Package ‘micompr’

July 8, 2021

Title Multivariate Independent Comparison of Observations

Version 1.1.1

Date 2021-07-08

Maintainer Nuno Fachada <faken@fakenmc.com>

Description A procedure for comparing multivariate samples associated with different groups. It uses principal component analysis to convert multivariate observations into a set of linearly uncorrelated statistical measures, which are then compared using a number of statistical methods. The procedure is independent of the distributional properties of samples and automatically selects features that best explain their differences, avoiding manual selection of specific points or summary statistics. It is appropriate for comparing samples of time series, images, spectrometric measures or similar multivariate observations.

Depends R (>= 3.5.0)

Imports utils, graphics, methods, stats

Suggests biotools, MVN (>= 5.0), deseasonalize, testthat (>= 0.8), knitr, roxygen2, devtools

License MIT + file LICENSE

URL https://github.com/fakenmc/micompr

BugReports https://github.com/fakenmc/micompr/issues

LazyData true

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.1.1

NeedsCompilation no

Author Nuno Fachada [aut, cre] (<https://orcid.org/0000-0002-8487-5837>)

Repository CRAN

Date/Publication 2021-07-08 09:40:02 UTC

R topics documented:

`assumptions` *Parametric tests assumptions*

Description

Generic function to get the assumptions for parametric tests applied to the comparison of output observations.

Usage

```
assumptions(obj)
```

Arguments

obj	Object from which to get the assumptions.

Value

Assumptions for parametric tests applied to the comparison of outputs.

See Also

`assumptions.cmpoutput`, `assumptions.micomp`

`assumptions.cmpoutput` *Get assumptions for parametric tests performed on output comparisons*

Description

Get assumptions for parametric tests performed on output comparisons (i.e. from objects of class `cmpoutput`).

Usage

```
## S3 method for class 'cmpoutput'
assumptions(obj)
```

Arguments

obj	Object of class `cmpoutput`.

Value

Object of class `assumptions_cmpoutput` containing the assumptions for parametric tests performed on an output comparison. Basically a list containing the assumptions for the MANOVA (list of objects of class `assumptions_manova`, one per explained variance) and univariate parametric tests for each principal component (object of class `assumptions_paruv`).

Examples

```
# Create a cmpoutput object from the provided datasets
cmp <- cmpoutput("All", 0.9, pphpc_ok$data[["All"]], pphpc_ok$obs_lvls)

# Get the assumptions for the parametric tests performed in cmp
acmp <- assumptions(cmp)
```

---

assumptions.micomp — *Get assumptions for parametric tests performed on each comparisons*

Description

Get assumptions for parametric tests performed on multiple comparisons (i.e. from objects of class `micomp`).

Usage

```
## S3 method for class 'micomp'
assumptions(obj)
```

Arguments

`obj`	Object of class `micomp`.

Value

Object of class `assumptions_micomp` containing the assumptions for parametric tests performed for the multiple comparisons held by the `mcmp` object. This object is a multi-dimensional list of `assumptions_cmpoutput` objects. Rows are associated with individual outputs, while columns are associated with separate comparisons.

Examples

```
# Create a micomp object, use provided dataset
mic <- micomp(6, 0.8,
              list(list(name = "NLOKvsJEXOK", grpout = pphpc_ok),
                   list(name = "NLOKvsJEXNOSHUFF", grpout = pphpc_noshuff),
                   list(name = "NLOKvsJEXDIFF", grpout = pphpc_diff)))

# Create an object containing the statistic tests evaluating the assumptions
# of the comparisons performed in the mic object
a <- assumptions(mic)
```

assumptions_manova *Determine the assumptions for the MANOVA test*

Description

Determine two assumptions for the MANOVA test: a) multivariate normality of each group; b) homogeneity of covariance matrices.

Usage

```
assumptions_manova(data, factors)
```

Arguments

`data`	Data used for the MANOVA test (rows correspond to observations, columns to dependent variables).
`factors`	Groups to which rows of `data` belong to (independent variables).

Value

An object of class `assumptions_manova` which is a list containing two elements:

`mvntest` List of results from the Royston multivariate normality test (`mvn`), one result per group.

`vartest` Result of Box's M test for homogeneity of covariance matrices (`boxM`).

Note

This function requires the `MVN` and `biotools` packages.

Examples

```
# Determine the assumptions of applying MANOVA to the iris data
# (i.e. multivariate normality of each group and homogeneity of covariance
# matrices)
a <- assumptions_manova(iris[, 1:4], iris[, 5])
```

assumptions_paruv *Determine the assumptions for the parametric comparison test*

Description

Determine two assumptions for the parametric comparison tests (i.e. either `t.test` or `aov`) for each principal component, namely: a) univariate normality of each group; b) homogeneity of variances.

Usage

```
assumptions_paruv(data, factors)
```

Arguments

`data`	Data used in the parametric test (rows correspond to observations, columns to principal components).
`factors`	Groups to which rows of `data` belong to.

Value

An object of class `assumptions_paruv` which is a list containing two elements:

- `uvntest` List of results from the Shapiro-Wilk normality test (`shapiro.test`), one result per group per principal component.
- `vartest` Result of Bartlett test for homogeneity of variances (`bartlett.test`).

Examples

```
# Determine the assumptions of applying ANOVA to each column (dependent
# variable) of the iris data (i.e. normality of each group and homogeneity of
# variances)
a <- assumptions_paruv(iris[, 1:4], iris[, 5])
```

centerscale *Center and scale vector*

Description

Center and scale input vector using the specified method.

Usage

```
centerscale(v, type)
```

Arguments

`v`	Vector to center and scale.
`type`	Type of scaling: "center", "auto", "range", "iqrange", "vast", "pareto", "level" or "none".

Value

Center and scaled vector using the specified method.

