# Package ‘bmdx’

October 24, 2023

**Title** BMDx: Dose dependent analysis

**Version** 2.0

**Description** The BMDx R package offers a robust solution for Benchmark Dose (BMD) analysis of transcriptomics data. The package employs a sophisticated approach involving the fitting of diverse models and selecting the optimal one based on the Akaike Information Criterion (AIC) or model average. Key functionalities of BMDx include the computation of BMD, related values, and IC50/EC50 estimations.BMDx particularly excels in comparing BMD values across different time points within a transcriptomics experiment. BMDx is adept at handling and analyzing multiple experiments concurrently, enhancing the efficiency of dataset assessment and promoting informed decision-making through thorough data analysis.

**License** GPL (>= 3)

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.2.3

**Depends** R (>= 3.5.0)

**Imports** car,
dendextend,
dplyr,
drc,
ggplot2,
heatmaply,
investr,
limma,
modelr,
readxl,
trend,
viridis,
utils,
minpack.lm,
stats,
forcats,
plotly,
UpSetR,
igraph,
visNetwork,
networkD3

## R topics documented:

---

## `add_average_models` *Add average models to a list of fitted models*

---

### Description

This function adds average models to a list of fitted models based on the model averaging approach described in "A brief guide to model selection, multimodel inference and model averaging in behavioural ecology using Akaike's information criterion" by Matthew R. E. Symonds and Adnan Moussalli. The average model is computed using the provided models and added to the list under the name "average". The average model is fitted using the data from the first model in the list. The function returns the updated list of fitted models.

### Usage

```
add_average_models(fitted_models)
```

### Arguments

| | |
|---|---|
| `fitted_models` | The list of fitted models. |

### Value

The updated list of fitted models with the average model added.

---

## `aggregate_rows_time` *Aggregate Rows by Time and Other Features*

---

### Description

Aggregates rows of a data frame based on time and other specified features.

### Usage

```
aggregate_rows_time(
  mod_stats,
  gen_feat = "Feature",
  first_feat = "SACRI_PERIOD",
  group_by = c("Experiment", "DILI"),
  filter_column = NULL,
  filter_by = list(c("acetaminophen", "bucetin"))
)
```

### Arguments

| | |
|---|---|
| `mod_stats` | A data frame containing model statistics. |
| `gen_feat` | The general feature to aggregate. |
| `first_feat` | The first feature used in aggregation. |
| `group_by` | A character vector specifying the grouping features. |
| `filter_column` | A character vector specifying the columns to filter. |
| `filter_by` | A list of character vectors containing filter values for each column. |

### Value

Returns a ggplot2 plot of aggregated data or an empty plot if conditions are not met.

---

## `bmr_factor` *Calculate the benchmark response level*

---

### Description

This function calculates the benchmark response level based on the given risk factor, whether the response should be increased or decreased, and the background level. The calculation is based on Equation 14 from the paper referenced in the code.

### Usage

```
bmr_factor(risk_factor = 0.1, increase = TRUE, background_level = 0.01)
```

### Arguments

| | |
|---|---|
| `risk_factor` | The risk factor used to determine the benchmark response level. Default is 0.1. |
| `increase` | Logical value indicating whether the response should be increased. If TRUE, the response is increased; if FALSE, it is decreased. Default is TRUE. |
| `background_level` | The background level of the response. Default is 0.01. |

### Value

The benchmark response level calculated based on the input parameters.

## References

The benchmark response level calculation is based on Equation 14 from the following paper: Reference: doi:10.1111/j.1539-6924.1995.tb00095.x

---

`build.model.matrix` *Build Model Matrix*

---

## Description

This function constructs the design matrix for a linear model based on the provided phenotype data, variable of interest, and optional covariates.

## Usage

```
build.model.matrix(
  pd,
  intercept = -1,
  var.int,
  covariates = NULL,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| `pd` | The phenotype data as a data frame. |
| `intercept` | The value to be used for the intercept (default: -1). |
| `var.int` | The variable of interest used in the design matrix. |
| `covariates` | A character vector specifying the optional covariates to include in the design matrix (default: NULL). |
| `verbose` | Logical; if TRUE, print the formula used for model.matrix (default: TRUE). |

## Value

The design matrix for the linear model.

---

`build_aop_for_aop_fingeprints`
*Build AOP Enrichment Results for AOP Fingerprints*

---

## Description

This function constructs AOP enrichment results for AOP fingerprints based on specified criteria.

### Usage

```
build_aop_for_aop_fingeprints(
  aop_enrichment_results,
  ke_enrichment_results,
  time_var,
  min_aop_length = 6,
  percentage_enriched_ke = 0.33,
  aop_ke_table_hure
)
```

### Arguments

| | |
|---|---|
| `aop_enrichment_results` | Enrichment results for AOPs. |
| `ke_enrichment_results` | Enrichment results for KEs. |
| `time_var` | Variable representing time points. |
| `min_aop_length` | Minimum length of AOPs. Default is 6. |
| `percentage_enriched_ke` | Percentage of enriched KEs for an AOP to be considered significant. Default is 33%. |
| `aop_ke_table_hure` | AOP-KE mapping table. |

### Value

A list containing two data frames: detailed_results_only_enriched and detailed_results_all_ke_in_aop.

---

## `build_models` *Build multiple models based on the given model names.*

---

### Description

This function constructs a list of model objects based on the specified model names. The available model families include "linear", "hill", "power", "poly2", "poly3", "poly4", "poly5", "exp2", "exp3", "exp4", "exp5", "llog2", "llog3", "llog4", "llog5", "mm2", "weibul12", "weibul13", "weibul14", "weibul22", "weibul23", and "weibul24".

### Usage

```
build_models(
 model_names = c("linear", "hill", "power", "poly2", "poly3", "poly4", "poly5", "exp2",
   "exp3", "exp4", "exp5", "llog2", "llog3", "llog4", "llog5", "mm2", "weibul12",
    "weibul13", "weibul14", "weibul22", "weibul23", "weibul24"),
  max_iter = 1024,
  data_type = c("continuous", "binomial"),
  x,
  y
)
```

## Arguments

| | |
|---|---|
| `model_names` | A character vector specifying the model names to build. Default is all available model names. |
| `max_iter` | Maximum number of iterations for iterative model fitting. Default is 1024. |
| `data_type` | Type of data, either "continuous" or "binomial". Default is "continuous". |
| `x` | The predictor variable (independent variable). |
| `y` | The response variable (dependent variable). |

## Value

model_list_result A list containing the specified model objects.

---

`cluster_genes_pairs` *Cluster gene pairs based on their correlation patterns This function clusters gene pairs based on their correlation patterns, using the provided comparison_pairs and nclust (number of clusters).*

---

## Description

Cluster gene pairs based on their correlation patterns This function clusters gene pairs based on their correlation patterns, using the provided comparison_pairs and nclust (number of clusters).

## Usage

