

# Package ‘CEMiTool’

May 10, 2024

**Title** Co-expression Modules identification Tool

**Version** 1.28.0

**Description** The CEMiTool package unifies the discovery and the analysis of coexpression gene modules in a fully automatic manner, while providing a user-friendly html report with high quality graphs. Our tool evaluates if modules contain genes that are over-represented by specific pathways or that are altered in a specific sample group. Additionally, CEMiTool is able to integrate transcriptomic data with interactome information, identifying the potential hubs on each network.

**Depends** R (>= 4.0)

**Imports** methods, scales, dplyr, data.table (>= 1.9.4), WGCNA, grid, ggplot2, ggpmisc, ggthemes, ggrepel, sna, clusterProfiler, fgsea, stringr, knitr, rmarkdown, igraph, DT, htmltools, pracma, intergraph, grDevices, utils, network, matrixStats, ggdendro, gridExtra, gtable, fastcluster

**Suggests** testthat, BiocManager

**License** GPL-3

**Encoding** UTF-8

**biocViews** GeneExpression, Transcriptomics, GraphAndNetwork, mRNAArray, RNASeq, Network, NetworkEnrichment, Pathways, ImmunoOncology

**LazyData** true

**RoxygenNote** 7.1.2

**VignetteBuilder** knitr

**git\_url** <https://git.bioconductor.org/packages/CEMiTool>

**git\_branch** RELEASE\_3\_19

**git\_last\_commit** f232237

**git\_last\_commit\_date** 2024-04-30

**Repository** Bioconductor 3.19

**Date/Publication** 2024-05-10

**Author** Pedro Russo [aut],  
 Gustavo Ferreira [aut],  
 Matheus Bürger [aut],  
 Lucas Cardozo [aut],  
 Diogenes Lima [aut],  
 Thiago Hirata [aut],  
 Melissa Lever [aut],  
 Helder Nakaya [aut, cre]

**Maintainer** Helder Nakaya <hnakaya@usp.br>

## Contents

adj_data . . . . .	3
cem . . . . .	4
cemitool . . . . .	5
CEMiTool-class . . . . .	7
diagnostic_report . . . . .	8
expr0 . . . . .	9
expr_data . . . . .	10
expr_pct_filter . . . . .	11
filter_genes . . . . .	11
find_modules . . . . .	12
fit_data . . . . .	14
generate_report . . . . .	14
get_adj . . . . .	15
get_beta_data . . . . .	16
get_cemitool_r2_beta . . . . .	17
get_connectivity . . . . .	18
get_hubs . . . . .	19
get_merged_mods . . . . .	20
get_mods . . . . .	21
get_phi . . . . .	22
gsea_data . . . . .	22
interactions_data . . . . .	23
module_genes . . . . .	24
module_to_gmt . . . . .	25
mod_colors . . . . .	25
mod_gene_num . . . . .	26
mod_gsea . . . . .	27
mod_names . . . . .	28
mod_ora . . . . .	29
mod_summary . . . . .	30
new_cem . . . . .	30
nmodules . . . . .	32
ora_data . . . . .	32
plot_beta_r2 . . . . .	33
plot_gsea . . . . .	34

plot_hist . . . . .	35
plot_interactions . . . . .	36
plot_mean_k . . . . .	37
plot_mean_var . . . . .	37
plot_ora . . . . .	38
plot_ora_single . . . . .	39
plot_profile . . . . .	40
plot_qq . . . . .	41
plot_sample_tree . . . . .	41
read_gmt . . . . .	42
sample_annot . . . . .	43
sample_annotation . . . . .	44
save_plots . . . . .	45
select_genes . . . . .	46
show,CEMiTool-method . . . . .	47
show_plot . . . . .	47
vst . . . . .	48
write_files . . . . .	48
<b>Index</b>	<b>50</b>

---

adj_data	<i>Get or set adjacency matrix value</i>
----------	------------------------------------------

---

## Description

This function takes a CEMiTool object containing expression values and returns a CEMiTool object with an adjacency matrix in the adjacency slot.

## Usage

```
adj_data(cem, ...)
```

```
## S4 method for signature 'CEMiTool'
```

```
adj_data(cem)
```

```
adj_data(cem) <- value
```

```
## S4 replacement method for signature 'CEMiTool'
```

```
adj_data(cem) <- value
```

## Arguments

cem	Object of class CEMiTool
...	Optional parameters.
value	Object of class matrix containing adjacency data. Only used for setting adjacency values to CEMiTool object.

**Value**

Object of class matrix with adjacency values or object of class CEMiTool.