References

Berg, R., Hoefsloot, H., Westerhuis, J., Smilde, A., and Werf, M. (2006). Centering, scaling, and transformations: improving the biological information content of metabolomics data. *BMC Genomics* **7**, 142. DOI: 10.1186/1471-2164-7-142

Examples

```
v <- c(-100, 3, 4, 500, 10, 25, -8, -33, 321, 0, 2)

centerscale(v, "center")
# [1] -165.81818  -62.81818  -61.81818  434.18182  -55.81818  -40.81818
# [7]  -73.81818  -98.81818  255.18182  -65.81818  -63.81818

centerscale(v, "auto")
# [1] -0.9308937 -0.3526577 -0.3470437  2.4374717 -0.3133601 -0.2291509
# [7] -0.4144110 -0.5547596  1.4325760 -0.3694995 -0.3582716

centerscale(v, "range")
# [1] -0.2763636 -0.1046970 -0.1030303  0.7236364 -0.0930303 -0.0680303
# [7] -0.1230303 -0.1646970  0.4253030 -0.1096970 -0.1063636

centerscale(v, "iqrange")
# [1] -6.085071 -2.305254 -2.268557 15.933278 -2.048374 -1.497915 -2.708924
# [8] -3.626355  9.364470 -2.415346 -2.341952

centerscale(v, "vast")
# [1] -0.34396474 -0.13030682 -0.12823247  0.90064453 -0.11578638 -0.08467115
# [7] -0.15312466 -0.20498338  0.52933609 -0.13652987 -0.13238117

centerscale(v, "pareto")
# [1] -12.424134  -4.706731  -4.631804  32.531614  -4.182247  -3.058353
# [7]  -5.530919  -7.404075  19.119816  -4.931509  -4.781657

centerscale(v, "level")
# [1] -2.5193370 -0.9544199 -0.9392265  6.5966851 -0.8480663 -0.6201657
# [7] -1.1215470 -1.5013812  3.8770718 -1.0000000 -0.9696133

centerscale(v, "none")
# [1] -100    3    4  500   10   25   -8  -33  321    0    2
```

cmpoutput *Compares output observations from two or more groups*

Description

Compares output observations from two or more groups.

Usage

```
cmpoutput(name, ve_npcs, data, obs_lvls, lim_npcs = TRUE, mnv_test = "Pillai")
```

Arguments

name	Comparison name (useful when calling this function to perform multiple comparisons).
ve_npcs	Percentage (0 < ve_npcs < 1) of variance explained by the q principal components (i.e. number of dimensions) used in MANOVA, or the number of principal components (ve_npcs > 1, must be integer). Can be a vector, in which case the MANOVA test will be applied multiple times, one per specified variance to explain / number of principal components.
data	A n x m matrix, where n is the total number of output observations (runs) and m is the number of variables (i.e. output length).
obs_lvls	Levels or groups associated with each observation.
lim_npcs	Limit number of principal components used for MANOVA to minimum number of observations per group?
mnv_test	The name of the test statistic to be used in MANOVA, as described in summary.manova.

Value

Object of class cmpoutput containing the following data:

scores n x n matrix containing projections of output data in the principal components space. Rows correspond to observations, columns to principal components.

obs_lvls Levels or groups associated with each observation.

varexp Percentage of variance explained by each principal component.

npcs Number of principal components specified in ve_npcs OR which explain the variance percentages given in ve_npcs.

ve Percentage (between 0 and 1) of variance explained by the q principal components (i.e. number of dimensions) used in MANOVA.

name Comparison name (useful when calling this function to perform multiple comparisons).

p.values P-values for the performed statistical tests, namely:

- **manova** List of p-values for the MANOVA test for each number of principal component in npcs.

parametric Vector of p-values for the parametric test applied to groups along each principal component (t-test for 2 groups, ANOVA for more than 2 groups).

nonparametric Vector of p-values for the non-parametric test applied to groups along each principal component (Mann-Whitney U test for 2 groups, Kruskal-Wallis test for more than 2 groups).

parametric_adjusted Same as field `parametric`, but p-values are adjusted using weighted Bonferroni procedure. Percentages of explained variance are used as weights.

nonparametric_adjusted Same as field `nonparametric`, but p-values are adjusted using weighted Bonferroni procedure. Percentages of explained variance are used as weights.

tests **manova** Objects returned by the `manova` function for each value specified in `ve_npcs`.

parametric List of objects returned by applying `t.test` (two groups) or `aov` (more than two groups) to each principal component.

nonparametric List of objects returned by applying `wilcox.test` (two groups) or `kruskal.test` (more than two groups) to each principal component.

Examples

```
# Comparing the first output ("Pop.Sheep") of one the provided datasets.
cmp <-
 cmpoutput("SheepPop", 0.8, pphpc_ok$data[["Pop.Sheep"]], pphpc_ok$obs_lvls)

# Compare bogus outputs created from 2 random sources, 5 observations per
# source, 20 variables each, yielding a 10 x 20 data matrix.
data <- matrix(c(rnorm(100), rnorm(100, mean = 1)), nrow = 10, byrow = TRUE)
olvls <- factor(c(rep("A", 5), rep("B", 5)))
cmp <- cmpoutput("Bogus", 0.7, data, olvls)
```

`concat_outputs` *Concatenate multiple outputs with multiple observations*

Description

Concatenate multiple outputs with multiple observations.

Usage

```
concat_outputs(outputlist, centscal = "none")
```

Arguments

`outputlist`	List of outputs. Each output is a n x m matrix, where n is the number of observations and m is the number of variables (i.e. output length).
`centscal`	Centering and scaling method: "center", "auto", "range", "iqrange", "vast", "pareto", "level" or "none". This task is delegated to the `centerscale` function.

Value

An n x p matrix, representing the n observations of the concatenated output, each observation of length p, which is the sum of individual output lengths.

Examples

```
# Collect 20 observations of 3 outputs with different scales and lengths

# Output 1, length 100
out1 <- matrix(rnorm(2000, mean = 0, sd = 1), nrow = 20)

# Output 2, length 200
out2 <- matrix(rnorm(4000, mean = 100, sd = 200), nrow = 20)

# Output 1, length 50
out3 <- matrix(rnorm(1000, mean = -1000, sd = 10), nrow = 20)

# Concatenate and range scale outputs, resulting matrix dimensions will be
# 20 x 350
outconcat <- concat_outputs(list(out1, out2, out3), "range")
```

---

grpoutputs — *Load and group outputs from files*

Description

Load and group outputs from files containing multiple observations of the groups to be compared.