```
cluster_genes_pairs(comparison_pairs, nclust = 2)
```

## Arguments

| | |
|---|---|
| `comparison_pairs` | A data frame containing gene pairs comparison, including Feature 1, Feature 2, and CorGenePatteerns. |
| `nclust` | An integer specifying the number of clusters to create (default is 2). |

## Value

A list containing heatmap analysis and cluster assignments for each gene pair.

## `compute_deviations` *Compute deviations for BMD modeling*

### Description

This function computes deviations for BMD modeling based on the specified deviation type. Three different types of deviation are available: standard deviation, relative deviation, and absolute deviation. Refer to the EPA documentation for detailed information on the deviation types (slide 7) from the provided link.

### Usage

```
compute_deviations(deviation_type = "standard", model, rl = 1.349)
```

### Arguments

| | |
|---|---|
| `deviation_type` | Character string specifying the type of deviation to compute. Possible values are "standard", "relative", and "absolute". |
| `model` | The model object used for computation. |
| `rl` | The relative level used in the deviation calculation. Default is 1.349. |

### Value

The computed deviation value based on the specified deviation type.

### References

For detailed information on the deviation types, refer to the EPA documentation available at: `https://clu-in.org/conf/tio/bmds/slides/BMDS_Continuous_Models.pdf`

## `compute_gene_frequencies` *Compute Gene Frequencies and Create Lollipop Plots*

### Description

This function computes gene frequencies based on the provided model statistics and generates lollipop plots for the top genes based on their frequencies.

### Usage

```
compute_gene_frequencies(
  mod_stats,
  th = 0.7,
  rel_variable = "Experiment",
  group_by = "None",
  split_by = "None"
)
```

### Arguments

| | |
|---|---|
| `mod_stats` | A data frame containing the model statistics and gene information. |
| `th` | Threshold value for gene percentage. Only genes with a percentage above this threshold will be plotted. |
| `rel_variable` | The name of the variable representing the experiments or conditions. |
| `group_by` | The name of the variable to group the data for generating lollipop plots. |
| `split_by` | The name of the variable to split the data and generate separate lollipop plots. |

### Value

A list containing gene lists, lollipop plots, and matrices for each split-by value or for all data.

---

## `compute_model_statistics` *Compute model statistics for fitted models*

---

### Description

This function computes model statistics for a list of fitted models and returns the results as a data frame.

### Usage

```
compute_model_statistics(
  fitted_models,
  other_variables_id_col,
  is_parallel = TRUE,
  nCores = 2
)
```

### Arguments

| | |
|---|---|
| `fitted_models` | A list of fitted models. |
| `other_variables_id_col` | The name of the column representing the other variables in the model statistics data frame. |
| `is_parallel` | Whether to compute the statistics in parallel. Default is TRUE. |
| `nCores` | The number of cores to use for parallel computation. Default is 2. |

### Value

A data frame containing the computed model statistics.

`compute_the_closest_AOs`

*Compute closest adverse outcomes to a given vertex*

## Description

This function takes in a vertex (or set of vertices) of interest, an igraph object representing the KE-KE network, a vector of adverse outcomes (AO), and optional parameters threshold and distance, and returns the closest adverse outcomes to the vertex of interest in the KEKE_net. If distance is set to TRUE, it returns both the names of the closest adverse outcomes and the corresponding distances between the vertex of interest and the adverse outcomes.

## Usage

```
compute_the_closest_AOs(
  interesting_vertex,
  KEKE_net,
  AO,
  threshold = 20,
  distance = FALSE
)
```

## Arguments

| | |
|---|---|
| `interesting_vertex` | A string representing the name of the vertex/vertices of interest in the KEKE_net. |
| `KEKE_net` | An igraph object representing the knowledge exchange network. |
| `AO` | A vector of adverse outcomes. |
| `threshold` | An integer representing the number of closest adverse outcomes to return (default is 20). |
| `distance` | A logical value (TRUE or FALSE) indicating whether to return both the names of the closest adverse outcomes and the corresponding distances (default is FALSE). |

## Value

If distance is set to FALSE, the function returns a vector of the names of the closest adverse outcomes to the vertex of interest in the KEKE_net. If distance is set to TRUE, the function returns a list of two elements: a vector of the names of the closest adverse outcomes, and a vector of the corresponding distances between the vertex of interest and the adverse outcomes.

---

`compute_the_closest_MIEs`
*Compute closest molecular initiating events to a given vertex/set of vertices*

---

## Description

This function takes in a vertex (or set of vertices) of interest, an igraph object representing the KE-KE network, a vector of adverse outcomes (AO), and optional parameters threshold and distance, and returns the closest adverse outcomes to the vertex of interest in the KEKE_net. If distance is set to TRUE, it returns both the names of the closest adverse outcomes and the corresponding distances between the vertex of interest and the adverse outcomes.

## Usage

```
compute_the_closest_MIEs(
  interesting_vertex,
  KEKE_net,
  MIE,
  threshold = 5,
  distance = TRUE
)
```

## Arguments

| | |
|---|---|
| `interesting_vertex` | A string representing the name of the vertex/vertices of interest in the KEKE_net. |
| `KEKE_net` | An igraph object representing the knowledge exchange network. |
| `MIE` | A character vector representing the list of molecular initiating events (MIEs). |
| `threshold` | An integer representing the number of closest adverse outcomes to return (default is 20). |
| `distance` | A logical value (TRUE or FALSE) indicating whether to return both the names of the closest adverse outcomes and the corresponding distances (default is FALSE). |

## Value

If distance is set to FALSE, the function returns a vector of the names of the closest adverse outcomes to the vertex of interest in the KEKE_net. If distance is set to TRUE, the function returns a list of two elements: a vector of the names of the closest adverse outcomes, and a vector of the corresponding distances between the vertex of interest and the adverse outcomes.

## create_data_structure *this function convert the data in input into the format required for dose-dependent modelling*

### Description

this function convert the data in input into the format required for dose-dependent modelling

### Usage

```
create_data_structure(
  experimental_data,
  metadata,
  sample_id_col = "BARCODE",
  dose_id_col = "DOSE",
  other_variables_id_col = c("SACRI_PERIOD"),
  x = "dose",
  y = "expr"
)
```

### Arguments

| | |
|---|---|
| `experimental_data` | a list of dataframe containing experimental data. Each row is a feature (e.g. gene) and each column is a sample |
| `metadata` | a list of dataframe containing the metadata for the experimental data. Each row is a sample and the columns represent the different variables. A column for dose/concentration is required |
| `sample_id_col` | a character specifying the name of the column containing the samples id |
| `dose_id_col` | a character specifying the name of the column containing the doses/concentration |
| `other_variables_id_col` | a vector of characters specifying the name of the column used to group the data |
| `x` | a characters specifying the name of the x variable in the model. Default is dose. |
| `y` | a characters specifying the name of the y variable in the model. Default is expr. |

### Value

a dictionary containing the data frame for modelling. Dictionary Keys are n-uple specifying the experiment name, other variables of interests and feature names (e.g. drug, time, gene)

## `diff.gene.expr` *Differential Gene Expression Analysis*

### Description

This function performs differential gene expression analysis using limma's linear model with specified contrasts and adjustment method.

### Usage

