Package ‘NOVA’

October 16, 2025

Type Package

Title Neural Output Visualization and Analysis

Version 0.1.1

Description A comprehensive toolkit for analyzing and visualizing neural data outputs, including Principal Component Analysis (PCA) trajectory plotting, Multi-Electrode Array (MEA) heatmap generation, and variable importance analysis. Provides publication-ready visualizations with flexible customization options for neuroscience research applications.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 3.6.0)

Imports readr, tibble, DT, knitr, scales, readxl, writexl, dplyr, ggplot2, tidyr (>= 1.1.0), purrr (>= 0.3.0), rlang (>= 0.4.0), stringr (>= 1.4.0), RColorBrewer (>= 1.1.0), viridis (>= 0.5.0), pheatmap (>= 1.0.0), gridExtra (>= 2.3.0)

Suggests utils, testthat (>= 3.0.0), rmarkdown (>= 2.0)

URL https://github.com/atudoras/NOVA

BugReports https://github.com/atudoras/NOVA/issues

NeedsCompilation no

Author Alex Tudoras [aut, cre]

Maintainer Alex Tudoras <alex.tudorasmiravet@ucsf.edu>

Repository CRAN

Date/Publication 2025-10-16 18:00:07 UTC

Contents

aggregate_data *Aggregate Data by Groups*

Description

Aggregates values within groups using specified method

Usage

```
aggregate_data(data, group_col, variable_column, value_column, method)
```

Arguments

`data`	Data frame to aggregate
`group_col`	Column name for grouping
`variable_column`	Column name containing variable identifiers
`value_column`	Column name containing values to aggregate
`method`	Aggregation method: "mean", "median", "sum"

Value

Aggregated data frame

Examples

```
test_data <- data.frame(
  Group = rep(c("A", "B"), each = 10),
  Variable = rep(paste0("V", 1:5), 4),
  Value = rnorm(20)
)
agg <- aggregate_data(test_data, "Group", "Variable", "Value", "mean")
```

`analyze_pca_variable_importance_general`

Analyze and Visualize PCA Variable Importance

Description

This function performs comprehensive analysis of variable importance in Principal Component Analysis, generating multiple visualization types including loading biplots, importance rankings, PC comparisons, and heatmaps. It extracts variable contributions to specified principal components and creates publication-ready plots with detailed statistical summaries.

Usage

```
analyze_pca_variable_importance_general(
  pca_result = NULL,
  output_dir = tempdir(),
  experiment_name = "PCA_Analysis",
  pc_x = "PC1",
  pc_y = "PC2",
  color_scheme = "default",
  top_n = 15,
  min_loading_threshold = 0.1,
  save_plots = TRUE,
  show_labels = TRUE,
  verbose = TRUE
)
```

Arguments

`pca_result` A PCA result object. Can be either a `prcomp` object directly, or a list containing a PCA object in fields named 'pca_result', 'pca', 'result', or 'prcomp'.

`output_dir` Character string specifying the directory for saving plots and results (default: "pca_plots").

`experiment_name` Character string used as a prefix for output files and plot titles (default: "PCA_Analysis").

`pc_x` Character string specifying the principal component for x-axis analysis (default: "PC1").

`pc_y` Character string specifying the principal component for y-axis analysis (default: "PC2").

`color_scheme` Character string specifying the color palette. Options: "default", "viridis", "colorbrewer" (default: "default").

`top_n` Numeric value specifying the number of top variables to focus on in detailed analyses (default: 15).

`min_loading_threshold` Numeric value specifying the minimum loading threshold for importance filtering (default: 0.1).

`save_plots` Logical indicating whether to save plots and results to disk (default: TRUE).

`show_labels` Logical indicating whether to show variable labels on the biplot (default: TRUE).

`verbose` Logical indicating whether to print detailed progress messages (default: TRUE).