Usage

```
grpoutputs(
  outputs,
  folders,
  files,
  lvls = NULL,
  concat = F,
  centscal = "range",
  ...
)
```

Arguments

outputs	A vector with the labels of each output, or an integer with the number of outputs (in which case output labels will be assigned automatically). In either case, the number of outputs should account for an additional concatenated output, as specified in the concat parameter.

`folders`	Vector of folder names where to read files from. These are recycled if `length(folders) < length(files)`.
`files`	Vector of filenames or file sets to load in each folder. File sets can be given as regular expressions, or as wildcards by wrapping them with `glob2rx`.
`lvls`	Vector of factor levels (groups). Must be the same length as `files`, i.e. each file set will be associated with a different level or group. If not given, default group names will be used.
`concat`	If TRUE add an additional output which corresponds to the concatenation of all outputs, properly centered and scaled.
`centscal`	Method for centering and scaling outputs if `concat` is TRUE. It can be one of "center", "auto", "range" (default), "iqrange", "vast", "pareto" or "level". Centering and scaling is performed by the `centerscale` function.
`...`	Options passed to `read.table`, which is used to read the files specified in the `files` parameter.

Details

Each file corresponds to an observation, and should have a tabular format where columns correspond to outputs and rows to variables or dimensions. Observations (files) are grouped by factor levels which correspond to the file groups given in the `files` parameter. Factor levels differentiate observations from distinct groups.

Value

Object of class `grpoutputs` containing the following data:

data List of all outputs, each one grouped into a *n* x *m* matrix, where *n* is the total number of output observations and *m* is the number of variables or dimensions (i.e. output length).

groupsize Vector containing number of observations for each level or group.

obs_lvls Factor vector of levels or groups associated with each observation.

lvls Vector of factor levels in the order they occur (as given in parameter with the same name).

concat Boolean indicating if this object was created with an additional concatenated output.

Examples

```
# Determine paths for data folders, each containing outputs for 10 runs of
# the PPHPC model
dir_nl_ok <- system.file("extdata", "nl_ok", package = "micompr")
dir_jex_ok <- system.file("extdata", "j_ex_ok", package = "micompr")
files <- glob2rx("stats400v1*.tsv")

# Create a grouped outputs object using outputs from NetLogo and Java
# implementations of the PPHPC model
go <- grpoutputs(7, c(dir_nl_ok, dir_jex_ok), c(files, files),
                 lvls = c("NL", "JEX"), concat = TRUE)

# Do the same, but specify output names and don't specify levels
go <- grpoutputs(c("a", "b", "c", "d", "e", "f"),
                 c(dir_nl_ok, dir_jex_ok), c(files, files))
```

micomp *Multiple independent comparisons of observations*

Description

Performs multiple independent comparisons of output observations.

Usage

```
micomp(
  outputs,
  ve_npcs,
  comps,
  concat = F,
  centscal = "range",
  lim_npcs = TRUE,
  mnv_test = "Pillai",
  ...
)
```

Arguments

`outputs` A vector with the labels of each output, or an integer with the number of outputs (in which case output labels will be assigned automatically).

`ve_npcs` Percentage (`0` < `ve_npcs` < `1`) of variance explained by the q principal components (i.e. number of dimensions) used in MANOVA, or the number of principal components (`ve_npcs` > `1`, must be integer). Can be a vector, in which case the MANOVA test will be applied multiple times, one per specified variance to explain / number of principal components.

`comps` A list of lists, where each list contains information regarding an individual comparison. Each list can have one of two configurations:

1. Lists with the first configuration are used to load data from files, and require the following fields:
   - **name** A string specifying the comparison name.
   - **folders** Vector of folder names where to read files from. These are recycled if `length(folders)` < `length(files)`.
   - **files** Vector of filenames (with wildcards) to load in each folder.
   - **lvls** Vector of level or group names, must be the same length as `files`, i.e. each file set will be associated with a different group. If not given, default group names will be set.
2. Lists with the second configuration are used to load data from environment variables, and require the following fields:
   - **name** A string specifying the comparison name.
   - **grpout** Either an object of class `grpoutputs` or a list with the following two fields:

data List of all outputs, where tags correspond to output names and values correspond to the output data. Output data is a n x m matrix, where n is the total number of output observations and m is the number of variables (i.e. output length).

obs_lvls Levels or groups associated with each observation.

`concat` Create an additional, concatenated output? Ignored for sublists passed in the `comps` which follow the second configuration.

`centscal` Method for centering and scaling outputs if `concat` is TRUE. It can be one of "center", "auto", "range" (default), "iqrange", "vast", "pareto" or "level". Centering and scaling is performed by the `centerscale` function.

`lim_npcs` Limit number of principal components used for MANOVA to minimum number of observations per group?

`mnv_test` The name of the test statistic to be used in MANOVA, as described in `summary.manova`.

`...` Options passed to `read.table`, which is used to read the files specified in lists using the first configuration in the `comp` parameter.

Value

An object of class `micomp`, which is a two-dimensional list of `cmpoutput` objects. Rows are associated with individual outputs, while columns are associated with separate comparisons.

Examples

```
# Create a micomp object from existing files and folders

dir_nl_ok <-
  system.file("extdata", "nl_ok", package = "micompr")
dir_jex_ok <-
  system.file("extdata", "j_ex_ok", package = "micompr")
dir_jex_noshuff <-
  system.file("extdata", "j_ex_noshuff", package = "micompr")
dir_jex_diff <-
  system.file("extdata", "j_ex_diff", package = "micompr")
files <- glob2rx("stats400v1*.tsv")

mic <- micomp(7, 0.8,
              list(list(name = "NLOKvsJEXOK",
                        folders = c(dir_nl_ok, dir_jex_ok),
                        files = c(files, files),
                        lvls = c("NLOK", "JEXOK")),
                   list(name = "NLOKvsJEXNOSHUFF",
                        folders = c(dir_nl_ok, dir_jex_noshuff),
                        files = c(files, files),
                        lvls = c("NLOK", "JEXNOSHUFF")),
                   list(name = "NLOKvsJEXDIFF",
                        folders = c(dir_nl_ok, dir_jex_diff),
                        files = c(files, files),
                        lvls = c("NLOK", "JEXDIFF"))),
              concat = TRUE)
```

```
# Create a micomp object from package datasets (i.e. grpoutputs objects)
# directly

mic <- micomp(c("o1", "o2", "o3", "o4"), 0.9,
              list(list(name = "NLOKvsJEXOK", grpout = pphpc_ok),
                   list(name = "NLOKvsJEXNOSHUFF", grpout = pphpc_noshuff),
                   list(name = "NLOKvsJEXDIFF", grpout = pphpc_diff)))

# Create a micomp object using manually inserted data

mic <- micomp(6, 0.5, list(
  list(name = "NLOKvsJEXOK",
       grpout = list(data = pphpc_ok$data,
                     obs_lvls = pphpc_ok$obs_lvls)),
  list(name = "NLOKvsJEXNOSHUFF",
       grpout = list(data = pphpc_noshuff$data,
                     obs_lvls = pphpc_noshuff$obs_lvls)),
  list(name = "NLOKvsJEXDIFF",
       grpout = list(data = pphpc_diff$data,
                     obs_lvls = pphpc_diff$obs_lvls))))
```

