(gear),
    carb = factor(carb)
  ) |>
  pcp_select(1:11) |>  # select everything
  pcp_scale() |>
  pcp_arrange()

 base <- mtcars_pcp |> ggplot(aes_pcp())
 # Just the base plot:
 base + geom_pcp()

 # with the pcp theme
 base + geom_pcp() + theme_pcp()

 # with boxplots:
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_boxplot(aes(x = pcp_x, y = pcp_y),
   inherit.aes=FALSE,
   data = dplyr::filter(mtcars_pcp, pcp_class!="factor")) +
  theme_pcp()

# base plot with boxes and labels
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_pcp_boxes() +
  geom_pcp_labels() +
  theme_pcp()
```

geom_pcp_axes *Generalized Parallel Coordinate plots*

Description

The ggpcp package for generalized parallel coordinate plots is implemented as a `ggplot2` extension. In particular, this implementation makes use of `ggplot2`'s layer framework, allowing for a lot of flexibility in the choice and order of showing graphical elements.

command	**graphical element**
`geom_pcp`	line segments
`geom_pcp_axes`	vertical lines to represent all axes
`geom_pcp_box`	boxes for levels on categorical axes
`geom_pcp_labels`	labels for levels on categorical axes

These ggpcp specific layers can be mixed with `ggplot2`'s regular geoms, such as e.g. `ggplot2::geom_point()`, `ggplot2::geom_boxplot()`, `ggdensity::geom_hdr()`, etc.

Usage

```
geom_pcp_axes(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

`mapping` Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

`data` The data to be displayed in this layer. There are three options:

If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.

A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.

A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).

`stat` The statistical transformation to use on the data for this layer, either as a `ggproto Geom` subclass or as a string naming the stat stripped of the `stat_` prefix (e.g. `"count"` rather than `"stat_count"`)

`position` Position adjustment, either as a string naming the adjustment (e.g. `"jitter"` to use `position_jitter`), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.

`na.rm` If `FALSE` (the default), removes missing values with a warning. If `TRUE` silently removes missing values.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.

`inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

`...` other arguments passed on to `layer`. These are often aesthetics, used to set an aesthetic to a fixed value, like `color = 'red'` or `size = 3`. They may also be parameters to the paired geom/stat.

Value

a list consisting of a `ggplot2::layer()` object and its associated scales.

About Parallel Coordinate Plots

Parallel coordinate plots are a multivariate visualization that allows several aspects of an observed entity to be shown in a single plot. Each aspect is represented by a vertical axis (giving the plot its name), values are marked on each of these axes. Values corresponding to the same entity are connected by line segments between adjacent axes. This type of visualization was first used by d'Ocagne (1985). Modern re-inventions go back to Inselberg (1985) and Wegman (1990). This implementation takes a more general approach in that it is also able to deal with categorical in the same principled way that allows a tracking of individual observations across multiple dimensions.

Data wrangling

The data pipeline feeding geom_pcp is implemented in a three-step modularized form rather than in a stat_pcp function more typical for ggplot2 extensions. The three steps of data pre-processing are:

command	**data processing step**
pcp_select	variable selection (and horizontal ordering)
pcp_scale	(vertical) scaling of values
pcp_arrange	dealing with tie-breaks on categorical axes

Note that these data processing steps are executed before the call to ggplot2 and the identity function is used by default in all of the ggpcp specific layers. Besides the speed-up by only executing the processing steps once for all layers, the separation has the additional benefit, that it provides the users with the possibility to make specific choices at each step in the process. Additionally, separation allows for a cleaner user interface: parameters affecting the data preparation process can be moved to the relevant (set of) function(s) only, thereby reducing the number of arguments without any loss of functionality.

References

M. d'Ocagne. (1885) *Coordonnées parallèles et axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèles.* Gauthier-Villars, page 112, https://archive.org/details/coordonnesparal00ocaggoog/page/n10.

Al Inselberg. (1985) *The plane with parallel coordinates.* The Visual Computer, 1(2):69–91, doi:10.1007/BF01898350.

Ed J. Wegman. (1990) *Hyperdimensional data analysis using parallel coordinates.* Journal of the American Statistical Association, 85:664–675, doi:10.2307/2290001.