```
## S3 method for class 'gene.expr'
diff(data, des, contrasts, adjust.method)
```

### Arguments

| | |
|---|---|
| `data` | The gene expression data as a data frame or matrix. |
| `des` | The design matrix representing the experimental design. |
| `contrasts` | A character vector specifying the contrasts for analysis. |
| `adjust.method` | The method for p-value adjustment (default: "none"). Options are "holm", "hochberg", "hommel", "bonferroni", "BH", "BY", "fdr", "none". |

### Value

A list containing topTable results for each contrast specified.

## `dose_response_analysis` *Perform dose-response analysis on a list of models*

### Description

This function fits a list of models to the same end-point and performs dose-response analysis, including estimation of BMD, BMDL, BMDU, and AC50 values.

### Usage

```
dose_response_analysis(
  data,
  model_list,
  deviation_type = "standard",
  rl = 1.349,
  variance_type = "constant",
  confidence_interval = 0.95,
  significance_level = 0.05
)
```

### Arguments

| | |
|---|---|
| `data` | The data used for fitting the models. |
| `model_list` | A list of models to fit to the data. |
| `deviation_type` | Character string specifying the type of deviation from the fitted model to use for BMD calculation. Default is "standard". Allowed values are standard and relative |
| `rl` | The relative level used to calculate the BMD. Default is 1.349. |
| `variance_type` | Character string specifying the type of variance to use in model fitting. Default is "constant". Other possible values are "non constant", "model" and "inferred" |
| `confidence_interval` | The confidence level for the confidence interval. Default is 0.95. |
| `significance_level` | The significance level for model fitting. Default is 0.05. |

### Value

A list of models with additional attributes for BMDL, BMDU, and AC50 values. Models that fail to fit or encounter an error during estimation are excluded from the returned list.

---

## `ecdf_plots` *Make ECDF or Histogram Plots for BMD Data*

---

### Description

This function creates ECDF (Empirical Cumulative Distribution Function) or Histogram plots for the BMD (Benchmark Dose) data in the provided model statistics data frame.

### Usage

```
ecdf_plots(
  mod_stats,
  rel_variable = "Experiment",
  group_by = NULL,
  is_group_by_numeric = TRUE,
  other_variables = NULL,
  number_of_column = 2,
  scaling = TRUE,
  filter_column = c("Model"),
  filter_by = list(c("linear")),
  plot_type = "ecdf"
)
```

### Arguments

| | |
|---|---|
| `mod_stats` | The model statistics data frame. |
| `rel_variable` | The column name in `mod_stats` representing the experimental condition or treatment (default: "Experiment"). |
| `group_by` | The column name in `mod_stats` used to group the data for faceting (default: NULL). |

| | |
|---|---|
| `is_group_by_numeric` | Logical; if TRUE, treat the `group_by` variable as numeric (default: TRUE). |
| `other_variables` | A character vector specifying additional variables for faceting (default: NULL). |
| `number_of_column` | The number of columns in the facet grid (default: 2). |
| `scaling` | Logical; if TRUE, scale the BMD values (default: TRUE). |
| `filter_column` | The column name(s) in `mod_stats` to use for filtering (default: "Model"). |
| `filter_by` | A list of character vectors containing filtering criteria for each `filter_column` (default: list(c("linear"))). |
| `plot_type` | The type of plot to create; either "ecdf" for ECDF or "histogram" for histogram (default: "ecdf"). |

## Value

A plotly object representing the ECDF or histogram plot.

---

## `enrich_KEs_AOPs` *Perform Enrichment Analysis for Key Events*

---

### Description

Conduct enrichment analysis for key events (KEs) using the provided data and parameters.

### Usage

```
enrich_KEs_AOPs(
  BMD_TAB,
  experimental_data,
  experiment_var,
  time_var,
  other_variables_id_col,
  list_gene_sets,
  aop_ke_table_hure,
  mapped_genes,
  only_significant = FALSE,
  pval_th = 0.05,
  adj.method = "fdr",
  merge_by = "Ke"
)
```

### Arguments

| | |
|---|---|
| `BMD_TAB` | A data frame containing benchmark dose (BMD) information. |
| `experimental_data` | A list of data frames containing experimental data for different experiments. |
| `experiment_var` | The variable representing the experiment ID. |
| `time_var` | The variable representing the time point ID. |

| | |
|---|---|
| `other_variables_id_col` | The column name for other variables used as IDs. |
| `list_gene_sets` | A list of gene sets for enrichment analysis. |
| `aop_ke_table_hure` | A data frame containing Adverse Outcome Pathway (AOP) information. |
| `mapped_genes` | A vector of mapped genes. |
| `only_significant` | Logical indicating whether to include only significant results. |
| `pval_th` | The p-value threshold for significance. |
| `adj.method` | The method for p-value adjustment. |
| `merge_by` | The variable for merging the results. |

## Value

A list of enriched key events (KEs) and associated results.

## `ens_human_gene_graph` *ens_human_gene_graph*

### Description

Graph whose nodes are Human genes Ensembl ID and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
ens_human_gene_graph
```

### Format

An object of class `igraph` of length 6854.

### Source

https://doi.org/10.1093/bib/bbaa417

## `ens_mouse_gene_graph` *ens_mouse_gene_graph*

### Description

Graph whose nodes are mouse genes Ensembl ID and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
ens_mouse_gene_graph
```

### Format

An object of class `igraph` of length 4740.

### Source

https://doi.org/10.1093/bib/bbaa417

---

## `ens_rat_gene_graph` *ens_rat_gene_graph*

---

### Description

Graph whose nodes are rat genes Ensembl ID and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
ens_rat_gene_graph
```

### Format

An object of class `igraph` of length 3258.

### Source

https://doi.org/10.1093/bib/bbaa417

---

## `ent_human_gene_graph` *ent_human_gene_graph*

---

### Description

Graph whose nodes are Human entrez genes and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
ent_human_gene_graph
```

### Format

An object of class `igraph` of length 120356739.

### Source

https://doi.org/10.1093/bib/bbaa417

## `ent_mouse_gene_graph` *ent_mouse_gene_graph*

### Description

Graph whose nodes are mouse entrez genes and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
ent_mouse_gene_graph
```

### Format

An object of class `igraph` of length 102636989.