Details

The function calculates multiple importance metrics for each variable:

- **PC loadings**: Direct loading values for specified principal components
- **Combined importance**: Euclidean distance combining both PC loadings
- **Contribution percentages**: Percent contribution to each PC's total variance
- **Ranking**: Variables ranked by combined importance score

Four visualization types are generated:

- **Loading Biplot**: Scatter plot showing variable loadings on both PCs with size indicating importance
- **Importance Bar Chart**: Ranked bar chart of top variables by combined importance
- **PC Comparison**: Side-by-side comparison of absolute loadings for both PCs
- **Loading Heatmap**: Color-coded matrix showing loading values and directions

The function automatically:

- Validates input PCA objects from various sources
- Calculates variance explained by each principal component
- Creates publication-ready plots with consistent theming
- Exports detailed CSV files with variable rankings and analysis summaries
- Provides comprehensive statistical summaries

Color schemes provide different aesthetic options:

- `default`: Blue/red palette suitable for most publications
- `viridis`: Colorblind-friendly viridis color scale
- `colorbrewer`: ColorBrewer palettes optimized for scientific visualization

View top variables using head(results$selected_variables)

Value

A list containing:

plots Named list of ggplot objects: 'biplot', 'importance_bar', 'pc_comparison', 'heatmap'

variable_importance Data frame with comprehensive variable importance metrics for all variables

selected_variables Data frame containing the top N most important variables with detailed statistics

analysis_summary List with key analysis metrics and variance explained information

config_used List documenting all parameters used in the analysis

Output Files

When `save_plots = TRUE`, the function creates files in the specified output directory (default: "pca_plots"). For CRAN compliance, use `tempdir()` for the output directory:

- PNG files for each visualization type
- CSV file with complete variable importance rankings
- CSV file with selected top variables and detailed metrics
- CSV file with analysis summary and metadata

See Also

`prcomp` for PCA computation, `biplot` for basic PCA plotting

`apply_scaling_enhanced`

Apply Enhanced Scaling Methods

Description

Applies various scaling methods to matrix data for heatmap visualization

Usage

```
apply_scaling_enhanced(matrix_data, scale_method, verbose = FALSE)
```

Arguments

`matrix_data`	Numeric matrix to scale
`scale_method`	Scaling method: "variable_0_10", "robust", "row", "column", "none"
`verbose`	Whether to print scaling information

Value

Scaled matrix

clean_heatmap_matrix *Clean Heatmap Matrix*

Description

Removes rows and columns with insufficient finite values from matrix

Usage

```
clean_heatmap_matrix(matrix_data, min_finite = 2, verbose = FALSE)
```

Arguments

matrix_data	Numeric matrix to clean
min_finite	Minimum number of finite values required per row/column
verbose	Whether to print cleaning information

Value

Cleaned matrix or NULL if insufficient data

create_annotations_enhanced *Create Enhanced Annotations for Heatmaps*

Description

Creates annotation data frames and color schemes for heatmap visualization

Usage

```
create_annotations_enhanced(rownames_vector, factor_cols)
```

Arguments

rownames_vector	Vector of combined row names to parse
factor_cols	Vector of factor column names

Value

List containing annotations data frame and color schemes

create_color_palette_enhanced

Create Enhanced Color Palettes

Description

Creates color palettes and breaks for heatmap visualization

Usage

```
create_color_palette_enhanced(
  palette_name = "yellow_purple",
  custom_colors = NULL,
  data_matrix = NULL
)
```

Arguments

`palette_name`	Name of color palette to use
`custom_colors`	Vector of custom colors (optional)
`data_matrix`	Data matrix to determine color range

Value

List containing colors and breaks

create_mea_heatmaps_enhanced

Create Enhanced Heatmaps for Multi-Electrode Array (MEA) Data Analysis

Description

This function generates comprehensive heatmap visualizations for MEA data analysis, including individual grouping variable heatmaps, combined interaction heatmaps, and variable correlation matrices. It provides flexible scaling, clustering, and customization options with automatic quality filtering and missing data handling.

Usage

```
create_mea_heatmaps_enhanced(
  data = NULL,
  processing_result = NULL,
  config = NULL,
  value_column = "Normalized_Value",
  variable_column = "Variable",
  grouping_columns = c("Treatment", "Genotype"),
  sample_id_columns = c("Well"),
  timepoint_column = "Timepoint",
  scale_method = "z_score",
  aggregation_method = "mean",
  missing_value_handling = "remove",
  cluster_method = "euclidean",
  cluster_rows = TRUE,
  cluster_cols = TRUE,
  create_individual_heatmaps = TRUE,
  create_combined_heatmap = TRUE,
  create_variable_correlation = TRUE,
  output_dir = NULL,
  save_plots = FALSE,
  plot_format = "png",
  plot_width = 10,
  plot_height = 8,
  dpi = 300,
  fontsize = 10,
  angle_col = 45,
  show_rownames = TRUE,
  show_colnames = TRUE,
  return_data = TRUE,
  verbose = TRUE,
  quality_threshold = 0.8,
  min_observations = 3
)
```

Arguments

`data` A data frame containing MEA measurement data. If NULL, must provide processing_result.

