Package 'spectrakit'

April 6, 2026

Type Package

Title Spectral Data Handling and Visualization

Version 0.2.0

Description Provides functions to combine, normalize and visualize spectral data, perform principal component analysis (PCA), and assemble customizable image grids suitable for publication-quality scientific figures.

License MIT + file LICENSE

URL https://rpackagelab.blogspot.com/2026/04/introducing-spectrakit-r-package.html

Encoding UTF-8

Imports dplyr, readr, ggplot2, ggrepel, tibble, purrr, rlang, magick, glue, data.table, scales, stats

Suggests RColorBrewer

RoxygenNote 7.3.3

NeedsCompilation no

Author Gianluca Pastorelli [aut, cre] (ORCID: <https://orcid.org/0000-0001-6926-1952>)

Maintainer Gianluca Pastorelli <gianluca.pastorelli@gmail.com>

Depends R (>= 4.1.0)

Repository CRAN

Date/Publication 2026-04-06 13:50:02 UTC

Contents

`combineSpectra` *Combine and Normalize Spectral Data from Multiple Files*

Description

Reads spectral data from multiple files in a folder, merges them by a common column (e.g., wavelength), optionally filters by a range, applies normalization and returns the data in either row-wise or column-wise format.

Usage

```
combineSpectra(
  folder = ".",
  file_type = "csv",
  sep = ",",
  header = TRUE,
  common_col_pos = 1,
  data_col_pos = 2,
  range = NULL,
  normalization = c("none", "simple", "min-max", "z-score", "area", "vector"),
  norm_scope = c("global", "range"),
  orientation = c("columns", "rows")
)
```

Arguments

`folder`	Character. Path to the folder containing spectra files. Default is ‘"."‘ (working directory).
`file_type`	Character. File extension (without dot) to search for. Default is ‘"csv"‘.
`sep`	Character. Delimiter for file columns. Use ‘","‘ for comma-separated (default) or ‘"\t"‘ for tab-delimited files.
`header`	Logical. Whether the files contain a header row. Default is ‘TRUE‘.
`common_col_pos`	Integer. Column position for the common variable (e.g., wavelength). Default is ‘1‘.
`data_col_pos`	Integer. Column position for the spectral intensity values. Default is ‘2‘.
`range`	Numeric vector of length 2. Range filter for the common column (e.g., wavelength limits). Default is ‘NULL‘ (no filtering).
`normalization`	Character. Normalization method to apply to intensity values. One of:
	‘"none"‘ No normalization is applied (default).
	‘"simple"‘ Divide by the maximum intensity.
	‘"min-max"‘ Scale intensities to the [0,1] range.
	‘"z-score"‘ Subtract the mean and divide by the standard deviation of intensities.
	‘"area"‘ Divide by the total sum of intensities so the spectrum area = 1.

‘"vector"‘ Normalize the spectrum as a unit vector by dividing by the square root of the sum of squared intensities; also known as L2 normalization.

`norm_scope` Character. Determines whether normalization is applied to the full spectrum or only to a specified range. One of: ‘"global"‘ (full spectrum), or ‘"range"‘ (specified range). Default is ‘"global"‘.

`orientation` Character. Output organization. One of: ‘"columns"‘, to keep each spectrum as a column, or ‘"rows"‘, to transpose so each spectrum is a row. Default is ‘"columns"‘.

Value

A ‘tibble‘ that can be exported as, for example, a CSV file. Each spectrum is either a column (default) or row, depending on ‘orientation‘; a sum or mean spectrum can optionally be computed using ‘rowSums()‘ or ‘colSums()‘, or ‘rowMeans()‘ or ‘colMeans()‘, respectively. The common column (e.g., wavelength) is retained.

Examples

```
# Create a temporary directory for mock CSV files
tmp_dir <- tempdir()

# Define file paths
tmp1 <- file.path(tmp_dir, "file1.csv")
tmp2 <- file.path(tmp_dir, "file2.csv")

# Write two mock CSV files in the temporary folder
write.csv(data.frame(ID = c("A", "B", "C"), val = c(1, 2, 3)), tmp1, row.names = FALSE)
write.csv(data.frame(ID = c("A", "B", "C"), val = c(4, 5, 6)), tmp2, row.names = FALSE)

# Merge the CSV files in the temporary folder, normalize with z-score, and return transposed
result <- combineSpectra(
  folder = tmp_dir,
  file_type = "csv",
  sep = ",",
  common_col_pos = 1,
  data_col_pos = 2,
  normalization = "z-score",
  orientation = "rows"
)
```

`makeComposite` *Create a labeled image grid*

Description

Generates a composite image grid with customizable layout, labels and resizing options. Suitable for spectra and other image types.