### Source

https://doi.org/10.1093/bib/bbaa417

## `ent_rat_gene_graph` *ent_rat_gene_graph*

### Description

Graph whose nodes are rat entrez genes and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
ent_rat_gene_graph
```

### Format

An object of class `igraph` of length 120099275.

### Source

https://doi.org/10.1093/bib/bbaa417

## filter_df *this function filters the dataframe of the models statistic*

### Description

this function filters the dataframe of the models statistic

### Usage

```
filter_df(mod_stats, filter_column, filter_by)
```

### Arguments

| | |
|---|---|
| `mod_stats` | the dataframe containing all the statistics |
| `filter_column` | a vector of dataframe columns to be used for filtering |
| `filter_by` | a list containing the selected values to be mantained after filtering. Each position of the list correspond to one of the selected column and contains a vector of admissible values #The length of filter_column and filter_by must be the same |

### Value

the filtered dataframe

## filter_df_no *Filter DataFrame by Column Values*

### Description

Filters a data frame based on specified columns and filter values.

### Usage

```
filter_df_no(mod_stats, filter_column, filter_by)
```

### Arguments

| | |
|---|---|
| `mod_stats` | A data frame containing model statistics. |
| `filter_column` | A character vector specifying the columns to filter. |
| `filter_by` | A list of character vectors containing filter values for each column. |

### Value

Returns a filtered data frame if filtering conditions are met; otherwise, NULL.

## filter_pval *Filter results based on p-value threshold*

### Description

This function filters the results based on a specified p-value threshold. It returns the filtered ANOVA results and the indices of the filtered rows.

### Usage

```
filter_pval(filtering_res, Pval.th = 0.05)
```

### Arguments

| | |
|---|---|
| `filtering_res` | The filtering results table. |
| `Pval.th` | The p-value threshold for filtering the results. Default is 0.05. |

### Value

A list containing the filtered results and the indices of the filtered rows.

## filter_pval_th *Filter Fold Change DataFrame by p-value and fold change threshold*

### Description

This function filters a fold change data frame based on given p-value and fold change thresholds.

### Usage

```
filter_pval_th(fold_change_dataframe, fcPval.th = 0.05, fc.th = 0.58)
```

### Arguments

| | |
|---|---|
| `fold_change_dataframe` | A data frame containing fold change results. |
| `fcPval.th` | The p-value threshold for filtering (default: 0.05). |
| `fc.th` | The fold change threshold for filtering (default: 0.58). |

### Value

A numeric vector containing the indices of rows that pass the filtering criteria.

`find_best_model_aic` *Find the best model based on AIC (Akaike Information Criterion) This function finds the best model based on the Akaike Information Criterion (AIC) among a list of models.*

## Description

Find the best model based on AIC (Akaike Information Criterion) This function finds the best model based on the Akaike Information Criterion (AIC) among a list of models.

## Usage

```
find_best_model_aic(model_list)
```

## Arguments

| | |
|---|---|
| `model_list` | A list containing multiple models to be compared. |

## Value

The model from the list that has the lowest AIC value.

`fitting_list` *Fit dose-response models to data dictionary*

## Description

This function fits dose-response models to a data dictionary using the specified model list. It returns a list of fitted models.

## Usage

```
fitting_list(
  data_dictionary,
  model_list,
  deviation_type = "relative",
  rl = 1.349,
  confidence_interval = 0.95,
  variance_type = "constant",
  significance_level = 0.05,
  is_parallel = FALSE,
  nCores = 2
)
```

### Arguments

| | |
|---|---|
| `data_dictionary` | The data dictionary containing the data frames to fit the models to. |
| `model_list` | A list of dose-response models to fit. |
| `deviation_type` | The type of deviation to use for BMD calculations. Default is "relative". |
| `rl` | The constant value for relative deviation. Default is 1.349. |
| `confidence_interval` | The confidence level for the BMD calculation. Default is 0.95. |
| `variance_type` | The type of variance to assume for the models. Default is "constant". |
| `significance_level` | The significance level for the BMD calculation. Default is 0.05. |
| `is_parallel` | Whether to perform the fitting in parallel. Default is FALSE. |
| `nCores` | The number of cores to use for parallel computation. Default is 2. |

### Value

A list containing the fitted models.

---

## `gene_pairs_analysis` *Perform gene pairs analysis for selected experiments and times This function performs gene pairs analysis for selected experiments and times, using the provided filtered_optimal_models, BMD_tab, length_vectors, nCores, phenoList, doseColID, timeColID, and other_variables_id_col.*

---

### Description

Perform gene pairs analysis for selected experiments and times This function performs gene pairs analysis for selected experiments and times, using the provided filtered_optimal_models, BMD_tab, length_vectors, nCores, phenoList, doseColID, timeColID, and other_variables_id_col.

### Usage

```
gene_pairs_analysis(
  select_experiment,
  select_time,
  filtered_optimal_models,
  BMD_tab,
  length_vectors,
  nCores,
  phenoList,
  doseColID,
  timeColID,
  other_variables_id_col
)
```

### Arguments

| | |
|---|---|
| `select_experiment` | A character vector specifying the selected experiments for the analysis. |
| `select_time` | A numeric vector specifying the selected times for the analysis. |
| `filtered_optimal_models` | A list of filtered optimal models. |
| `BMD_tab` | The BMD (Benchmark Dose) table. |
| `length_vectors` | An integer specifying the length of vectors. |
| `nCores` | An integer specifying the number of cores to use for parallel processing. |
| `phenoList` | A list containing the phenotype data. |
| `doseColID` | The column ID for the dose in the phenotype data. |
| `timeColID` | The column ID for the time in the phenotype data. |
| `other_variables_id_col` | The column ID for other variables in the phenotype data. |

### Value

A list containing gene pairs statistics and the newdata used for the analysis.

---

## `gene_pairs_comparison` *Compare gene pairs for their dose-dependent patterns This function compares gene pairs for their dose-dependent patterns using the provided filtered_optimal_models, other_variables_id_col, and newdata. It calculates the difference, behavior, correlation, and coefficient for each pair of genes in the models.*

---

### Description

Compare gene pairs for their dose-dependent patterns This function compares gene pairs for their dose-dependent patterns using the provided filtered_optimal_models, other_variables_id_col, and newdata. It calculates the difference, behavior, correlation, and coefficient for each pair of genes in the models.

### Usage

```
gene_pairs_comparison(
  filtered_optimal_models,
  other_variables_id_col,
  newdata,
  nCores = 40
)
```

### Arguments

| | |
|---|---|
| `filtered_optimal_models` | A list of filtered optimal models containing gene-related information. |
| `other_variables_id_col` | The column ID for other variables in the gene data. |
| `newdata` | A data frame containing dose information for new observations. |
| `nCores` | An integer specifying the number of cores to use for parallel processing (default is 40). |

### Value

A data frame containing gene pairs statistics, including Experiment 1, Model 1, Experiment 2, Model 2, Difference Trend, Coefficient, and Correlation of Gene Patterns.

---

## get_model_stats — *given a model the function creates a named vector with all the model statistics*

---

### Description

given a model the function creates a named vector with all the model statistics

### Usage

```
get_model_stats(model)
```

### Arguments

| | |
|---|---|
| `model` | an object of class bmdx |

### Value

model_stats a named vector with statistics

---

## inner.check.model — *Check if a model passes the filtering criteria*

---

### Description

This function checks if a model passes the specified filtering criteria. The criteria include thresholds, lack of fit, ratios, missing values, R-squared, and monotonicity. The function returns TRUE if the model passes all the criteria, and FALSE otherwise.

### Usage