`processing_result`
A list object from MEA data processing containing normalized_data or raw_data components. Takes precedence over the data parameter if provided.

`config` Configuration list from MEA processing. If NULL and processing_result is provided, will attempt to use config from processing_result$config_used.

`value_column` Character string specifying the column containing measurement values (default: "Normalized_Value").

`variable_column`
Character string specifying the column containing variable names (default: "Variable").

`grouping_columns`
Character vector of column names to use for grouping (default: c("Treatment", "Genotype")). Function will auto-detect which columns are available.

`sample_id_columns`
Character vector of columns identifying individual samples (default: c("Well")).

`timepoint_column`
Character string specifying the timepoint column (default: "Timepoint").

`scale_method` Character string specifying scaling method. Options: "z_score" (default), "min_max", "robust", "none".

`aggregation_method`
Character string specifying how to aggregate multiple measurements. Options: "mean" (default), "median", "sum".

`missing_value_handling`
Character string specifying how to handle missing values. Options: "remove" (default), "impute_mean", "impute_zero".

`cluster_method` Character string specifying clustering distance method. Options: "euclidean" (default), "correlation", "manhattan".

`cluster_rows` Logical indicating whether to cluster rows (default: TRUE).

`cluster_cols` Logical indicating whether to cluster columns (default: TRUE).

`create_individual_heatmaps`
Logical indicating whether to create separate heatmaps for each grouping variable (default: TRUE).

`create_combined_heatmap`
Logical indicating whether to create interaction heatmap when multiple grouping variables are present (default: TRUE).

`create_variable_correlation`
Logical indicating whether to create variable correlation heatmap (default: TRUE).

`output_dir` Character string specifying output directory (default: NULL, no files saved)

`save_plots` Logical indicating whether to save plots to disk (default: FALSE)

`plot_format` Character string specifying file format for saved plots (default: "png").

`plot_width` Numeric value specifying plot width in inches (default: 10).

`plot_height` Numeric value specifying plot height in inches (default: 8).

`dpi` Numeric value specifying resolution for saved plots (default: 300).

`fontsize` Numeric value specifying font size for heatmap labels (default: 10).

`angle_col` Numeric value specifying angle for column labels in degrees (default: 45).

`show_rownames` Logical indicating whether to show row names (default: TRUE).

`show_colnames` Logical indicating whether to show column names (default: TRUE).

`return_data` Logical indicating whether to return processed data matrices (default: TRUE).

`verbose` Logical indicating whether to print progress messages (default: TRUE).

`quality_threshold`
: Numeric value between 0-1 specifying minimum data completeness per variable (default: 0.8).

`min_observations`
: Numeric value specifying minimum observations required per group (default: 3).

Details

The function performs several key operations:

- Quality filtering: Removes variables with insufficient data completeness
- Missing value handling: Multiple strategies for dealing with NA values
- Data aggregation: Combines multiple measurements per group using specified method
- Scaling: Applies normalization methods appropriate for heatmap visualization
- Clustering: Hierarchical clustering of rows and/or columns using specified distance metrics
- Visualization: Creates publication-ready heatmaps with proper color schemes and annotations

For scaling methods:

- z_score: Centers data around mean with unit variance (best for comparing relative changes)
- min_max: Scales to 0-1 range (best for absolute comparisons)
- robust: Uses median and MAD for outlier-resistant scaling
- none: No scaling applied

The function automatically adjusts plot dimensions based on data size and uses optimized color palettes appropriate for the scaling method chosen (diverging palettes for z_score/robust, sequential palettes for min_max).