**Examples**

```
# Get example expression data
data(expr0)
# Initialize new CEMiTool object with expression
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)
# Calculate adjacency matrix with example beta value 8
cem <- get_adj(cem, beta=8)
# Return adjacency matrix
adj <- adj_data(cem)
# Check result
adj[1:5, 1:5]
# Set adjacency matrix to CEMiTool object
adj_data(cem) <- adj
```

---

cem

*CEMiTool Object*

---

**Description**

This object can be used as input for CEMiTool functions. Data used are from expr and sample\_annot.

**Usage**

```
data(cem)
```

**Format**

An object of class CEMiTool

**Examples**

```
# Get example CEMiTool object
data(cem)
cem
```

---

cemitool	<i>Full gene co-expression analysis</i>
----------	-----------------------------------------

---

**Description**

Defines co-expression modules and runs several different analyses.

**Usage**

```
cemitool(  
  expr,  
  annot,  
  gmt,  
  interactions,  
  filter = TRUE,  
  filter_pval = 0.1,  
  apply_vst = FALSE,  
  n_genes,  
  eps = 0.1,  
  cor_method = c("pearson", "spearman"),  
  cor_function = "cor",  
  network_type = "unsigned",  
  tom_type = "signed",  
  set_beta = NULL,  
  force_beta = FALSE,  
  sample_name_column = "SampleName",  
  class_column = "Class",  
  merge_similar = TRUE,  
  rank_method = "mean",  
  ora_pval = 0.05,  
  gsea_scale = TRUE,  
  gsea_min_size = 15,  
  gsea_max_size = 1000,  
  min_nngen = 30,  
  diss_thresh = 0.8,  
  plot = TRUE,  
  plot_diagnostics = TRUE,  
  order_by_class = TRUE,  
  center_func = "mean",  
  directed = FALSE,  
  verbose = FALSE  
)
```

**Arguments**

expr	Gene expression data.frame.
annot	Sample annotation data.frame.

gmt	A data.frame containing two columns, one with pathways and one with genes
interactions	A data.frame containing two columns with gene names.
filter	Logical. If TRUE, will filter expression data.
filter_pval	P-value threshold for filtering. Default 0.1.
apply_vst	Logical. If TRUE, will apply Variance Stabilizing Transform before filtering genes. Currently ignored if parameter filter is FALSE.
n_genes	Number of genes left after filtering.
eps	A value for accepted R-squared interval between subsequent beta values. Default is 0.1.
cor_method	A character string indicating which correlation coefficient is to be computed. One of "pearson" or "spearman". Default is "pearson".
cor_function	A character string indicating the correlation function to be used. Supported functions are currently 'cor' and 'bicor'. Default is "cor"
network_type	A character string indicating if network type should be computed as "signed" or "unsigned". Default is "unsigned"
tom_type	A character string indicating if the TOM type should be computed as "signed" or "unsigned". Default is "signed"
set_beta	A value to override the automatically selected beta value. Default is NULL.
force_beta	Whether or not to automatically force a beta value based on number of samples. Default is FALSE.
sample_name_column	A character string indicating the sample column name of the annotation table.
class_column	A character string indicating the class column name of the annotation table.
merge_similar	Logical. If TRUE, merge similar modules.
rank_method	Character string indicating how to rank genes. Either "mean" (the default) or "median".
ora_pval	P-value for overrepresentation analysis. Default 0.05.
gsea_scale	If TRUE, apply z-score transformation for GSEA analysis. Default is TRUE
gsea_min_size	Minimum size of gene sets for GSEA analysis. Default is 15
gsea_max_size	Maximum size of gene sets for GSEA analysis. Default is 1000
min_nngen	Minimal number of genes per submodule. Default 30.
diss_thresh	Module merging correlation threshold for eigengene similarity. Default 0.8.
plot	Logical. If TRUE, plots all figures.
plot_diagnostics	Logical. If TRUE, creates diagnostic plots. Overwritten if plot=FALSE.
order_by_class	Logical. If TRUE, samples in profile plot are ordered by the groups defined by the class_column slot in the sample annotation file. Ignored if there is no sample_annotation file. Default TRUE.
center_func	Character string indicating the centrality measure to show in the plot. Either 'mean' (the default) or 'median'.
directed	Logical. If TRUE, the igraph objects in interactions slot will be directed.
verbose	Logical. If TRUE, reports analysis steps.