Examples

```
library(ggplot2)
data(mtcars)
mtcars_pcp <- mtcars |>
  dplyr::mutate(
    cyl = factor(cyl),
    vs = factor(vs),
    am = factor(am),
    gear = factor(gear),
    carb = factor(carb)
  ) |>
  pcp_select(1:11) |>  # select everything
  pcp_scale() |>
  pcp_arrange()

 base <- mtcars_pcp |> ggplot(aes_pcp())


 # Just the base plot:
 base + geom_pcp()

 # with the pcp theme
 base + geom_pcp() + theme_pcp()

 # with boxplots:
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_boxplot(aes(x = pcp_x, y = pcp_y),
   inherit.aes=FALSE,
   data = dplyr::filter(mtcars_pcp, pcp_class!="factor")) +
  theme_pcp()

# base plot with boxes and labels
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_pcp_boxes() +
  geom_pcp_labels() +
  theme_pcp()
```

---

geom_pcp_boxes *Generalized Parallel Coordinate plots*

Description

The ggpcp package for generalized parallel coordinate plots is implemented as a ggplot2 extension. In particular, this implementation makes use of ggplot2's layer framework, allowing for a lot of flexibility in the choice and order of showing graphical elements.

command	**graphical element**
geom_pcp	line segments
geom_pcp_axes	vertical lines to represent all axes
geom_pcp_box	boxes for levels on categorical axes
geom_pcp_labels	labels for levels on categorical axes

These ggpcp specific layers can be mixed with ggplot2's regular geoms, such as e.g. ggplot2::geom_point(), ggplot2::geom_boxplot(), ggdensity::geom_hdr(), etc.

Usage

```
geom_pcp_boxes(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  boxwidth = 0.2,
  ...
)
```

Arguments

mapping — Set of aesthetic mappings created by aes(). If specified and inherit.aes = TRUE (the default), it is combined with the default mapping at the top level of the plot. You must supply mapping if there is no plot mapping.

data — The data to be displayed in this layer. There are three options:

If NULL, the default, the data is inherited from the plot data as specified in the call to ggplot().

A data.frame, or other object, will override the plot data. All objects will be fortified to produce a data frame. See fortify() for which variables will be created.

A function will be called with a single argument, the plot data. The return value must be a data.frame, and will be used as the layer data. A function can be created from a formula (e.g. ~ head(.x, 10)).

stat — The statistical transformation to use on the data for this layer, either as a ggproto Geom subclass or as a string naming the stat stripped of the stat_ prefix (e.g. "count" rather than "stat_count")

position — Position adjustment, either as a string naming the adjustment (e.g. "jitter" to use position_jitter), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.

na.rm — If FALSE (the default), removes missing values with a warning. If TRUE silently removes missing values.

show.legend — logical. Should this layer be included in the legends? NA, the default, includes if any aesthetics are mapped. FALSE never includes, and TRUE always includes. It can also be a named logical vector to finely select the aesthetics to display.

inherit.aes — If FALSE, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. borders().

boxwidth — width of the box for a level on a categorical axis, defaults to 0.2.

... — other arguments passed on to layer. These are often aesthetics, used to set an aesthetic to a fixed value, like color = 'red' or size = 3. They may also be parameters to the paired geom/stat.

Value

a list consisting of a ggplot2::layer() object and its associated scales.

About Parallel Coordinate Plots

Parallel coordinate plots are a multivariate visualization that allows several aspects of an observed entity to be shown in a single plot. Each aspect is represented by a vertical axis (giving the plot its name), values are marked on each of these axes. Values corresponding to the same entity are connected by line segments between adjacent axes. This type of visualization was first used by d'Ocagne (1985). Modern re-inventions go back to Inselberg (1985) and Wegman (1990). This implementation takes a more general approach in that it is also able to deal with categorical in the same principled way that allows a tracking of individual observations across multiple dimensions.