Value

A list containing:

individual_heatmaps Named list of heatmap objects for each grouping variable

combined_heatmap Heatmap object for grouping variable interactions (if applicable)

variable_correlation List with correlation heatmap and correlation matrix

metadata List containing processing information and parameters used

Each heatmap object contains: heatmap (pheatmap object), scaled_data (processed matrix), raw_data (aggregated input data), annotation (row annotations), annotation_colors (color schemes), and scaling_info (scaling parameters).

`discover_mea_structure`

Discover MEA Data Structure

Description

This function scans a directory containing MEA (Multi-Electrode Array) experiment folders and analyzes the structure of CSV files to identify experiments, timepoints, measured variables, treatments, and genotypes. It provides a comprehensive overview of the data organization without loading all files into memory.

Usage

```
discover_mea_structure(
  main_dir,
  experiment_pattern = "MEA\\d+",
  file_pattern = "\\.csv$",
  verbose = TRUE
)
```

Arguments

`main_dir`	Character. Path to the main directory containing experiment folders
`experiment_pattern`	Character. Regex pattern to identify experiment directories (default: "MEA\d+")
`file_pattern`	Character. Regex pattern to identify data files (default: "\.csv$")
`verbose`	Logical. Whether to print progress messages (default: TRUE)

Details

The function expects MEA CSV files with standard format: - Row 121: Well identifiers (A1, A2, B1, etc.) - Row 122: Treatment conditions - Row 123: Genotype information - Row 124: Exclusion flags - Rows 125-168: Variable names and measurements

Discover structure of MEA data (requires data directory)

Value

A list containing: - experiments: List of experiment info (directories, files, timepoints, metadata) - all_timepoints: Vector of all unique timepoints found across experiments - all_variables: Vector of all unique measured variables - potential_baselines: Timepoints that might serve as baseline conditions - experiment_count: Total number of experiments found - discovery_timestamp: When the analysis was performed

`handle_missing_values` *Handle Missing Values in MEA Data*

Description

Handles missing values in MEA datasets using various imputation strategies or removal methods.

Usage

```
handle_missing_values(data, value_column, method, verbose)
```

Arguments

data	Data frame containing MEA data
value_column	Character string specifying the column with values to process
method	Character string specifying handling method: "remove", "impute_mean", "impute_zero"
verbose	Logical indicating whether to print progress messages

Value

Data frame with missing values handled according to specified method

Examples

```
test_data <- data.frame(
  ID = 1:10,
  Value = c(1.2, NA, 3.4, 2.1, NA, 5.6, 4.3, NA, 2.8, 3.9)
)
cleaned <- handle_missing_values(test_data, "Value", "remove", FALSE)
```

`null_coalesce` *Null Coalescing Operator*

Description

Returns the left-hand side if not NULL, otherwise the right-hand side

Usage

```
null_coalesce(lhs, rhs)
```

Arguments

`lhs`	Left-hand side value
`rhs`	Right-hand side value (default/fallback)

Value

lhs if not NULL, otherwise rhs

Examples

```
null_coalesce(5, 10)
null_coalesce(NULL, 10)
```

`pca_analysis_enhanced` *Enhanced PCA Analysis for MEA Data*

Description

This function performs Principal Component Analysis (PCA) on MEA data with extensive flexibility for data input sources, parameter configuration, and output options. It handles missing values, applies variance filtering, creates visualization plots, and provides comprehensive results suitable for downstream analysis.

Usage

```
pca_analysis_enhanced(
  normalized_data = NULL,
  data_path = NULL,
  config = NULL,
  processing_result = NULL,
  min_var = NULL,
  impute = NULL,
  scale_data = NULL,
  n_components = NULL,
  variance_cutoff = NULL,
  grouping_variables = NULL,
  sample_id_components = NULL,
  value_column = "Normalized_Value",
  variable_column = "Variable",
  timepoint_column = "Timepoint",
  output_path = NULL,
  verbose = TRUE
)
```

Arguments

`normalized_data`
Data.frame. Pre-loaded MEA data in long format (default: NULL)

`data_path` Character. Path to Excel file containing MEA data (default: NULL)

`config` List. Configuration object with analysis parameters (default: NULL)

`processing_result`
List. Output from process_mea_flexible function (default: NULL)

`min_var` Numeric. Minimum variance threshold for variable inclusion (default: 0.01)

`impute` Logical. Whether to impute missing values (default: TRUE)