Usage

```
makeComposite(
  folder = ".",
  custom_order,
  rows,
  cols,
  spacing = 15,
  resize_mode = c("none", "fit", "fill", "width", "height", "both"),
  labels = list(),
  label_settings = list(),
  background_color = "white",
  desired_width = 15,
  width_unit = "cm",
  ppi = 300,
  output_format = "tiff",
  output_folder = NULL
)
```

Arguments

folder	Character. Path to the folder containing images. Default is ‘"."‘ (working directory).
custom_order	Character vector. Ordered set of filenames (use NA for blank slots). Argument is required.
rows	Integer. Number of rows in the grid. Argument is required.
cols	Integer. Number of columns in the grid. Argument is required.
spacing	Integer. Spacing (in pixels) between tiles. Default is ‘15‘
resize_mode	Character. Method to resize panels in the composite. One of: **‘"none"‘** Keeps each panel at its original size (default). **‘"fit"‘** Scales each panel to fit within the smallest width and height among all images, preserving aspect ratio; no cropping occurs, empty space may remain. **‘"fill"‘** Scales each panel to completely cover the smallest width and height among all images, preserving aspect ratio, then crops any excess. **‘"width"‘** Resizes each panel to match the minimum width among all images, preserving aspect ratio; height scales accordingly. **‘"height"‘** Resizes each panel to match the minimum height among all images, preserving aspect ratio; width scales accordingly. **‘"both"‘** Resizes each panel to exactly match the minimum width and height among all images, without preserving aspect ratio; may cause distortion.
labels	List of up to 4 character vectors. Labels to apply to each panel. Each vector corresponds to one label layer and must be the same length as the number of non-NA images. Use empty strings "" or NULL entries to omit specific labels. Default is ‘list()‘ (no labels).
label_settings	List of named lists. Each named list specifies styling options for a label layer. Options include:

'size' Font size (e.g., '100').

'color' Font color (e.g., '"black"'.

'font' Font family (e.g., '"Arial"').

'boxcolor' Background color behind text (e.g., '"white"'), or 'NA' for none.

'location' Offset from the gravity anchor (e.g., '"+10+10"').

'gravity' Placement anchor for the label (e.g., '"northwest"').

'weight' Font weight (e.g., '400' = normal, '700' = bold).

Default is 'list()' (default styling is used).

`background_color`
: Character. Background color used for blank tiles and borders. Use '"none"' for transparency. Default is '"white"'.

`desired_width`
: Numeric. Desired width of final image (in centimeters, inches or pixels). Default is '15'

`width_unit`
: Character. One of: '"cm"', '"in"', or '"px"'. Default is '"cm"'

`ppi`
: Numeric. Resolution (pixels per inch) for output file. Default is '300'

`output_format`
: Character. File format for saving image. Examples: '"tiff"', '"png"', '"pdf"'. Default is '"tiff"'.

`output_folder`
: Character. Path to folder where the composite image is saved. If NULL (default), the image is not saved and a 'magick image' object is returned; If specified, the image is saved automatically; if '"."', the image is saved in the working directory.

Value

If 'output_folder = NULL', the function returns a 'magick image' object. When 'output_folder' is specified, the composite image is written directly to disk and returned invisibly.

Examples

```
library(magick)

tmp_dir <- file.path(tempdir(), "spectrakit_imgs")
dir.create(tmp_dir, showWarnings = FALSE)

# Create and save img1
img1 <- image_blank(100, 100, "white")
img1 <- image_draw(img1)
symbols(50, 50, circles = 30, inches = FALSE, add = TRUE, bg = "red")
dev.off()
img1_path <- file.path(tmp_dir, "img1.png")
image_write(img1, img1_path)

# Create and save img2
img2 <- image_blank(100, 100, "white")
img2 <- image_draw(img2)
rect(20, 20, 80, 80, col = "blue", border = NA)
dev.off()
img2_path <- file.path(tmp_dir, "img2.png")
```

```
image_write(img2, img2_path)

# Create composite
makeComposite(
        folder = tmp_dir,
        custom_order = c("img1.png", "img2.png"),
        rows = 1,
        cols = 2,
        labels = list(c("Red Circle", "Blue Rectangle")),
        label_settings = list(
                list(size = 5, font = "Arial", color = "black", boxcolor = "white",
                     gravity = "northwest", location = "+10+10", weight = 400)
        ),
        resize_mode = "none",
        desired_width = 10,
        width_unit = "cm",
        ppi = 300,
        output_format = "png",
        output_folder = tmp_dir
)
```

---

plotPCA *Perform PCA and Create a Plot of Scores, Loadings, or Biplot for Selected Principal Components*

Description

Computes principal component analysis (PCA) on numeric variables in a dataset and generates a plot of two selected principal components (scores, loadings, or biplot).