Data wrangling

The data pipeline feeding `geom_pcp` is implemented in a three-step modularized form rather than in a `stat_pcp` function more typical for `ggplot2` extensions. The three steps of data pre-processing are:

command	**data processing step**
`pcp_select`	variable selection (and horizontal ordering)
`pcp_scale`	(vertical) scaling of values
`pcp_arrange`	dealing with tie-breaks on categorical axes

Note that these data processing steps are executed before the call to `ggplot2` and the identity function is used by default in all of the `ggpcp` specific layers. Besides the speed-up by only executing the processing steps once for all layers, the separation has the additional benefit, that it provides the users with the possibility to make specific choices at each step in the process. Additionally, separation allows for a cleaner user interface: parameters affecting the data preparation process can be moved to the relevant (set of) function(s) only, thereby reducing the number of arguments without any loss of functionality.

References

M. d'Ocagne. (1885) *Coordonnées parallèles et axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèles.* Gauthier-Villars, page 112, `https://archive.org/details/coordonnesparal00ocaggoog/page/n10`.

Al Inselberg. (1985) *The plane with parallel coordinates.* The Visual Computer, 1(2):69–91, doi:10.1007/BF01898350.

Ed J. Wegman. (1990) *Hyperdimensional data analysis using parallel coordinates.* Journal of the American Statistical Association, 85:664–675, doi:10.2307/2290001.

Examples

```
library(ggplot2)
data(mtcars)
mtcars_pcp <- mtcars |>
  dplyr::mutate(
    cyl = factor(cyl),
    vs = factor(vs),
    am = factor(am),
    gear = factor(gear),
    carb = factor(carb)
  ) |>
  pcp_select(1:11) |>  # select everything
  pcp_scale() |>
  pcp_arrange()

 base <- mtcars_pcp |> ggplot(aes_pcp())


 # Just the base plot:
 base + geom_pcp()

 # with the pcp theme
 base + geom_pcp() + theme_pcp()

 # with boxplots:
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_boxplot(aes(x = pcp_x, y = pcp_y),
   inherit.aes=FALSE,
   data = dplyr::filter(mtcars_pcp, pcp_class!="factor")) +
  theme_pcp()

# base plot with boxes and labels
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_pcp_boxes() +
  geom_pcp_labels() +
  theme_pcp()
```

geom_pcp_labels *Generalized Parallel Coordinate plots*

Description

The ggpcp package for generalized parallel coordinate plots is implemented as a ggplot2 extension. In particular, this implementation makes use of ggplot2's layer framework, allowing for a lot of flexibility in the choice and order of showing graphical elements.

command	**graphical element**
geom_pcp	line segments
geom_pcp_axes	vertical lines to represent all axes
geom_pcp_box	boxes for levels on categorical axes
geom_pcp_labels	labels for levels on categorical axes

These ggpcp specific layers can be mixed with ggplot2's regular geoms, such as e.g. ggplot2::geom_point(), ggplot2::geom_boxplot(), ggdensity::geom_hdr(), etc.

Usage

```
geom_pcp_labels(
  mapping = NULL,
  data = NULL,
  stat = "identity",
  position = "identity",
  na.rm = FALSE,
  show.legend = NA,
  inherit.aes = TRUE,
  ...
)
```

Arguments

`mapping` Set of aesthetic mappings created by `aes()`. If specified and `inherit.aes = TRUE` (the default), it is combined with the default mapping at the top level of the plot. You must supply `mapping` if there is no plot mapping.

`data` The data to be displayed in this layer. There are three options:

If `NULL`, the default, the data is inherited from the plot data as specified in the call to `ggplot()`.

A `data.frame`, or other object, will override the plot data. All objects will be fortified to produce a data frame. See `fortify()` for which variables will be created.

A `function` will be called with a single argument, the plot data. The return value must be a `data.frame`, and will be used as the layer data. A `function` can be created from a `formula` (e.g. `~ head(.x, 10)`).

`stat` The statistical transformation to use on the data for this layer, either as a `ggproto Geom` subclass or as a string naming the stat stripped of the `stat_` prefix (e.g. `"count"` rather than `"stat_count"`)

`position` Position adjustment, either as a string naming the adjustment (e.g. `"jitter"` to use `position_jitter`), or the result of a call to a position adjustment function. Use the latter if you need to change the settings of the adjustment.

`na.rm` If `FALSE` (the default), removes missing values with a warning. If `TRUE` silently removes missing values.