```
inner.check.model(
  mod,
  loofth = 0.1,
  lower_bound_th = 0.1,
  upper_bound_th = 0.1,
  bmd_bmdl_th = 20,
  bmdu_bmd_th = 20,
  bmdu_bmdl_th = 40,
  filter_lower_bound = TRUE,
  filter_upper_bound = TRUE,
  filter_by_lack_of_fit = TRUE,
  ratio_filter = TRUE,
  bmd_na_filter = TRUE,
  bmdl_na_filter = TRUE,
  bmdu_na_filter = TRUE,
  ic50_na_filter = TRUE,
  r2_filter = FALSE,
  r2_th = 0.6,
  filter_by_monotonicity = FALSE
)
```

## Arguments

| | |
|---|---|
| `mod` | The model to be checked. |
| `loofth` | The threshold for lack of fit. Default is 0.1. |
| `lower_bound_th` | The lower bound threshold. Default is 0.1% of the lowest dose. |
| `upper_bound_th` | The upper bound threshold. Default is 0.1% of the highest dose. |
| `bmd_bmdl_th` | The threshold for the ratio of BMD to BMDL. Default is 20. |
| `bmdu_bmd_th` | The threshold for the ratio of BMDU to BMD. Default is 20. |
| `bmdu_bmdl_th` | The threshold for the ratio of BMDU to BMDL. Default is 40. |
| `filter_lower_bound` | Logical value indicating whether to filter the model based on the lower bound threshold. Default is TRUE. |
| `filter_upper_bound` | Logical value indicating whether to filter the model based on the upper bound threshold. Default is TRUE. |
| `filter_by_lack_of_fit` | Logical value indicating whether to filter the model based on lack of fit. Default is TRUE. |
| `ratio_filter` | Logical value indicating whether to filter the model based on ratios. Default is TRUE. |
| `bmd_na_filter` | Logical value indicating whether to filter the model with missing BMD values. Default is TRUE. |
| `bmdl_na_filter` | Logical value indicating whether to filter the model with missing BMDL values. Default is TRUE. |
| `bmdu_na_filter` | Logical value indicating whether to filter the model with missing BMDU values. Default is TRUE. |
| `ic50_na_filter` | Logical value indicating whether to filter the model with missing IC50 values. Default is TRUE. |
| `r2_filter` | Logical value indicating whether to filter the model based on R-squared. Default is FALSE. |
| `r2_th` | The threshold for R-squared. Default is 0.6. |
| `filter_by_monotonicity` | Logical value indicating whether to filter the model based on monotonicity. Default is FALSE. |

## Value

TRUE if the model passes the filtering criteria, FALSE otherwise.

## is_strictly_monotonic *Check if a fitted model is strictly monotonic*

### Description

This function checks if a fitted model exhibits strict monotonicity over the range of doses. It returns TRUE if the model is strictly monotonic and FALSE otherwise.

### Usage

```
is_strictly_monotonic(fittedModel)
```

### Arguments

| | |
|---|---|
| fittedModel | The fitted model to be checked. |

### Value

TRUE if the model is strictly monotonic, FALSE otherwise.

## loop_enrichment_v2 *Loop Enrichment Analysis for Multiple Experiment Genes*

### Description

This function performs loop enrichment analysis for multiple sets of experiment genes using gene sets.

### Usage

```
loop_enrichment_v2(
  list_experiment_genes,
  list_gene_sets,
  background,
  aop_ke_table_hure,
  only_significant = TRUE,
  pval_th = 0.05,
  adj.method = "fdr",
  merge_by = "Ke"
)
```

### Arguments

| | |
|---|---|
| list_experiment_genes | A list of experiment gene data frames. |
| list_gene_sets | A list of gene sets for enrichment analysis. |
| background | Background gene set for enrichment analysis. |
| aop_ke_table_hure | AOP-KE mapping table. |

| | |
|---|---|
| `only_significant` | Logical, whether to include only significant results. |
| `pval_th` | P-value threshold for significance. |
| `adj.method` | Adjustment method for p-values. |
| `merge_by` | Column used to merge results with the AOP-KE mapping table. Available options "Ke" or "Aop" |

## Value

A data frame containing the enriched results.

---

`make_d3_network` *Create a D3 Network Visualization*

---

## Description

This function generates a D3 network visualization based on statistical data and a graph structure.

## Usage

```
make_d3_network(statistics, g, th = 0, positive = TRUE)
```

## Arguments

| | |
|---|---|
| `statistics` | A data frame containing the statistics on the gene-gene correlation data. |
| `g` | An igraph graph object representing gene-gene interactions. |
| `th` | A threshold for filtering correlations. |
| `positive` | Logical, whether to consider positive correlations. |

## Value

A list containing the D3 network visualization, a data frame of node statistics, and an igraph graph object.

---

`make_empty_plot` *Create Empty ggplot2 Plot Creates an empty ggplot2 plot with a void theme and no x-label.*

---

## Description

Create Empty ggplot2 Plot Creates an empty ggplot2 plot with a void theme and no x-label.

## Usage

```
make_empty_plot()
```

## Value

Returns an empty ggplot2 plot.

## make_visNetwork *Create a Visualization Network*

### Description

This function generates a visualization network (visNetwork) based on enrichment data and a network structure.

### Usage

```
make_visNetwork(
  experiment,
  time_var,
  tp = "24",
  detailed_results,
  KEKE_net,
  filter_ke_by_aop_fingerprint = TRUE,
  enlarge_ke_selection = TRUE,
  aop_ke_table_hure
)
```

### Arguments

| | |
|---|---|
| `experiment` | The name of the experiment. |
| `time_var` | The name of the time variable. |
| `tp` | Time point for which the network is created. |
| `detailed_results` | Detailed enrichment results data frame. |
| `KEKE_net` | The network structure. |
| `filter_ke_by_aop_fingerprint` | Logical, whether to filter KEs by AOP fingerprint. |
| `enlarge_ke_selection` | Logical, whether to enlarge KE selection. |
| `aop_ke_table_hure` | a dataframe containing 5 variables: AOP ids, KE ids, KE type description, KE name, AOP name |

### Value

A list containing nodes, edges, and a visNetwork object.

## `model_filtering` *Filter a list of fitted models based on various criteria*

### Description

This function filters a list of fitted models based on specified criteria, such as thresholds, lack of fit, ratios, missing values, R-squared, and monotonicity. The function returns a filtered list of models that pass the specified criteria.

### Usage