**Value**

Object of class CEMiTool

**Examples**

```
# Get example expression data
data(expr0)
# Run CEMiTool analyses
cem <- cemitool(expr=expr0)
# Run CEMiTool applying Variance Stabilizing Transformation to data
cem <- cemitool(expr=expr0, apply_vst=TRUE)
# Run CEMiTool with additional processing messages
cem <- cemitool(expr=expr0, verbose=TRUE)

## Not run:
# Run full CEMiTool analysis
## Get example sample annotation data
data(sample_annot)
## Read example pathways file
gmt_fname <- system.file("extdata", "pathways.gmt", package = "CEMiTool")
gmt_in <- read_gmt(gmt_fname)
## Get example interactions file
int_df <- read.delim(system.file("extdata", "interactions.tsv", package = "CEMiTool"))
## Run CEMiTool
cem <- cemitool(expr=expr0, annot=sample_annot, gmt=gmt_in,
               interactions=int_df, verbose=TRUE, plot=TRUE)

# Create report as html file
generate_report(cem, directory = "./Report", output_format="html_document")

# Write analysis results into files
write_files(cem, directory="./Tables", force=TRUE)

# Save all plots
save_plots(cem, "all", directory="./Plots")

## End(Not run)
```

---

CEMiTool-class

*An S4 class to represent the CEMiTool analysis.*

---

**Description**

An S4 class to represent the CEMiTool analysis.

**Slots**

expression Gene expression data.frame.  
 sample\_annotation Sample annotation data.frame.

`fit_indices` `data.frame` containing scale-free model fit, soft-threshold and network parameters.  
`selected_genes` Character vector containing the names of genes selected for analysis  
`module` Genes in modules information `data.frame`.  
`enrichment` `list` with modules enrichment results for sample classes.  
`ora` Over-representation analysis results `data.frame`.  
`interactions` `list` containing gene interactions present in modules.  
`interaction_plot` `list` of `ggplot` graphs with module gene interactions.  
`profile_plot` `list` of `ggplot` graphs with gene expression profile per module.  
`enrichment_plot` `ggplot` graph for enrichment analysis results.  
`beta_r2_plot` `ggplot` graph with scale-free topology fit results for each soft-threshold.  
`mean_k_plot` `ggplot` graph with mean network connectivity.  
`barplot_ora` `list` of `ggplot` graphs with over-representation analysis results per module.  
`sample_tree_plot` `gtable` containing sample dendrogram with class labels and clinical data (if available in `sample_annotation(cem)`).  
`mean_var_plot` Mean x variance scatterplot.  
`hist_plot` Expression histogram.  
`qq_plot` Quantile-quantile plot.  
`sample_name_column` character string containing the name of the column with sample names in the annotation file.  
`class_column` character string containing the name of the column with class names in the annotation file.  
`mod_colors` character vector containing colors associated with each network module.  
`parameters` `list` containing analysis parameters.  
`adjacency` `matrix` containing gene adjacency values based on correlation

### Examples

```

# Get example expression data
data(expr0)
# Initialize CEMiTool object with expression
cem <- new("CEMiTool", expression=expr0)

```

---

diagnostic\_report      *Diagnostic report*

---

### Description

Creates report for identifying potential problems with data.



**Usage**

```

diagnostic_report(cem, ...)

## S4 method for signature 'CEMiTool'
diagnostic_report(
  cem,
  title = "Diagnostics",
  directory = "./Reports/Diagnostics",
  force = FALSE,
  ...
)

```

**Arguments**

cem	Object of class CEMiTool.
...	parameters to rmarkdown::render
title	Character string with the title of the report.
directory	Directory name for results.
force	If the directory exists, execution will not stop.

**Value**

An HTML file with an interactive diagnostic report.

---

 expr0

*Yellow Fever gene expression data from GEO study GSE13485*

---

**Description**

Modified data from a yellow fever vaccination study by Querec et al, 2009. In order to reduce package size, only the 4000 genes with the highest variance were selected for this dataset.

**Usage**

```
data(expr0)
```

**Format**

An object of class data.frame

**Source**

**GEO**

## References

Querec TD, Akondy RS, Lee EK, Cao W et al. Systems biology approach predicts immunogenicity of the yellow fever vaccine in humans. *Nat Immunol* 2009 Jan;10(1):116-25. PMID: 19029902  
[PubMed](#)

## Examples

```
data(expr0)
# Run CEMiTool analysis
## Not run: cemitoool(expr0)
```

---

expr_data	<i>Retrieve and set expression attribute</i>
-----------	----------------------------------------------

---

## Description

Retrieve and set expression attribute

## Usage

```
expr_data(cem, ...)
```

## S4 method for signature 'CEMiTool'

```
expr_data(cem, filter = TRUE, apply_vst = FALSE, filter_pval = 0.1, ...)
```

```
expr_data(cem) <- value
```