`scale_data` Logical. Whether to scale variables before PCA (default: TRUE)

`n_components` Integer. Number of principal components to extract (default: 2)

`variance_cutoff`
Numeric. Cumulative variance percentage threshold (default: 70)

`grouping_variables`
Character vector. Variables for sample grouping (default: c("Treatment", "Genotype"))

`sample_id_components`
Character vector. Variables to create unique sample IDs (default: c("Well", "Timepoint", "Treatment", "Genotype"))

`value_column` Character. Name of column containing values for PCA (default: "Normalized_Value")

`variable_column`
Character. Name of column containing variable names (default: "Variable")

`timepoint_column`
Character. Name of column containing timepoint information (default: "Timepoint")

`output_path` Character. Optional path to save elbow plot (default: NULL, no file saved)

`verbose` Logical. Whether to print detailed progress messages (default: TRUE)

Details

The function provides three flexible data input methods: 1. **processing_result**: Direct output from process_mea_flexible function 2. **data_path**: Path to Excel file with normalized_data sheet 3. **normalized_data**: Pre-loaded data frame in long format

Data processing includes: - Automatic detection of available columns - Flexible sample ID creation from specified components - Missing value imputation (mean, median, or zero) - Variance-based variable filtering - Automatic scaling option - Creation of elbow plot for component selection

The function handles common MEA data challenges: - Missing timepoint or treatment information - Inconsistent column naming - Mixed data types and missing values - Variable numbers of experiments and conditions

Method 1: Use output from MEA processing function process_mea_flexible("/path/to/data", baseline_timepoint = "baseline") pca_analysis_enhanced(processing_result = mea_result)

Method 2: Load from saved Excel file pca_analysis_enhanced(data_path = "/path/to/processed_data.xlsx")

Method 3: Use pre-loaded data with custom parameters normalized_data = my_data

Value

A list containing: - pca_result: Complete prcomp() object with PCA results - plot_data: Data frame ready for plotting with PC scores and metadata - variance_explained: Vector of variance explained by each component - cumulative_variance: Vector of cumulative variance explained - elbow_plot: ggplot2 object showing variance explained by components - elbow_data: Data frame underlying the elbow plot - components_needed: Number of components needed for various variance thresholds - count_summary: Summary of sample counts by groups (if applicable) - data_info: Information about data processing steps - config_used: Configuration parameters actually used - processing_source: Source of input data ("processing_result", "excel_file", or "direct_data")

`pca_plots_enhanced` *Enhanced PCA Plotting for Neural and Omics Data*

Description

Creates publication-ready PCA plots with scientific color palettes, flexible aesthetic mapping, and multiple visualization options. Designed specifically for neural activity and omics datasets with support for complex experimental designs including treatments, genotypes, and timepoints.

Usage

```
pca_plots_enhanced(
  pca_output = NULL,
  plot_data = NULL,
  pca_result = NULL,
  output_dir = NULL,
  processing_result = NULL,
  experiment_name = NULL,
  grouping_variables = NULL,
  color_variable = "Treatment",
  shape_variable = "Genotype",
  secondary_shape_variable = "Timepoint",
  pannels_var = NULL,
  components = c(1, 2),
  gray_color_value = NULL,
  save_plots = FALSE,
  plot_width = 12,
  plot_height = 10,
  dpi = 300,
  verbose = TRUE
)
```

Arguments

`pca_output`	List. Complete PCA output object from pca_analysis_enhanced() (optional)
`plot_data`	Data.frame. Data containing PC coordinates and metadata variables

`pca_result` List. PCA result object (e.g., from prcomp() or princomp())

`output_dir` Character. Directory path for saving plots (default: NULL, no files saved)

`processing_result` List. Result object from process_mea_flexible() (optional)

`experiment_name` Character. Name for the experiment (used in titles and filenames)

`grouping_variables` Character vector. Available metadata variables for plotting (default: c("Treatment", "Genotype", "Timepoint"))

`color_variable` Character. Variable name for color aesthetic (default: "Treatment")

`shape_variable` Character. Variable name for shape aesthetic (default: "Genotype")

`secondary_shape_variable` Character. Alternative shape variable (default: "Timepoint")

`pannels_var` Character. Variable for panel faceting (default: NULL)

`components` Numeric vector. PC components to plot (default: c(1, 2))

`gray_color_value` Character. Specific value of color_variable to display in gray (default: NULL)

`save_plots` Logical. Whether to save plots to files (default: FALSE)

`plot_width` Numeric. Plot width in inches (default: 12)

`plot_height` Numeric. Plot height in inches (default: 10)

`dpi` Numeric. Plot resolution (default: 300)

`verbose` Logical. Whether to print progress messages (default: TRUE)

Details

The function creates up to 5 different plot variants. Files are only saved when save_plots = TRUE AND output_dir is explicitly provided.