`micompr` *micompr: multivariate independent comparison of observations*

Description

The micompr R package implements a procedure for comparing multivariate samples associated with different groups. The procedure uses principal component analysis to convert multivariate observations into a set of linearly uncorrelated statistical measures, which are then compared using a number of statistical methods. This technique is independent of the distributional properties of samples and automatically selects features that best explain their differences, avoiding manual selection of specific points or summary statistics. The procedure is appropriate for comparing samples of time series, images, spectrometric measures or similar multivariate observations.

Note

MIT License

Author(s)

Nuno Fachada

References

micompr: An R Package for Multivariate Independent Comparison of Observations, Nuno Fachada, João Rodrigues, Vitor V. Lopes, Rui C. Martins and Agostinho C. Rosa , The R Journal (2016) 8:2, pages 405-420.

See Also

cmpoutput, micomp, grpoutputs

---

plot.assumptions_cmpoutput

Plot p-*values for testing the assumptions of the parametric tests used in output comparison*

Description

Plot method for objects of class assumptions_cmpoutput containing *p*-values produced by testing the assumptions of the parametric tests used for comparing an output.

Usage

```
## S3 method for class 'assumptions_cmpoutput'
plot(x, ...)
```

Arguments

x	Objects of class assumptions_cmpoutput.
...	Extra options passed to plot.default.

Details

Several bar plots are presented, showing the *p*-values yielded by the Shapiro-Wilk (shapiro.test) and Royston tests (mvn) for univariate and multivariate normality, respectively, and for the Bartlett (bartlett.test) and Box's M (boxM) for testing homogeneity of variances and of covariance matrices, respectively. The following bar plots are shown:

- One bar plot for the *p*-values of the Bartlett test, one bar (*p*-value) per individual principal component.
- s bar plots for *p*-values of the Shapiro-Wilk test, where s is the number of groups being compared. Individual bars in each plot are associated with a principal component.
- t bar plot for the *p*-values of the Royston test with s bars each, where t is the number of unique MANOVA tests performed (one per requested explained variances) and s is the number of groups being compared. These plots will not show if there is only one principal component being considered.
- One plot for the *p*-values of the Box's M test, one bar (*p*-value) per unique MANOVA tests performed (one per requested explained variances).

Value

None.

Examples

```
# Create a cmpoutput object from the provided datasets
cmp <- cmpoutput("All", 0.9, pphpc_ok$data[["All"]], pphpc_ok$obs_lvls)

# Display a bar plot with the p-values of the assumptions for the parametric
# tests performed in cmp
plot(assumptions(cmp))
```

---

plot.assumptions_manova

Plot p-values for testing the multivariate normality assumptions of the MANOVA test

---

Description

Plot method for objects of class `assumptions_manova` which presents a bar plot containing the *p*-values produced by the Royston multivariate normality test (`mvn`) for each group being compared.

Usage

```
## S3 method for class 'assumptions_manova'
plot(x, ...)
```

Arguments

x	Objects of class `assumptions_manova`.
...	Extra options passed to `barplot`. The `col` parameter defines colors for *p*-values below 1, 0.05 and 0.01, respectively.

Value

None.

Examples

```
# Plot the Royston test p-value for multivariate normality of each group
# (species) of the iris data
plot(assumptions_manova(iris[, 1:4], iris[, 5]))

# Plot the same data with logarithmic scale for p-values
plot(assumptions_manova(iris[, 1:4], iris[, 5]), log = "y")
```

plot.assumptions_micomp

Plot p-values for testing the assumptions of the parametric tests used in multiple output comparison

Description

Plot method for objects of class assumptions_cmpoutput containing p-values produced by testing the assumptions of the parametric tests used for multiple output comparisons.

Usage

```
## S3 method for class 'assumptions_micomp'
plot(x, ...)
```

Arguments

x	Object of class assumptions_micomp.
...	Extra options passed to barplot.

Details

Several bar plots are presented, one for each comparison and output combination, showing the several statistical tests employed to verify the assumptions of the parametric tests.

Value

None.

Examples

```
# Create a micomp object, use provided dataset
mic <- micomp(6, 0.65,
              list(list(name = "NLOKvsJEXOK", grpout = pphpc_ok),
                   list(name = "NLOKvsJEXNOSHUFF", grpout = pphpc_noshuff),
                   list(name = "NLOKvsJEXDIFF", grpout = pphpc_diff)))

# Plot the p-values of the statistic tests evaluating the assumptions of the
# comparisons performed in the mic object
plot(assumptions(mic))
```

plot.assumptions_paruv

Plot p*-values for testing the assumptions of the parametric tests used in output comparison*

Description

Plot method for objects of class `assumptions_paruv` containing *p*-values produced by testing the assumptions of the parametric tests used for comparing outputs.

Usage

```
## S3 method for class 'assumptions_paruv'
plot(x, ...)
```

Arguments

`x`	Objects of class `assumptions_paruv`.
`...`	Extra options passed to `barplot`. The `col` parameter defines colors for *p*-values below 1, 0.05 and 0.01, respectively.