`show.legend` logical. Should this layer be included in the legends? `NA`, the default, includes if any aesthetics are mapped. `FALSE` never includes, and `TRUE` always includes. It can also be a named logical vector to finely select the aesthetics to display.

`inherit.aes` If `FALSE`, overrides the default aesthetics, rather than combining with them. This is most useful for helper functions that define both data and aesthetics and shouldn't inherit behaviour from the default plot specification, e.g. `borders()`.

`...` other arguments passed on to `layer`. These are often aesthetics, used to set an aesthetic to a fixed value, like `color = 'red'` or `size = 3`. They may also be parameters to the paired geom/stat.

Value

a list consisting of a `ggplot2::layer()` object and its associated scales.

About Parallel Coordinate Plots

Parallel coordinate plots are a multivariate visualization that allows several aspects of an observed entity to be shown in a single plot. Each aspect is represented by a vertical axis (giving the plot its name), values are marked on each of these axes. Values corresponding to the same entity are connected by line segments between adjacent axes. This type of visualization was first used by d'Ocagne (1985). Modern re-inventions go back to Inselberg (1985) and Wegman (1990). This implementation takes a more general approach in that it is also able to deal with categorical in the same principled way that allows a tracking of individual observations across multiple dimensions.

Data wrangling

The data pipeline feeding `geom_pcp` is implemented in a three-step modularized form rather than in a `stat_pcp` function more typical for `ggplot2` extensions. The three steps of data pre-processing are:

command	**data processing step**
`pcp_select`	variable selection (and horizontal ordering)
`pcp_scale`	(vertical) scaling of values
`pcp_arrange`	dealing with tie-breaks on categorical axes

Note that these data processing steps are executed before the call to `ggplot2` and the identity function is used by default in all of the `ggpcp` specific layers. Besides the speed-up by only executing the processing steps once for all layers, the separation has the additional benefit, that it provides the users with the possibility to make specific choices at each step in the process. Additionally, separation allows for a cleaner user interface: parameters affecting the data preparation process can be moved to the relevant (set of) function(s) only, thereby reducing the number of arguments without any loss of functionality.

References

M. d'Ocagne. (1885) *Coordonnées parallèles et axiales: Méthode de transformation géométrique et procédé nouveau de calcul graphique déduits de la considération des coordonnées parallèles.* Gauthier-Villars, page 112, `https://archive.org/details/coordonnesparal00ocaggoog/page/n10`.

Al Inselberg. (1985) *The plane with parallel coordinates.* The Visual Computer, 1(2):69–91, doi:10.1007/BF01898350.

Ed J. Wegman. (1990) *Hyperdimensional data analysis using parallel coordinates.* Journal of the American Statistical Association, 85:664–675, doi:10.2307/2290001.

Examples

```
library(ggplot2)
data(mtcars)
mtcars_pcp <- mtcars |>
  dplyr::mutate(
    cyl = factor(cyl),
    vs = factor(vs),
    am = factor(am),
    gear = factor(gear),
    carb = factor(carb)
  ) |>
  pcp_select(1:11) |>  # select everything
  pcp_scale() |>
  pcp_arrange()

 base <- mtcars_pcp |> ggplot(aes_pcp())


 # Just the base plot:
 base + geom_pcp()

 # with the pcp theme
 base + geom_pcp() + theme_pcp()

 # with boxplots:
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_boxplot(aes(x = pcp_x, y = pcp_y),
   inherit.aes=FALSE,
   data = dplyr::filter(mtcars_pcp, pcp_class!=”factor”)) +
  theme_pcp()

# base plot with boxes and labels
 base +
  geom_pcp(aes(colour = cyl)) +
  geom_pcp_boxes() +
  geom_pcp_labels() +
  theme_pcp()
```

nasa *Data set: NASA - Data Expo 2006*

Description

The data are geographic and atmospheric measures on a very coarse 24 by 24 grid covering Central America. This data was provided by the NASA Langley Research Center Atmospheric Sciences Data Center as part of the ASA Data Expo in 2006. Monthly averages of a set of atmospheric measurements are provided for Jan 1995 to Dec 2000. A subset of this data is available from the `GGally` package.