```
model_filtering(
  fitted_models,
  loofth = 0.1,
  lower_bound_th = 0.1,
  upper_bound_th = 0.1,
  bmd_bmdl_th = 20,
  bmdu_bmd_th = 20,
  bmdu_bmdl_th = 40,
  filter_lower_bound = TRUE,
  filter_upper_bound = TRUE,
  filter_by_lack_of_fit = TRUE,
  ratio_filter = TRUE,
  bmd_na_filter = TRUE,
  bmdl_na_filter = TRUE,
  bmdu_na_filter = TRUE,
  ic50_na_filter = TRUE,
  r2_filter = FALSE,
  r2_th = 0.6,
  filter_by_monotonicity = FALSE
)
```

### Arguments

| | |
|---|---|
| `fitted_models` | A list of fitted models to be filtered. |
| `loofth` | The threshold for lack of fit. Default is 0.1. |
| `lower_bound_th` | The lower bound threshold. Default is 0.1% of the lowest dose. |
| `upper_bound_th` | The upper bound threshold. Default is 0.1% of the highest dose. |
| `bmd_bmdl_th` | The threshold for the ratio of BMD to BMDL. Default is 20. |
| `bmdu_bmd_th` | The threshold for the ratio of BMDU to BMD. Default is 20. |
| `bmdu_bmdl_th` | The threshold for the ratio of BMDU to BMDL. Default is 40. |
| `filter_lower_bound` | Logical value indicating whether to filter models based on the lower bound threshold. Default is TRUE. |
| `filter_upper_bound` | Logical value indicating whether to filter models based on the upper bound threshold. Default is TRUE. |
| `filter_by_lack_of_fit` | Logical value indicating whether to filter models based on lack of fit. Default is TRUE. |

| | |
|---|---|
| `ratio_filter` | Logical value indicating whether to filter models based on ratios. Default is TRUE. |
| `bmd_na_filter` | Logical value indicating whether to filter models with missing BMD values. Default is TRUE. |
| `bmdl_na_filter` | Logical value indicating whether to filter models with missing BMDL values. Default is TRUE. |
| `bmdu_na_filter` | Logical value indicating whether to filter models with missing BMDU values. Default is TRUE. |
| `ic50_na_filter` | Logical value indicating whether to filter models with missing IC50 values. Default is TRUE. |
| `r2_filter` | Logical value indicating whether to filter models based on R-squared. Default is FALSE. |
| `r2_th` | The threshold for R-squared. Default is 0.6. |
| `filter_by_monotonicity` | Logical value indicating whether to filter models based on monotonicity. Default is FALSE. |

### Value

A filtered list of models that pass the specified criteria.

---

## `my_enrichment` *Perform Enrichment Analysis*

---

### Description

This function performs enrichment analysis using Fisher's exact test for gene sets based on provided statistical data and gene sets.

### Usage

```
my_enrichment(genes, reference, genesets, adj = "fdr", verbose = FALSE)
```

### Arguments

| | |
|---|---|
| `genes` | A data frame containing the gene-related BMD estimates |
| `reference` | A vector of reference genes. |
| `genesets` | A list of gene sets. |
| `adj` | The method used for adjusting p-values. |
| `verbose` | Logical, whether to display progress messages. |

### Value

A data frame with enriched gene sets and associated statistical information.

## perform_anova *Perform ANOVA on data dictionary*

### Description

This function performs ANOVA on a data dictionary, computes p-values, performs adjustment if specified, filters the results based on a significance threshold, and returns the ANOVA results and filtered data dictionary.

### Usage

```
perform_anova(
  data_dictionary,
  anovaAdjustment = "Nominal",
  anovaPval.th = 0.05,
  anovaCores = 1,
  x = "dose",
  y = "expr",
  other_variables_id_col = NULL
)
```

### Arguments

| | |
|---|---|
| `data_dictionary` | The data dictionary containing the data frames to perform ANOVA on. |
| `anovaAdjustment` | The adjustment method for p-values. Options include "Nominal" (no adjustment) and methods supported by the `p.adjust` function. Default is "Nominal". |
| `anovaPval.th` | The significance threshold for filtering the ANOVA results. Default is 0.05. |
| `anovaCores` | The number of cores to use for parallel computation. Default is 1. |
| `x` | The column name for the independent variable. Default is "dose". |
| `y` | The column name for the dependent variable. Default is "expr". |
| `other_variables_id_col` | Additional column names for other variables to include in the ANOVA. |

### Value

anova_res_list A list containing the ANOVA results dataframe, filtered data dictionary, unfiltered ANOVA results dataframe, and a plot list.

## perform_differential_expression_analysis_filtering

*Perform Differential Expression Analysis*

### Description

This function filter the data based on their differential analysis

### Usage

```
perform_differential_expression_analysis_filtering(
  data_dictionary,
  experimental_data,
  phTable,
  time_point_variable,
  dose_variable,
  samples_variable,
  fcAdjustment = "Nominal",
  fcPval.th = 0.05,
  fc.th = 1.5,
  nCores = 1,
  x = "dose",
  y = "expr",
  other_variables_id_col = NULL
)
```

### Arguments

| | |
|---|---|
| `data_dictionary` | A list of data frames representing the data dictionary. |
| `experimental_data` | A list of data frames containing the experimental data. |
| `phTable` | A list of data frames representing the phenotype table. |
| `time_point_variable` | The name of the variable representing time points. |
| `dose_variable` | The name of the variable representing doses. |
| `samples_variable` | The name of the variable representing samples. |
| `fcAdjustment` | The type of fold change adjustment (default: "Nominal"). Options are "none" or "whatever_other_adjustment". |
| `fcPval.th` | The p-value threshold for fold change filtering (default: 0.05). |
| `fc.th` | The fold change threshold (default: 1.5). |
| `nCores` | The number of cores for parallel processing (default: 1). |
| `x` | The name of the x-axis variable for plotting (default: "dose"). |
| `y` | The name of the y-axis variable for plotting (default: "expr"). |
| `other_variables_id_col` | A vector of other variable names used for filtering. |

### Value

A list containing the fold change results (filtered and unfiltered), the filtered data dictionary, and the filtered fold change results.

---

## `perform_trend_test` *Perform the trend test on multiple datasets.*

---

### Description

This function performs the trend test on a list of datasets stored in the `data_dictionary`. The trend test is performed using the `mk.test` function from the `trend` package. It calculates the trend p-value for each dataset and adjusts the p-values using the specified `trendAdjustment` method. It also performs filtering based on the `trendPval.th` threshold.

### Usage

```
perform_trend_test(
  data_dictionary,
  trendAdjustment = "Nominal",
  trendPval.th = 0.05,
  trendCores = 1,
  x = "dose",
  y = "expr",
  other_variables_id_col = NULL
)
```

### Arguments

| | |
|---|---|
| `data_dictionary` | A list of datasets, each stored as a data frame in the list. |
| `trendAdjustment` | The method for adjusting p-values. Default is "Nominal". Available options include "Bonferroni", "Holm", "Hochberg", "BH", "BY", "fdr", "none". |
| `trendPval.th` | The threshold for filtering p-values. Default is 0.05. |
| `trendCores` | The number of CPU cores to use for parallel processing. Default is 1. |
| `x` | The column name representing the predictor variable (independent variable) in each dataset. |
| `y` | The column name representing the response variable (dependent variable) in each dataset. |
| `other_variables_id_col` | The column name containing the identifier for other variables. Default is NULL. |

### Value

A list containing the trend test results and filtered datasets.

## plot_bmdx *Plot BMD (Benchmark Dose) Model*

### Description

This function generates a plot for a Benchmark Dose (BMD) model.

### Usage