Details

One bar plot is presented for the Bartlett test (`bartlett.test`), showing the respective *p*-values along principal component. *s* bar plots are presented for the Shapiro-Wilk (`shapiro.test`), where *s* is the number of groups being compared; individual bars in each plot represent the *p*-values associated with each principal component.

Value

None.

Examples

```
# Plot the Shapiro-Wilk and Bartlett test p-values for each dependent
# variable of the iris data
plot(assumptions_paruv(iris[, 1:4], iris[, 5]))

# Plot the same data with logarithmic scale for p-values
plot(assumptions_paruv(iris[, 1:4], iris[, 5]), log = "y")
```

plot.cmpoutput *Plot comparison of an output*

Description

Plot objects of class `cmpoutput`.

Usage

```
## S3 method for class 'cmpoutput'
plot(x, ...)
```

Arguments

`x`	Object of class `cmpoutput`.
`...`	Extra options passed to `plot.default`. The `col` option determines the colors to use on observations of different groups (scatter plot only).

Details

This method produces four sub-plots, namely:

- Scatter plot containing the projection of output observations on the first two dimensions of the principal components space.
- Bar plot of the percentage of variance explain per principal component.
- Bar plot of *p*-values for the parametric test for each principal component.
- Bar plot of *p*-values for the non-parametric test for each principal component.

Value

None.

Examples

```
# Comparing the concatenated output of the pphpc_ok dataset, which
# contains simulation output data from two similar implementations of the
# PPHPC model.

plot(cmpoutput("All", 0.95, pphpc_ok$data[["All"]], pphpc_ok$obs_lvls))
```

`plot.grpoutputs` *Plot grouped outputs*

Description

Plot objects of class `grpoutputs`.

Usage

```
## S3 method for class 'grpoutputs'
plot(x, ...)
```

Arguments

`x`	Object of class `grpoutputs`.
`...`	Extra options passed to `plot.default`.

Details

Each output is plotted individually, and observations are plotted on top of each other. Observations from different groups are plotted with different colors (which can be controlled through the `col` parameter given in ...).

This function can be very slow for a large number of observations.

Value

None.

Examples

```
# Determine paths for the data folder containing outputs of different
# lengths
dir_na <- system.file("extdata", "testdata", "NA", package = "micompr")

# Sets of files A and B have 3 files each
filesA <- glob2rx("stats400v1*n20A.tsv")
filesB <- glob2rx("stats400v1*n20B.tsv")

# Instantiate grpoutputs object
go <-
 grpoutputs(7, dir_na, c(filesA, filesB), lvls = c("A", "B"), concat = TRUE)

# Plot grpoutputs object
plot(go)
```

plot.micomp — *Plot projection of output observations on the first two dimensions of the principal components space*

Description

For each comparison and output combination, draw a scatter plot containing the projection of output observations on the first two dimensions of the principal components space.

Usage

```
## S3 method for class 'micomp'
plot(x, ...)
```

Arguments

`x`	An object of class `micomp`.
`...`	Extra options passed to `plot.default`. The `col` option determines the colors to use on observations of different groups.

Value

None.

Examples

```
plot(micomp(c("SheepPop", "WolfPop", "GrassQty"), 0.95,
            list(list(name = "I", grpout = pphpc_ok),
                 list(name = "II", grpout = pphpc_noshuff),
                 list(name = "III", grpout = pphpc_diff))))
```

pphpc_diff — *Data from two implementations of the PPHPC model, one of which setup with a different parameter*

Description

A dataset containing simulation output data from two implementations of the PPHPC model, one of which setup with a different parameter.

Usage

```
pphpc_diff
```

Format

A grpoutputs object containing simulation output data from 20 runs of the PPHPC model, 10 runs from each implementation. The model has six outputs, but the object contains a seventh output corresponding to the concatenation of the six outputs

Source

Runs are obtained from the NetLogo and Java (EX with 8 threads) implementations of the PPHPC model available at https://github.com/fakenmc/pphpc. The config400v1.txt configuration was used in both cases, with the exception of restart parameter, c_r, in the Java implementation, which was set to 9 instead of 10.

---

pphpc_noshuff *Data from two implementations of the PPHPC model, one of which has agent list shuffling deactivated*

Description

A dataset containing simulation output data from two implementations of the PPHPC model, one of which has agent list shuffling deactivated.

Usage

```
pphpc_noshuff
```

Format

A grpoutputs object containing simulation output data from 20 runs of the PPHPC model, 10 runs from each implementation. The model has six outputs, but the object contains a seventh output corresponding to the concatenation of the six outputs

Source

Runs are obtained from the NetLogo and Java (EX with 8 threads) implementations of the PPHPC model available at https://github.com/fakenmc/pphpc. The config400v1.txt configuration was used in both cases. Runs with the Java implementation were performed with the '-u' option, i.e. with agent list shuffling turned off.

pphpc_ok *Data from two similar implementations of the PPHPC model*

Description

A dataset containing simulation output data from two implementations of the PPHPC model.

Usage

```
pphpc_ok
```

Format

A `grpoutputs` object containing simulation output data from 20 runs of the PPHPC model, 10 runs from each implementation. The model has six outputs, but the object contains a seventh output corresponding to the concatenation of the six outputs

Source

Runs are obtained from the NetLogo and Java (EX with 8 threads) implementations of the PPHPC model available at https://github.com/fakenmc/pphpc. The `config400v1.txt` configuration was used in both cases.

print.assumptions_cmpoutput *Print method for the assumptions of parametric tests used in a comparison of an output*

Description

Print method for objects of class `assumptions_cmpoutput`, which contain the assumptions for the parametric tests used in a comparison of an output.

Usage

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

Arguments

`x`	Object of class `assumptions_cmpoutput`.
`...`	Currently ignored.

Value

None.

Examples

```
# Create a cmpoutput object from the provided datasets
cmp <- cmpoutput("All", c(0.7, 0.8, 0.9),
                 pphpc_diff$data[["All"]], pphpc_diff$obs_lvls)
```

`print.assumptions_manova`
Print information about the assumptions of the MANOVA test

Description

Print information about objects of class `assumptions_manova`, which represent the assumptions of the MANOVA test performed on a comparison of outputs.