Value

A list containing:

plots Named list of ggplot objects for each plot type

plot_data Data.frame with plotting data and metadata

variance_explained Numeric vector of variance explained by each component

components_plotted Numeric vector of components used in plots

color_palette Named character vector of colors used

shape_palette Named numeric vector of shapes used

plotting_config List of configuration parameters used

saved_files Character vector of saved file paths (if save_plots = TRUE)

See Also

`process_mea_flexible` for MEA data processing, `discover_mea_structure` for automatic data structure detection

`perform_mea_pca` *Perform MEA PCA Analysis*

Description

Template function for performing PCA on MEA data

Usage

```
perform_mea_pca(data, variables = NULL, scale = TRUE, center = TRUE, ...)
```

Arguments

`data`	Data frame or tibble with processed MEA data
`variables`	Character vector. Variables to include in PCA (if NULL, uses all numeric)
`scale`	Logical. Whether to scale variables before PCA (default: TRUE)
`center`	Logical. Whether to center variables before PCA (default: TRUE)
`...`	Additional PCA parameters

Value

List containing PCA results (scores, loadings, variance explained, etc.)

Perform PCA analysis (requires processed MEA data)

`plot_pca_trajectories_general` *Plot PCA Trajectories for Time Series Data*

Description

This function creates comprehensive visualizations of PCA trajectories over time, showing both individual and group-averaged trajectories with optional smoothing.

Usage

```
plot_pca_trajectories_general(
  pca_results,
  pc_x = "PC1",
  pc_y = "PC2",
  trajectory_grouping = NULL,
  timepoint_var = "Timepoint",
  timepoint_order = NULL,
  individual_var = "Experiment",
  point_size = 3,
  alpha = 0.7,
  line_size = 2,
  smooth_lines = FALSE,
  color_palette = NULL,
  save_plots = FALSE,
  output_dir = NULL,
  plot_prefix = "PCA_trajectories",
  width = 12,
  height = 8,
  dpi = 150,
  return_list = TRUE,
  verbose = TRUE
)
```

Arguments

`pca_results`	A data frame or list containing PCA results
`pc_x`	Character string specifying the principal component for x-axis (default: "PC1")
`pc_y`	Character string specifying the principal component for y-axis (default: "PC2")
`trajectory_grouping`	Character vector of column names for grouping trajectories
`timepoint_var`	Character string specifying the timepoint column (default: "Timepoint")
`timepoint_order`	Character vector specifying the order of timepoints
`individual_var`	Character string for individual trajectory identification (default: "Experiment")
`point_size`	Numeric value controlling point size (default: 3)
`alpha`	Numeric value controlling transparency (default: 0.7)
`line_size`	Numeric value controlling line thickness (default: 2)
`smooth_lines`	Logical indicating whether to apply smoothing (default: FALSE)
`color_palette`	Character vector of colors for groups
`save_plots`	Logical indicating whether to save plots (default: FALSE)
`output_dir`	Character string specifying output directory (default: NULL)
`plot_prefix`	Character string prefix for filenames (default: "PCA_trajectories")
`width`	Numeric plot width in inches (default: 12)
`height`	Numeric plot height in inches (default: 8)
`dpi`	Numeric plot resolution (default: 150)
`return_list`	Logical indicating whether to return results as list (default: TRUE)
`verbose`	Logical indicating whether to print messages (default: TRUE)

Value

A list containing plots, trajectories, and metadata

print_detailed_summary

Print Detailed PCA Variable Summary

Description

Prints formatted summary of PCA variable importance analysis

Usage

```
print_detailed_summary(
  top_vars,
  pc_x_top,
  pc_y_top,
  high_both,
  pc_x,
  pc_y,
  top_n,
  min_loading_threshold
)
```