```
plot_bmdx(
  model,
  cex = 6,
  x_pos = 15,
  y_pos_th = 0.85,
  confidence_interval = 0.95,
  title_label = "title"
)
```

### Arguments

| | |
|---|---|
| `model` | The fitted BMD model. |
| `cex` | The size of points in the plot (default: 6). |
| `x_pos` | The position of the x-axis (default: 15). |
| `y_pos_th` | The position of the y-axis threshold (default: 0.85). |
| `confidence_interval` | The confidence interval level (default: 0.95). |
| `title_label` | The title of the plot (default: "title"). |

### Value

A plot displaying the BMD model with confidence intervals and other related data points.

## plot_bmd_bmdl_bmdu_set_of_genes *Plot BMD, BMDL, and BMDU for a Set of Genes*

### Description

This function generates a plot showing BMD, BMDL, and BMDU values for a given set of genes.

### Usage

```
plot_bmd_bmdl_bmdu_set_of_genes(BMDFilMat, gi)
```

### Arguments

| | |
|---|---|
| `BMDFilMat` | A data frame containing BMD values for multiple genes. |
| `gi` | A vector of gene names for which to plot BMD values. |

### Value

A plot showing BMD, BMDL, and BMDU values for the selected set of genes.

---

## `plot_BMD_genes_in_set` *Plot BMD for Genes in Set*

---

### Description

This function generates a plot showing BMD values for a set of genes from the given data.

### Usage

```
plot_BMD_genes_in_set(
  BMD_TAB,
  enrichment_data,
  time_col,
  experiment_col = "Experiment",
  selectedrowindex
)
```

### Arguments

| | |
|---|---|
| `BMD_TAB` | A data frame containing BMD values. |
| `enrichment_data` | A data frame containing enrichment data. |
| `time_col` | Column containing time information. |
| `experiment_col` | Column containing experiment information. |
| `selectedrowindex` | Index of the selected row in the enrichment data. |

### Value

A plot showing BMD values for the selected set of genes.

---

## `plot_filtering_pie_chart` *Plot a pie chart of filtering results*

---

### Description

This function generates a pie chart to visualize the filtering results. The pie chart shows the distribution of variables and non-variables based on the filtering p-values in the table. The function takes the table and an p-value threshold as input.

### Usage

```
plot_filtering_pie_chart(tab_unfiltered, Pval.th)
```

## Arguments

| | |
|---|---|
| `tab_unfiltered` | The unfiltered table containing the filtering p-values. |
| `Pval.th` | The p-value threshold for determining variable significance. |

## Value

A pie chart visualization of the filtering results.

---

## `plot_gene_pairs` *Plot gene pairs comparison*

---

### Description

This function plots the comparison of two gene pairs given the models mod_i and mod_j and newdata containing dose information for new observations.

### Usage

```
plot_gene_pairs(mod_i, mod_j, newdata, main = "", feat1 = "g1", feat2 = "g2")
```

### Arguments

| | |
|---|---|
| `mod_i` | The model for gene pair 1. |
| `mod_j` | The model for gene pair 2. |
| `newdata` | A data frame containing dose information for new observations. |
| `main` | A character string specifying the main title of the plot (default is an empty string). |
| `feat1` | A character string specifying the label for gene pair 1 in the plot (default is "g1"). |
| `feat2` | A character string specifying the label for gene pair 2 in the plot (default is "g2"). |

### Value

A ggplot object showing the comparison of gene pairs.

---

## `plot_histogram` *Plot Histogram and Density*

---

### Description

Generate a histogram and density plot visualization based on provided data and variables.

### Usage

```
plot_histogram(
  mod_stats,
  y_val = "BMD",
  color_by = NULL,
  group_by = NULL,
  group_by2 = NULL,
  filter_column = NULL,
  filter_by = NULL,
  alpha_th = 0.5
)
```

### Arguments

| | |
|---|---|
| `mod_stats` | A data frame containing the data for plotting. |
| `y_val` | The variable to be plotted on the y-axis (numeric). |
| `color_by` | The categorical variable used for color-coding bars. |
| `group_by` | The grouping variable for additional data segmentation. |
| `group_by2` | The second grouping variable for further segmentation. |
| `filter_column` | The column name for filtering the data. |
| `filter_by` | The values used for filtering the data. |
| `alpha_th` | The transparency level of the density plot (ranges between 0-1). |

### Value

A ggplot2 histogram and density plot visualization.

---

## `plot_pie_chart` *this function allows to plot the histogram of one numeric column of the model statistics*

---

### Description

this function allows to plot the histogram of one numeric column of the model statistics

### Usage

```
plot_pie_chart(
  mod_stats,
  category = "Model",
  group_by = NULL,
  group_by2 = NULL,
  filter_column = NULL,
  filter_by = NULL
)
```

### Arguments

| | |
|---|---|
| `mod_stats` | A data frame containing the data for plotting. |
| `category` | The category variable to be used for the pie chart sectors. |
| `group_by` | The grouping variable for additional data segmentation. |
| `group_by2` | The second grouping variable for further segmentation. |
| `filter_column` | The column name for filtering the data. |
| `filter_by` | The values used for filtering the data. |

### Value

a ggplot object

---

## `plot_scatter` *Plot Scatter Plot*

---

### Description

Generate a scatter plot visualization based on provided data and variables.

### Usage

```
plot_scatter(
  mod_stats,
  x_val = "BMDL",
  y_val = "BMD",
  color_by = "Model",
  group_by = NULL,
  group_by2 = NULL,
  filter_column = NULL,
  filter_by = NULL
)
```

### Arguments

| | |
|---|---|
| `mod_stats` | A data frame containing the data for plotting. |
| `x_val` | The variable to be plotted on the x-axis. |
| `y_val` | The variable to be plotted on the y-axis. |
| `color_by` | The categorical variable used for color-coding points. |
| `group_by` | The grouping variable for additional data segmentation. |
| `group_by2` | The second grouping variable for further segmentation. |
| `filter_column` | The column name for filtering the data. |
| `filter_by` | The values used for filtering the data. |

### Value

A ggplot2 scatter plot visualization.

| | |
|---|---|
| `point_of_departure` | *Given a model, this function estimates the Benchmark Dose (BMD), Benchmark Dose Lower Confidence Limit (BMDL), Benchmark Dose Upper Confidence Limit (BMDU), and AC50 values.* |

## Description

Given a model, this function estimates the Benchmark Dose (BMD), Benchmark Dose Lower Confidence Limit (BMDL), Benchmark Dose Upper Confidence Limit (BMDU), and AC50 values.

## Usage

```
point_of_departure(
  model,
  deviation_type = "standard",
  rl = 1.349,
  confidence_interval = 0.95
)
```

## Arguments

| | |
|---|---|
| `model` | The model object representing the dose-response relationship. |
| `deviation_type` | Character string specifying the type of deviation from the fitted model to use for BMD calculation. Default is "standard". |
| `rl` | The relative level used to calculate the BMD. Default is 1.349. |
| `confidence_interval` | The confidence level for the confidence interval. Default is 0.95. |

## Value

A modified model object with additional attributes for BMDL, BMDU, and AC50. If an error occurs during the estimation, NA values are assigned to BMDL and BMDU.

| | |
|---|---|
| `pval_adjust` | *Adjust p-values in filtering results* |

## Description

This function adjusts the p-values in the filtering results using a specified adjustment method.

## Usage