Usage

```
plotPCA(
  data,
  pcs = c(1, 2),
  color_var = NULL,
  shape_var = NULL,
  plot_type = c("score", "loading", "biplot", "cumvar"),
  palette = "Dark2",
  score_labels = TRUE,
  loading_labels = TRUE,
  ellipses = FALSE,
  ellipse_var = NULL,
  display_names = FALSE,
  legend_title = NULL,
  return_pca = FALSE,
  output_format = "tiff",
  output_folder = NULL
)
```

Arguments

`data`	Data frame containing numeric variables. Numeric variables are centered and scaled by default; non-numeric columns are ignored. Argument is required.
`pcs`	Numeric vector of length 2. Indicates which principal components to plot. Default is c(1, 2).
`color_var`	Character. Column name for coloring points by group; converted to factor internally. Default is 'NULL' (all points same color).
`shape_var`	Character. Column name for shaping points by group; converted to factor internally. Default is 'NULL' (all points same shape).
`plot_type`	Character. Type of PCA plot to generate. One of: **'"score"'** Plot PCA scores, i.e., observations (default). **'"loading"'** Plot PCA loadings, i.e., variables. **'"biplot"'** Combine scores and loadings in a biplot. **'"cumvar"'** Plot cumulative variance explained across principal components.
`palette`	Character or vector. Color setting for groups. One of: • A single color repeated for all groups. • A ColorBrewer palette name (default is '"Dark2"'; requires the package to be installed). • A custom color vector, recycled to match the number of groups.
`score_labels`	Logical or character. Controls labeling of points (scores). One of: **TRUE** Uses row names for labels (default). **column name** Uses a column in the data frame for labels. **FALSE** No labels are shown.
`loading_labels`	Logical. If 'TRUE', displays labels for variables (loadings). Default is TRUE.
`ellipses`	Logical. If 'TRUE', draws confidence ellipses around groups in score/biplot. Grouping logic for ellipses follows this priority: **ellipse_var** If provided, ellipses are drawn by that variable. **color_var** If `ellipse_var` is not provided, ellipses are drawn by color groups. **shape_var** If neither `ellipse_var` nor `color_var` is provided, ellipses are drawn by shape groups. **none** If none are provided, no ellipses are drawn. Default is FALSE.
`ellipse_var`	Character. Name of the variable used to group ellipses. Takes precedence over all other grouping variables; converted to factor internally. Default is NULL.
`display_names`	Logical. Shows legend if TRUE. Default is FALSE.
`legend_title`	Character. Legend title corresponding to 'color_var' and/or 'shape_var'. Default is NULL.
`return_pca`	Logical. If TRUE, return a list with plot and PCA object. Default is FALSE.
`output_format`	Character. File format for saving plots. Examples: '"tiff"', '"png"', '"pdf"'. Default is '"tiff"'.
`output_folder`	Character. Path to folder where plots are saved. If NULL (default), plot is not saved and the ggplot object (or list with plot and PCA if 'return_pca = TRUE') is returned. If specified, plot is saved automatically (function returns PCA object only if 'return_pca = TRUE'); if '"."', plot is saved in the working directory.

Value

A ggplot2 object representing the PCA plot, or a list with ‘plot‘ and ‘pca‘ if ‘return_pca = TRUE‘.

Examples

```
plotPCA(
  data = iris,
  color_var = "Species",
  shape_var = "Species",
  plot_type = "biplot",
  palette = "Dark2",
  ellipses = FALSE,
  display_names = TRUE,
  legend_title = "Iris Species"
)
```

---

plotSpectra *Plot Spectral Data from Multiple Files*

Description

Reads, normalizes and plots spectral data from files in a folder. Supports multiple plot modes, color palettes, axis customization, annotations and automatic saving of plots to files.

Usage

```
plotSpectra(
  folder = ".",
  file_type = "csv",
  sep = ",",
  header = TRUE,
  normalization = c("none", "simple", "min-max", "z-score", "area", "vector"),
  norm_scope = c("global", "range"),
  x_config = NULL,
  x_reverse = FALSE,
  y_trans = c("linear", "log10", "sqrt"),
  x_label = NULL,
  y_label = NULL,
  line_size = 0.5,
  palette = "black",
  plot_mode = c("individual", "overlapped", "stacked"),
  display_names = FALSE,
  vertical_lines = NULL,
  shaded_ROIs = NULL,
  annotations = NULL,
  output_format = "tiff",
  output_folder = NULL
)
```