Arguments

`top_vars`	Data frame of top variables by combined importance
`pc_x_top`	Data frame of top variables for first PC
`pc_y_top`	Data frame of top variables for second PC
`high_both`	Data frame of variables important in both PCs
`pc_x`	Name of first principal component
`pc_y`	Name of second principal component
`top_n`	Number of top variables to display
`min_loading_threshold`	Minimum loading threshold

Value

NULL (prints to console)

process_mea_flexible *Process MEA Data Flexibly*

Description

This function processes Multi-Electrode Array (MEA) data files by reading CSV files, extracting measurements and metadata, applying filters, and optionally normalizing to baseline conditions. It automatically excludes standard deviation variables and handles exclusion flags to produce clean, analysis-ready datasets.

Usage

```
process_mea_flexible(
  main_dir,
  selected_experiments = NULL,
  selected_timepoints = NULL,
  grouping_variables = c("Treatment", "Genotype"),
  baseline_timepoint = NULL,
  unique_id_vars = c("Well", "Variable"),
  exclude_std_variables = TRUE,
  experiment_pattern = "MEA\\d+",
  timepoint_fusions = NULL,
  verbose = TRUE,
  output_path = NULL
)
```

Arguments

`main_dir` Character. Path to the main directory containing experiment folders

`selected_experiments`
Character vector. Experiment names to process (default: NULL = all)

`selected_timepoints`
Character vector. Timepoints to include (default: NULL = all)

`grouping_variables`
Character vector. Metadata columns to include ("Treatment", "Genotype")

`baseline_timepoint`
Character. Timepoint to use for normalization (default: NULL = no normalization)

`unique_id_vars` Character vector. Variables that uniquely identify observations for normalization

`exclude_std_variables`
Logical. Whether to automatically exclude standard deviation variables (default: TRUE)

`experiment_pattern`
Character. Regex pattern for experiment directories (default: "MEA\d+")

`timepoint_fusions`
Timepoint fusions to generate

`verbose`	Logical. Whether to print progress messages (default: TRUE)
`output_path`	Character. Optional path for output file (default: NULL saves to main_dir with auto-generated name)

Details

The function automatically detects and excludes variables containing "Std", "std", or "STD" in their names (e.g., "Number of Spikes - Std") while keeping average/mean variables (e.g., "Number of Spikes - Avg"). Wells marked with "Ex" or "ex" in row 124 are excluded.

By default, no files are written. To save output, provide an explicit output_path parameter. Normalization creates fold-change values relative to baseline timepoint.

Process data without saving (returns data frames only) Save output by providing explicit path

Value

A list containing: - raw_data: Processed data in long format - normalized_data: Baseline-normalized data (if baseline_timepoint specified) - processing_params: List of parameters used for processing - output_path: Path to saved Excel file (only if output_path was provided) - experiment_name: Combined experiment identifier

`quality_filter` *Filter Data by Quality Metrics*

Description

Filters variables and groups based on observation counts and data completeness

Usage

```
quality_filter(
  data,
  variable_column,
  value_column,
  grouping_columns,
  quality_threshold,
  min_observations,
  verbose
)
```

Arguments

`data`	Data frame to filter
`variable_column`	Column name containing variable identifiers
`value_column`	Column name containing values to assess

`grouping_columns`
: Vector of column names for grouping

`quality_threshold`
: Minimum data completeness ratio (0-1)

`min_observations`
: Minimum number of observations required

`verbose`
: Whether to print filtering results

Value

Filtered data frame

Examples

```
test_data <- data.frame(
  Variable = rep(paste0("V", 1:5), each = 20),
  Value = rnorm(100),
  Group = rep(c("A", "B"), 50)
)
filtered <- quality_filter(test_data, "Variable", "Value", "Group",
                           0.8, 5, FALSE)
```

`setup_color_scheme` *Setup Color Scheme*

Description

Sets up color schemes for plotting functions

Usage

```
setup_color_scheme(color_scheme, custom_colors)
```

Arguments

`color_scheme` Name of color scheme to use

`custom_colors` Custom color list (optional)

Value

List of colors for plotting

Index