Usage

```
nasa
```

Format

A data frame with 41472 (= 24 x 24 x 72) rows and 15 variables:

Structural variables

time time index for each month from 1 (= Jan 1995) to 72 (= Dec 2000)

id identifier for each grid point 1-1 to 24-24

lat, long geographic latitude and longitude

elevation altitude of the location in meters above sea level

month, year, date year/month of each measurement

Measured variables

cloudlow, cloudmid, cloudhigh Cloud cover (in percent) at low, middle, and high levels.

ozone mean ozone abundance (in dobson)

pressure mean surface pressure (in millibars)

surftemp, temperature mean surface/near surface air temperature (in Kelvin)

Source

https://community.amstat.org/jointscsg-section/dataexpo/dataexpo2006

Examples

```
data(nasa)
library(ggplot2)
nasa |>
  dplyr::filter(id == "1-10") |>
  pcp_select(starts_with("cloud"), ozone, temperature) |>
  pcp_scale() |>
  ggplot(aes_pcp()) +
 geom_pcp(aes(colour=month))
```

pcp_arrange *Data wrangling for GPCPs: Step 3 order observations in factor variables*

Description

Break ties for levels in factor variables, space cases out equally and set an order. Note that only ties in **factor** variables are addressed this way.

Usage

```
pcp_arrange(data, method = "from-right", space = 0.05, .by_group = TRUE)
```

Arguments

data	data frame - preferably processed using pcp_select and pcp_scale.
method	method for breaking ties, one of "from-right", "from-left" or "from-both".
space	number between 0 and 1, indicating the proportion of space used for separating multiple levels.
.by_group	logical value. If TRUE, scaling will respect any previous grouping variables. Applies to grouped data frames only.

Details

The data pipeline feeding any of the geom layers in the ggpcp package is implemented in a three-step modularized form rather than as the stat functions more typical for ggplot2 extensions. The three steps of data pre-processing are:

command	**data processing step**
pcp_select	variable selection (and horizontal ordering)
pcp_scale	(vertical) scaling of values
pcp_arrange	dealing with tie-breaks on categorical axes

Note that these data processing steps are executed before the call to ggplot2 and the identity function is used by default in all of the ggpcp specific layers. Besides the speed-up by only executing the processing steps once for all layers, the separation has the additional benefit, that it provides the users with the possibility to make specific choices at each step in the process. Additionally, separation allows for a cleaner user interface: parameters affecting the data preparation process can be moved to the relevant (set of) function(s) only, thereby reducing the number of arguments without any loss of functionality.

Value

data frame of the same size as the input data; values of pcp_y and pcp_yend are adjusted for pcp_class == "factor"

See Also

pcp_select(), pcp_scale()

Examples

```
library(ggplot2)
data(Carcinoma)
# select scores
pcp_data <- Carcinoma |>
  pcp_select(A:G) |>
  pcp_scale()

# y values are on five different values
table(pcp_data$pcp_y)
```

```
# spread out y values
pcp_data  |> pcp_arrange() |>
  ggplot(aes(x = pcp_y)) + geom_histogram(binwidth=0.05)
```

pcp_scale *Data wrangling for GPCPs: Step 2 scale values*

Description

The function pcp_scale provides access to a set of transformations to use in parallel coordinate plots. All transformations other than raw tend to produce y values in the interval from 0 and 1.

Usage

```
pcp_scale(data, method = "uniminmax", .by_group = TRUE)
```

Arguments

data data frame as returned by select_pcp

method string specifying the method that should be used for scaling the values in a parallel coordinate plot (see Details).

.by_group logical value. If TRUE, scaling will respect any previous grouping variables. Applies to grouped data frames only.