Arguments

folder	Character. Path to the folder containing spectra files. Default is ‘"."‘ (working directory).
file_type	Character. File extension (without dot) to search for. Default is ‘"csv"‘.
sep	Character. Delimiter for file columns. Use ‘","‘ for comma-separated (default) or ‘"\t"‘ for tab-delimited files.
header	Logical. Whether the files contain a header row. Default is ‘TRUE‘.
normalization	Character. Normalization method to apply to y-axis data. One of: **‘"none"‘** No normalization is applied (default). **‘"simple"‘** Divide by the maximum intensity. **‘"min-max"‘** Scale intensities to the [0,1] range. **‘"z-score"‘** Subtract the mean and divide by the standard deviation of intensities. **‘"area"‘** Divide by the total sum of intensities so the spectrum area = 1. **‘"vector"‘** Normalize the spectrum as a unit vector by dividing by the square root of the sum of squared intensities; also known as L2 normalization.
norm_scope	Character. Determines whether normalization is applied to the full spectrum or only to a specified range. One of: ‘"global"‘ (full spectrum), or ‘"range"‘ (specified range). Default is ‘"global"‘.
x_config	Numeric vector of length 3. Specifies x-axis range and breaks: ‘c(min, max, step)‘. If ‘NULL‘ (default), the full data range is used and axis breaks are set every 100 units.
x_reverse	Logical. If ‘TRUE‘, reverses the x-axis. Default is ‘FALSE‘.
y_trans	Character. Transformation for the y-axis. One of: ‘"linear"‘, ‘"log10"‘, or ‘"sqrt"‘. Default is ‘"linear"‘.
x_label	Character or expression. Label for the x-axis. Supports mathematical notation via ‘expression()‘, e.g., `expression(Wavenumber~(cm^{-1}))`. Default is ‘NULL‘ (no axis label).
y_label	Character or expression. Label for the y-axis. Supports mathematical notation via ‘expression()‘, e.g., `expression(Delta*E["00"]^{"*"})`. Default is ‘NULL‘ (no axis label).
line_size	Numeric. Width of the spectral lines. Default is ‘0.5‘.
palette	Character or vector. Color setting. One of: a single color (e.g., ‘"black"‘), a ColorBrewer palette name (e.g., ‘"Dark2"‘; requires the package to be installed), or a custom color vector. Default is ‘"black"‘.
plot_mode	Character. Plotting style. One of ‘"individual"‘ (one plot per spectrum), ‘"overlapped"‘ (all in one), or ‘"stacked"‘ (vertically offset). Default is ‘"individual"‘.
display_names	Logical. If ‘TRUE‘, adds file names as titles to individual spectra or a legend to combined spectra. Default is ‘FALSE‘.
vertical_lines	Numeric vector. Adds vertical dashed lines at given x positions. Default is ‘NULL‘.

shaded_ROIs	List of numeric vectors. Each vector must have two elements (‘xmin‘, ‘xmax‘) to define shaded x regions. Default is ‘NULL‘.
annotations	Data frame with columns ‘file‘ (file name without extension), ‘x‘, ‘y‘, and ‘label‘. Adds annotation labels to specific points in spectra. Default is ‘NULL‘.
output_format	Character. File format for saving plots. Examples: ‘"tiff"‘, ‘"png"‘, ‘"pdf"‘. Default is ‘"tiff"‘.
output_folder	Character. Path to folder where plots are saved. If NULL (default), plots are not saved and the ggplot object is returned. If specified, plots are saved automatically; if ‘"."‘, plots are saved in the working directory.

Details

Color settings can support color-blind-friendly palettes from ‘RColorBrewer‘. Use ‘display.brewer.all(colorblindFriendly = TRUE)‘ to preview.

Value

If ‘output_folder = NULL‘, the function returns a ‘ggplot‘ object. Otherwise, returns ‘NULL‘ invisibly after saving the plots to a specified output folder.

Examples

```
# Create a temporary directory and write mock spectra files
tmp_dir <- tempdir()
write.csv(data.frame(Energy = 0:30, Counts = rpois(31, lambda = 100)),
          file.path(tmp_dir, "spec1.csv"), row.names = FALSE)
write.csv(data.frame(Energy = 0:30, Counts = rpois(31, lambda = 120)),
          file.path(tmp_dir, "spec2.csv"), row.names = FALSE)

# Plot the mock spectra using various configuration options
plotSpectra(
  folder = tmp_dir,
  file_type = "csv",
  sep = ",",
  normalization = "min-max",
  x_config = c(0, 30, 5),
  x_reverse = FALSE,
  y_trans = "linear",
  x_label = expression(Energy~(keV)),
  y_label = expression(Counts/1000~s),
  line_size = 0.7,
  palette = c("black","red"),
  plot_mode = "overlapped",
  display_names = TRUE,
  vertical_lines = c(10, 20),
  shaded_ROIs = list(c(12, 14), c(18, 22)),
  output_format = "png",
  output_folder = NULL
)
```

Index