```
pval_adjust(filtering_res, adjustment_method = "fdr")
```

## Arguments

| | |
|---|---|
| `filtering_res` | The filtering results table. |
| `adjustment_method` | The adjustment method for p-values. Options include "Nominal" (no adjustment) and methods supported by the `p.adjust` function. Default is "fdr" (false discovery rate). |

### Value

The filtering results table with adjusted p-values and the column "usedFilteringPval" indicating the p-values used for filtering.

---

## `read_excel_allsheets` *Read all sheets from an Excel file.*

---

### Description

This function reads all sheets from an Excel file specified by the `filename`. It returns the data as a list of data frames, one for each sheet.

### Usage

```
read_excel_allsheets(
  filename,
  tibble = FALSE,
  first_col_as_rownames = FALSE,
  is_rnaseq_raw_count = FALSE
)
```

### Arguments

| | |
|---|---|
| `filename` | The path to the Excel file. |
| `tibble` | If `TRUE`, the output data frames will be converted to tibbles. Default is `FALSE`. |
| `first_col_as_rownames` | If `TRUE`, the first column of each sheet will be used as row names. Default is `FALSE`. |
| `is_rnaseq_raw_count` | If `TRUE`, the data is assumed to be RNA-Seq raw counts, and it will be converted to log2 counts using the `limma::voom` function. Default is `FALSE`. |

### Value

A list of data frames, one for each sheet in the Excel file.

---

## `render_aop_fingerprint_bubble_plot` *Render AOP Fingerprint Bubble Plot*

---

### Description

This function generates a bubble plot for rendering AOP fingerprint data.

### Usage

```
render_aop_fingerprint_bubble_plot(
  enrichement_data,
  group_by,
  group_by2,
  time_var,
  filter_column,
  filter_by,
  is_group_by_numeric,
  threshold_proportion,
  text_cex = 12
)
```

### Arguments

| | |
|---|---|
| `enrichement_data` | A data frame containing the enrichment data. |
| `group_by` | Variable used for grouping the data. |
| `group_by2` | Second variable used for subgrouping the data. |
| `time_var` | Column name of timepoint variable |
| `filter_column` | Column for filtering the data. |
| `filter_by` | Value for filtering the data. |
| `is_group_by_numeric` | Logical, whether the grouping variable is numeric. |
| `threshold_proportion` | Proportion threshold for filtering data. |
| `text_cex` | Text size for labels. |

### Value

A bubble plot visualizing AOP fingerprint data.

---

## `render_range_plot` *Render Range Plot*

---

### Description

This function generates a range plot for rendering enrichment data.

### Usage

```
render_range_plot(
  enrichement_data,
  group_by,
  group_by2,
  filter_column,
  filter_by,
  is_group_by_numeric,
  display = "Ke"
)
```

## Arguments

| | |
|---|---|
| `enrichement_data` | A data frame containing the enrichment data. |
| `group_by` | Variable used for grouping the data. |
| `group_by2` | Second variable used for subgrouping the data. |
| `filter_column` | Column for filtering the data. |
| `filter_by` | Value for filtering the data. |
| `is_group_by_numeric` | Logical, whether the grouping variable is numeric. |
| `display` | Display option: "Ke" or "a.name". |

## Value

A range plot visualizing enrichment data.

---

`scale_numbers` *Scale Numbers to the Range 0-1 This function scales numeric values to the range 0-1 by dividing each value by the maximum value in the input vector.*

---

## Description

Scale Numbers to the Range 0-1 This function scales numeric values to the range 0-1 by dividing each value by the maximum value in the input vector.

## Usage

```
scale_numbers(x)
```

## Arguments

| | |
|---|---|
| `x` | A numeric vector to be scaled. |

## Value

A numeric vector with scaled values in the range 0-1.

## select_optimal_models *Select optimal models from a list of current models*

### Description

This function selects the optimal models from a list of current models based on the specified method. The supported methods are "AIC" (Akaike Information Criterion) and "Model average". For the "AIC" method, the model with the lowest AIC value is selected for each dataset. For the "Model average" method, if average models are already being computed, the average model is selected; otherwise, the first model in the list is selected. The function returns a list containing the optimal models and a table of computed model statistics.

### Usage

```
select_optimal_models(
  current_models,
  method = "AIC",
  time_col_id,
  optional_col_ids = NULL,
  nCores = 1
)
```

### Arguments

| | |
|---|---|
| `current_models` | The list of current models. |
| `method` | The method for selecting the optimal models. Supported values are "AIC" and "Model average". |
| `time_col_id` | The identifier for the time column in the model statistics table. |
| `optional_col_ids` | Optional identifiers for additional columns in the model statistics table. |
| `nCores` | The number of CPU cores to use for parallel computation. |

### Value

A list containing the optimal models and the computed model statistics table.

## sym_human_gene_graph *sym_human_gene_graph*

### Description

Graph whose nodes are Human gene symbols and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
sym_human_gene_graph
```

### Format

An object of class `igraph` of length 6664.

### Source

https://doi.org/10.1093/bib/bbaa417

## sym_mouse_gene_graph *sym_mouse_gene_graph*

### Description

Graph whose nodes are mouse gene symbols and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
sym_mouse_gene_graph
```

### Format

An object of class `igraph` of length 4649.

### Source

https://doi.org/10.1093/bib/bbaa417

## sym_rat_gene_graph *sym_rat_gene_graph*

### Description

Graph whose nodes are rat gene symbols and edges represent the connection between corresponding proteins or gene gene regulation process (e.g. transcription factors or mirna regulation)

### Usage

```
sym_rat_gene_graph
```

### Format

An object of class `igraph` of length 3227.

### Source

https://doi.org/10.1093/bib/bbaa417

## upset_plot *Create an UpSet Plot*

### Description

This function generates an UpSet plot for the given data, which displays the intersections of sets of genes across different experiments or conditions.

### Usage

```
upset_plot(
  mod_stats,
  rel_variable = "Experiment",
  group_by = NULL,
  other_variables = NULL,
  filter_column = c("Model"),
  filter_by = list(c("linear")),
  nintersects = 3,
  group.by = "degree",
  order.by = "degree"
)
```

### Arguments

| | |
|---|---|
| `mod_stats` | A data frame containing the model statistics and gene information. |
| `rel_variable` | The name of the variable representing the experiments or conditions. |
| `group_by` | The name of the variable to group the data for generating UpSet plots. |
| `other_variables` | Additional variables used for plotting, if any. |
| `filter_column` | The name of the column to filter the data. |
| `filter_by` | A list of filter values to apply on the specified filter_column. |
| `nintersects` | The number of intersections to show in the UpSet plot. |
| `group.by` | The variable used to group the intersections (e.g., "degree" or "freq"). |
| `order.by` | The variable used to order the intersections (e.g., "frequency", or "degree"). |

### Value

An UpSet plot displaying the intersections of sets of genes across different experiments or conditions.