Usage

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

Arguments

`x`	Object of class `assumptions_manova`.
`...`	Currently ignored.

Value

The argument `x`, invisibly, as for all `print` methods.

Examples

```
# Print information concerning the assumptions of applying MANOVA to the iris
# data (i.e. multivariate normality of each group and homogeneity of
# covariance matrices)
assumptions_manova(iris[, 1:4], iris[, 5])
```

print.assumptions_micomp

Print information about the assumptions concerning the parametric tests performed on multiple comparisons of outputs

Description

Print information about objects of class `assumptions_micomp`, which represent the assumptions concerning the parametric tests performed on multiple comparisons of outputs.

Usage

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

Arguments

`x`	Object of class `assumptions_micomp`.
`...`	Currently ignored.

Value

The argument `x`, invisibly, as for all `print` methods.

Examples

```
# Create a micomp object, use provided dataset
mic <- micomp(c("SheepPop", "WolfPop", "GrassQty"), 0.7,
              list(list(name = "NLOKvsJEXOK", grpout = pphpc_ok),
                   list(name = "NLOKvsJEXNOSHUFF", grpout = pphpc_noshuff),
                   list(name = "NLOKvsJEXDIFF", grpout = pphpc_diff)))

# Print the results (p-values) of the statistic tests evaluating the
# assumptions of the comparisons performed in the mic object
assumptions(mic)
```

print.assumptions_paruv

Print information about the assumptions of the parametric test

Description

Print information about objects of class `assumptions_paruv`, which represent the assumptions of the parametric test (i.e. either `t.test` or `aov`) performed on a comparison of outputs.

Usage

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

Arguments

x	Object of class assumptions_paruv.
...	Currently ignored.

Value

The argument x, invisibly, as for all print methods.

Examples

```
# Print information about the assumptions of applying ANOVA to each column
# (dependent variable) of the iris data (i.e. normality of each group and
# homogeneity of variances)
assumptions_paruv(iris[, 1:4], iris[, 5])
```

print.cmpoutput *Print information about comparison of an output*

Description

Print information about objects of class cmpoutput.

Usage

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

Arguments

x	Object of class cmpoutput.
...	Currently ignored.

Value

The argument x, invisibly, as for all print methods.

Examples

```
# Comparing the fifth output of the pphpc_diff dataset, which contains
# simulation output data from two implementations of the PPHPC model executed
# with a different parameter.

cmpoutput("WolfPop", 0.7, pphpc_diff$data[[5]], pphpc_diff$obs_lvls)
```

---

print.grpoutputs *Print information about grouped outputs*

Description

Print information about objects of class grpoutputs.

Usage

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

Arguments

x	Object of class grpoutputs.
...	Currently ignored.

Value

The argument x, invisibly, as for all print methods.

Examples

```
# Determine paths for data folders, each containing outputs for 10 runs of
# the PPHPC model
dir_nl_ok <- system.file("extdata", "nl_ok", package = "micompr")
dir_jex_diff <- system.file("extdata", "j_ex_diff", package = "micompr")
files <- glob2rx("stats400v1*.tsv")

# Create a grpoutputs object
go <- grpoutputs(6, c(dir_nl_ok, dir_jex_diff), c(files, files))

# Print information about object (could just type "go" instead)
print(go)
```

print.micomp *Print information about multiple comparisons of outputs*

Description

Print information about objects of class micomp.

Usage

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

Arguments

x	Object of class micomp.
...	Currently ignored.

Value

The argument x, invisibly, as for all print. methods.

Examples

```
# A micomp object from package datasets (i.e. grpoutputs objects) directly

micomp(c("outA", "outB", "outC", "outD"), 0.9,
              list(list(name = "Comp1", grpout = pphpc_ok),
                   list(name = "Comp2", grpout = pphpc_noshuff),
                   list(name = "Comp3", grpout = pphpc_diff)))
```

pvalf *Format p-values*

Description

Generic function to format *p*-values.

Usage

```
pvalf(pval, params)
```

Arguments

pval	Numeric *p*-value to format (between 0 and 1).
params	A list of method-dependent options.

Value

A string representing the formatted *p*-value.

See Also

`pvalf.default`

`pvalf.default` *Default p-value formatting method*

Description

Format a *p*-value for printing in a `LaTeX` table. Requires the *ulem* `LaTeX` package for underlining the *p*-values.

Usage

```
## Default S3 method:
pvalf(pval, params = list())
```

Arguments

`pval`	Numeric value between 0 and 1.
`params`	A list of options. This function accepts the following options: **minval** If *p*-value is below this value, return this value preceded by a "<" sign instead instead. **lim1val** If *p*-value is below this value, it will be double-underlined. **lim2val** If *p*-value is below this value, it will be underlined. **na_str** String to use for NAs. By default NAs are returned as is.

Value

A string representing the formatted `pval`.

Examples

```
pvalf(0.1)
pvalf(0.000001)
pvalf(c(0.06, 0.04, 0.005, 0.00001), list(minval = 0.0001))
```

`summary.assumptions_cmpoutput`

Summary method for the assumptions of parametric tests used in a comparison of an output

Description

Summary method for objects of class `assumptions_cmpoutput`, which contain the assumptions for the parametric tests used in a comparison of an output.

Usage

```
## S3 method for class 'assumptions_cmpoutput'
summary(object, ...)
```

Arguments

`object`	Object of class `assumptions_cmpoutput`.
`...`	Currently ignored.

Value

A list with the following items:

manova A matrix of p-values for the MANOVA assumptions. All rows, expect the last one, correspond to the Royston test for multivariate normality for each group; the last row corresponds to Box's M test for homogeneity of covariance matrices. Columns correspond to number of principal components required to explain the percentage of user-specified variance.

ttest A matrix of p-values for the t-test assumptions. All rows, expect the last one, correspond to the Shapiro-Wilk normality test for each group; the last row corresponds to Bartlett's for equality of variances. Columns correspond to the principal components on which the t-test was applied.