Details

The data pipeline feeding any of the geom layers in the ggpcp package is implemented in a three-step modularized form rather than as the stat functions more typical for ggplot2 extensions. The three steps of data pre-processing are:

command	**data processing step**
pcp_select	variable selection (and horizontal ordering)
pcp_scale	(vertical) scaling of values
pcp_arrange	dealing with tie-breaks on categorical axes

Note that these data processing steps are executed before the call to ggplot2 and the identity function is used by default in all of the ggpcp specific layers. Besides the speed-up by only executing the processing steps once for all layers, the separation has the additional benefit, that it provides the users with the possibility to make specific choices at each step in the process. Additionally, separation allows for a cleaner user interface: parameters affecting the data preparation process can be moved to the relevant (set of) function(s) only, thereby reducing the number of arguments without any loss of functionality.

method is a character string that denotes how to scale the variables in the parallel coordinate plot. Options are named in the same way as the options in GGally::ggparcoord():

- raw: raw data used, no scaling will be done.
- std: univariately, subtract mean and divide by standard deviation. To get values into a unit interval we use a linear transformation of f(y) = y/4+0.5.
- robust: univariately, subtract median and divide by median absolute deviation. To get values into an expected interval of unit interval we use a linear transformation of f(y) = y/4+0.5.
- uniminmax: univariately, scale so the minimum of the variable is zero, and the maximum is one.
- globalminmax: global scaling; the global maximum is mapped to 1, global minimum across the variables is mapped to 0.

Value

data frame of the same size as the input data; values of pcp_y and pcp_yend are scaled according to the specified method.

See Also

pcp_select(), pcp_arrange()

Examples

```
data(Carcinoma)
dim(Carcinoma)
# select all variables
pcp_data <- Carcinoma |> pcp_select(1:9)
summary(pcp_data)
pcp_data |> pcp_scale() |> summary()
# scaling gets values of pcp_y and pcp_yend between 0 and 1
```

pcp_select *Data wrangling for GPCPs: Step 1 variable selection*

Description

The pcp_select function allows a selection of variables from a data set. These variables are transformed into an embellished long form of the data.

Usage

```
pcp_select(data, ...)
```

Arguments

data	a dataframe or tibble
...	choose the columns to be used in the parallel coordinate plot. Variables can be selected by position, name or any of the tidyselect selector functions.

Details

The data pipeline feeding any of the geom layers in the ggpcp package is implemented in a three-step modularized form rather than as the stat functions more typical for ggplot2 extensions. The three steps of data pre-processing are:

command	data processing step
pcp_select	variable selection (and horizontal ordering)
pcp_scale	(vertical) scaling of values
pcp_arrange	dealing with tie-breaks on categorical axes

Note that these data processing steps are executed before the call to ggplot2 and the identity function is used by default in all of the ggpcp specific layers. Besides the speed-up by only executing the processing steps once for all layers, the separation has the additional benefit, that it provides the users with the possibility to make specific choices at each step in the process. Additionally, separation allows for a cleaner user interface: parameters affecting the data preparation process can be moved to the relevant (set of) function(s) only, thereby reducing the number of arguments without any loss of functionality.

Value

dataframe of a long form of the selected variables with extra columns:

variable	functionality
pcp_x, pcp_y	values for the mappings to x and y axes
pcp_yend	vertical endpoint of a line segment
pcp_class	type of each of the input variables
pcp_level	preserves order of levels in categorical variables
pcp_id	identifier for each observation

The dimensions of the returned data set are: 6 + the number of input variables for its columns. The number of rows is given as the multiple of the number of selected variables and the number of rows in the original data.

See Also

pcp_scale(), pcp_arrange()

Examples

```
data(Carcinoma)
dim(Carcinoma)
# select all variables
pcp_data <- Carcinoma |> pcp_select(1:9)
dim(pcp_data) # 6 more columns, 9 times as many observations
head(pcp_data)
```

theme_pcp *Theme for parallel coordinate plots*

Description

The function theme_pcp provides a wrapper for thematic choices suitable for parallel coordinate plots. In particular, the labeling of axes in parallel coordinate plot is quite un-informative. In the default theme axes labels are based on variable names derived during the data wrangling step.

Usage

```
theme_pcp(base_size = 11, base_family = "")
```

Arguments

base_size	base font size, given in pts.
base_family	base font family

Value

A ggplot2 theme object based on ggplot2::theme_bw() without y axis and x axes labels.

See Also

ggplot2::theme_bw()

Examples

```
library(ggplot2)
gg <- iris |>
  pcp_select(tidyselect::everything()) |>
  pcp_scale() |>
  pcp_arrange() |>
  ggplot(aes_pcp(colour = Species)) +
    geom_pcp()

# plot with the default ggplot2 theme
gg
# better:
gg + theme_pcp()
```

Index