Examples

```
# Create a cmpoutput object from the provided datasets
cmp <- cmpoutput("All", c(0.5, 0.6, 0.7),
                 pphpc_ok$data[["All"]], pphpc_ok$obs_lvls)

# Obtain the summary of the assumptions of the cmpoutput object
summary(assumptions(cmp))
```

---

summary.assumptions_micomp
Summary method for the assumptions of parametric tests used in multiple comparisons of outputs

---

Description

Summary method for objects of class `assumptions_micomp`, which contain the assumptions for the parametric tests used in multiple comparisons of outputs.

Usage

```
## S3 method for class 'assumptions_micomp'
summary(object, ...)
```

Arguments

`object`	Object of class `assumptions_micomp`.
`...`	Currently ignored.

Value

A list in which each component is associated with a distinct comparison. Each component contains a matrix, in which columns represent individual outputs and rows correspond to the statistical tests evaluating the assumptions of the parametric tests used in each output. More specifically, each matrix has rows with the following information:

Royston (*group*, *ve=%/npcs=*) One row per group per variance to explain / number of PCs, with the *p*-value yielded by the Royston test (`mvn`) for the respective group and variance/npcs combination.

Box's M (*ve=%/npcs=*) One row per variance to explain with the *p*-value yielded by Box's M test (`boxM`).

Shapiro-Wilk (*group*) One row per group, with the *p*-value yielded by the Shapiro-Wilk test (`shapiro.test`) for the respective group.

Bartlett One row with the *p*-value yielded by Bartlett's test (`bartlett.test`).

Examples

```
# Create a micomp object, use provided dataset
mic <- micomp(5, c(0.7, 0.8, 0.9),
              list(list(name = "NLOKvsJEXOK", grpout = pphpc_ok),
                   list(name = "NLOKvsJEXNOSHUFF", grpout = pphpc_noshuff)),
              concat = TRUE)

# Get the assumptions summary
sam <- summary(assumptions(mic))
```

summary.cmpoutput — *Summary method for comparison of an output*

Description

Summary method for objects of class `cmpoutput`.

Usage

```
## S3 method for class 'cmpoutput'
summary(object, ...)
```

Arguments

`object`	Object of class `cmpoutput`.
`...`	Currently ignored.

Value

A list with the following components:

output.name Output name.

num.pcs Number of principal components which explain `var.exp` percentage of variance.

var.exp Minimum percentage of variance which must be explained by the number of principal components used for the MANOVA test.

manova.pvals *P*-value of the MANOVA test.

parametric.test Name of the used parametric test.

parametric.pvals Vector of p-values returned by applying the parametric test to each principal component.

parametric.pvals.adjusted Vector of p-values returned by applying the parametric test to each principal component, adjusted with the weighted Bonferroni procedure, percentage of explained variance used as weight.

nonparametric.test Name of the used non-parametric test.

nonparametric.pvals Vector of p-values returned by applying the non-parametric test to each principal component.

nonparametric.pvals.adjusted Vector of p-values returned by applying the non-parametric test to each principal component, adjusted with the weighted Bonferroni procedure, percentage of explained variance used as weight.

Examples

```
# Comparing the concatenated output of the pphpc_noshuff dataset, which
# contains simulation output data from two implementations of the PPHPC model
# executed with a minor implementation difference.

summary(
  cmpoutput("All", 0.6, pphpc_noshuff$data[["All"]], pphpc_noshuff$obs_lvls)
)
```

---

summary.grpoutputs — *Summary method for grouped outputs*

Description

Summary method for objects of class `grpoutputs`.

Usage

```
## S3 method for class 'grpoutputs'
summary(object, ...)
```

Arguments

`object`	Object of class `grpoutputs`.
`...`	Currently ignored.

Value

A list with the following components:

output.dims Dimensions for each output, i.e. number of observations and number of variables (i.e. output length).

group.sizes Number of output observations in each group.

Examples

```
# Determine paths for data folders, each containing outputs for 10 runs of
# the PPHPC model
dir_nl_ok <- system.file("extdata", "nl_ok", package = "micompr")
dir_jex_noshuff <-
 system.file("extdata", "j_ex_noshuff", package = "micompr")
files <- glob2rx("stats400v1*.tsv")

# Create a grpoutputs object
go <-
 grpoutputs(c("o1", "o2"), c(dir_nl_ok, dir_jex_noshuff), c(files, files))
```

`summary.micomp` *Summary method for multiple comparisons of outputs*

Description

Summary method for objects of class `micomp`.

Usage

```
## S3 method for class 'micomp'
summary(object, ...)
```

Arguments

`object`	Object of class `micomp`.
`...`	Currently ignored.

Value

A list in which each component is associated with a distinct comparison. Each component contains a matrix, in which columns represent individual outputs and rows have information about the outputs. More specifically, each matrix has the following rows:

#PCs (ve=%) Number of principal components required to explain the specified percentage of variance. There is one row of this kind for each percentage of variance specified when creating the `micomp` object.

MANOVA (ve=%) *P*-value for the MANOVA test applied to the #PCs required to explain the specified percentage of variance. There is one row of this kind for each percentage of variance specified when creating the `micomp` object.

par.test *P*-value for the parametric test (first principal component).

nonpar.test *P*-value for the non-parametric test (first principal component).

par.test.adjust *P*-value for the parametric test (first principal component), adjusted with the weighted Bonferroni procedure, percentage of explained variance used as weight.

nonpar.test.adjust *P*-value for the non-parametric test (first principal component), adjusted with the weighted Bonferroni procedure, percentage of explained variance used as weight.

Examples

```
# A micomp object from package datasets (i.e. grpoutputs objects) directly

summary(micomp(5, 0.85,
               list(list(name = "CompEq", grpout = pphpc_ok),
                    list(name = "CompNoShuf", grpout = pphpc_noshuff),
                    list(name = "CompDiff", grpout = pphpc_diff))))
```

tikzscat *Simple* TikZ *scatter plot*

Description

Create a simple 2D TikZ scatter plot, useful for plotting PCA data.

Usage

```
tikzscat(data, obs_lvls, marks, tscale, axes_color = "gray")
```

Arguments

data	Data to plot, m x 2 numeric matrix, where m is the number of observations or points to plot.
obs_lvls	Levels or groups associated with each observation.
marks	Character vector determining how to draw the points in TikZ, for example: c("mark=square*,mark options={color=red},mark size=0.8pt","mark=diamond*,mark options={color=black},mark size=1pt","mark=triangle*,mark options={color=green},mark size=1pt").
tscale	The scale property of the TikZ figure.
axes_color	Axes color (must be a LaTeX/TikZ color).

Details

This function creates a simple TikZ 2D scatter plot within a tikzpicture environment. The points are plotted on a normalized figure with x and y axes bounded between [-1, 1]. To render adequately, the final LaTeX document should load the plotmarks TikZ library.

Value

A string containing the TikZ figure code for plotting the specified data.

Examples

```
tikzscat(rbind(c(1.5, 2), c(0.5, 1)), factor(c(1,2)),
         c("mark=square*,mark options={color=red},mark size=0.8pt",
           "mark=diamond*,mark options={color=black},mark size=1pt"),
         6)
```

toLatex.cmpoutput — *Convert* cmpoutput *object to* LaTeX *table*

Description

This method converts cmpoutput objects to character vectors representing LaTeX tables.

Usage

```
## S3 method for class 'cmpoutput'
toLatex(object, cmp_name = "Comp. 1", ...)
```

Arguments

object	A cmpoutput object.
cmp_name	Comparison name (to appear in table).
...	Any options accepted by the toLatex.micomp function.

Details

This method simply wraps the cmpoutput object into a micomp object, and invokes toLatex.micomp on the wrapped object.

Value

A character vector where each element holds one line of the corresponding LaTeX table.

Examples

```
# Create a cmpoutput object by comparing the first output ("Pop.Sheep") of
# one the provided datasets.
cmp <-
 cmpoutput("SheepPop", 0.9, pphpc_ok$data[["Pop.Sheep"]], pphpc_ok$obs_lvls)

# Print latex table source to screen
toLatex(cmp)
```

toLatex.micomp *Convert* micomp *object to* LaTeX *table*

Description

This method converts micomp objects to character vectors representing LaTeX tables.

Usage

```
## S3 method for class 'micomp'
toLatex(
  object,
  ...,
  orientation = T,
  data_show = c("npcs-1", "mnvp-1", "parp-1", "nparp-1", "scoreplot"),
  data_labels = NULL,
  labels_cmp_show = T,
  labels_col_show = T,
  label_row_show = T,
  tag_comp = "Comp.",
  tag_data = "Data",
  tag_outputs = "Outputs",
  table_placement = "ht",
  latex_envs = c("center"),
  booktabs = F,
  booktabs_cmalign = "l",
  caption = NULL,
  caption_cmd = "\\caption",
  label = NULL,
  col_width = F,
  pvalf_f = pvalf.default,
  pvalf_params = list(),
  scoreplot_marks = c("mark=square*,mark options={color=red},mark size=0.8pt",
    "mark=diamond*,mark options={color=black},mark size=1pt",
    "mark=triangle*,mark options={color=green},mark size=1pt"),
  scoreplot_scale = 6,
  scoreplot_before = "\\raisebox{-.5\\height}{\\resizebox {1.2cm} {1.2cm} {",
  scoreplot_after = "}}"
)
```

Arguments

object	A micomp object.
...	Currently ignored.
orientation	If TRUE, outputs are placed along columns, while data is placed along rows. If FALSE, outputs are placed along rows, while data is placed along columns.

data_show Vector of strings specifying what data to show. Available options are:

npcs-i Number of principal components required to explain i-th user-specified percentage of variance.

mnvp-i MANOVA *p*-values for the i-th user-specified percentage of variance to explain.

parp-j Parametric test *p*-values for the j-th principal component.

nparp-j Non-parametric test *p*-values for the j-th principal component.

aparp-j Parametric test *p*-values adjusted with weighted Bonferroni procedure for the j-th principal component.

anparp-j Non-parametric test *p*-values adjusted with weighted Bonferroni procedure for the j-th principal component.

varexp-j Explained variance for the j-th principal component.

scoreplot Output projection on the first two principal components.

sep Place a separator (e.g. midrule) between data.

data_labels Vector of strings specifying the labels of the data to show. If NULL, default labels are used for all elements. If individual elements are set to NA, default labels will be used for those elements.

labels_cmp_show Show the column containing the comparison labels?

labels_col_show Show the column containing the data labels (orientation == T) or the output labels (orientation == F)?

label_row_show Show the tag_outputs tag? If TRUE, the row identifier part will have two levels, the tag_outputs label and output names (orientation == T), or the tag_data and data labels (orientation == F). If FALSE only the output names or data labels are shown.

tag_comp Tag identifying comparison labels.

tag_data Tag identifying data labels.

tag_outputs Tag identifying outputs.

table_placement LaTeX table placement.

latex_envs Wrap table in the specified LaTeX environments.

booktabs Use booktabs table beautifier package?

booktabs_cmalign How to align cmidule when using the booktabs package.

caption Table caption.

caption_cmd Command used for table caption.

label Table label for cross-referencing.

col_width Resize table to page column width?

pvalf_f *P*-value formatter function, which receives a numeric value between 0 and 1 and returns a string containing the formatted value. Default is pvalf.default (requires ulem LaTeX package).

pvalf_params — Parameters for pvalf_f function. Default is empty list.

scoreplot_marks — Vector of strings specifying how TikZ should draw points belonging to each group in the score plot.

scoreplot_scale — TikZ scale for each score plot figure.

scoreplot_before — LaTeX code to paste before each TikZ score plot figure.

scoreplot_after — LaTeX code to paste after each TikZ score plot figure.

Details

This method is inspired by the functionality provided by the xtable and print.xtable functions (from the xtable package), but follows the standard behavior of the toLatex generic.

Value

A character vector where each element holds one line of the corresponding LaTeX table.

Examples

```
# Create a micomp object, use provided dataset, three first outputs, plus
# a fourth concatenated output
mic <- micomp(4, 0.8,
              list(list(name = "NLOKvsJEXOK", grpout = pphpc_ok),
                   list(name = "NLOKvsJEXNOSHUFF", grpout = pphpc_noshuff),
                   list(name = "NLOKvsJEXDIFF", grpout = pphpc_diff)),
              concat = TRUE)

# Print latex table source to screen
toLatex(mic)
```

Index