## S4 replacement method for signature 'CEMiTool'

```
expr_data(cem) <- value
```

## Arguments

cem	Object of class CEMiTool
...	Additional parameters to filter_genes or select_genes functions.
filter	logical. If TRUE, retrieves filtered expression data (Default: TRUE)
apply_vst	logical. If TRUE, applies variance stabilizing transformation to expression data (Default: FALSE)
filter_pval	logical. Threshold for filter p-value. Ignored if filter = FALSE (Default: 0.1)
value	Object of class data.frame with gene expression data

## Value

Object of class data.frame with gene expression data

**Examples**

```
# Initialize an empty CEMiTool object
cem <- new_cem()
# Get example expression data
data(expr0)
# Add expression file to CEMiTool object
expr_data(cem) <- expr0
# Check expression file
head(expr_data(cem))
```

---

expr_pct_filter	<i>Filter genes based on expression.</i>
-----------------	------------------------------------------

---

**Description**

Filter genes based on expression.

**Usage**

```
expr_pct_filter(expr, pct = 0.75)
```

**Arguments**

expr	expression file containing genes in the rows and samples in the columns
pct	percentage of most expressed genes to maintain

**Value**

A data.frame containing the results

---

filter_genes	<i>Filter a gene expression data.frame</i>
--------------	--------------------------------------------

---

**Description**

Filter a gene expression data.frame

**Usage**

```
filter_genes(expr, pct = 0.75, apply_vst = FALSE)
```

**Arguments**

expr	A data.frame containing expression data
pct	Percentage of most expressed genes to keep.
apply_vst	Logical. If TRUE, will apply variance stabilizing transform before filtering data.

**Details**

This function takes a gene expression data.frame and applies a percentage filter (keeps the pct) most expressed genes). If apply\_vst is TRUE, a variance stabilizing transformation is applied on gene expression values as long as mean and variance values have a Spearman's rho of over 0.5. This transformation is intended to remove this dependence between the parameters. One should then apply the select\_genes function to get significant genes.

**Value**

A data.frame containing filtered expression data

**Examples**

```
# Get example expression data
data(expr0)
# Filter genes
expr_f <- filter_genes(expr0)
# Check selected genes
expr_f[1:5, 1:5]
# Filter genes and apply variance stabilizing transformation
expr_f2 <- filter_genes(expr0, apply_vst=TRUE)
# Check results
expr_f2[1:5, 1:5]
# Selected genes
selected <- select_genes(expr_f2)
# Get data.frame with only selected genes
expr_s <- expr_f2[selected, ]
# Check results
expr_s[1:5, 1:5]
```

---

find\_modules

*Co-expression modules definition*

---

**Description**

Defines co-expression modules

**Usage**

```
find_modules(cem, ...)
```

```
## S4 method for signature 'CEMiTool'
find_modules(
  cem,
  cor_method = c("pearson", "spearman"),
  cor_function = "cor",
  eps = 0.1,
  set_beta = NULL,
```

```

    force_beta = FALSE,
    min_nngen = 20,
    merge_similar = TRUE,
    network_type = "unsigned",
    tom_type = "signed",
    diss_thresh = 0.8,
    verbose = FALSE
  )

```

### Arguments

<code>cem</code>	Object of class CEMiTool.
<code>...</code>	Optional parameters.
<code>cor_method</code>	A character string indicating which correlation coefficient is to be computed. Default "pearson"
<code>cor_function</code>	A character string indicating the correlation function to be used. Default 'cor'.
<code>eps</code>	A value for accepted R-squared interval between subsequent beta values. Default is 0.1.
<code>set_beta</code>	A value to override the automatically selected beta value. Default is NULL.
<code>force_beta</code>	Whether or not to automatically force a beta value based on number of samples. Default is FALSE.
<code>min_nngen</code>	Minimal number of genes per submodule. Default 20.
<code>merge_similar</code>	Logical. If TRUE, (the default) merge similar modules.
<code>network_type</code>	A character string indicating to use either "unsigned" (default) or "signed" networks. Default "unsigned"
<code>tom_type</code>	A character string indicating to use either "unsigned" or "signed" (default) TOM similarity measure.
<code>diss_thresh</code>	Module merging correlation threshold for eigengene similarity. Default 0.8.
<code>verbose</code>	Logical. If TRUE, reports analysis steps. Default FALSE

### Value

Object of class CEMiTool

### Examples

```

# Get example expression data
data(expr0)
# Initialize CEMiTool object with expression
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)
# Define network modules
cem <- find_modules(cem)
# Check results
head(module_genes(cem))

```

---

fit_data	<i>Retrieve scale-free model fit data</i>
----------	-------------------------------------------

---

**Description**

Retrieve scale-free model fit data

**Usage**

```
fit_data(cem)

## S4 method for signature 'CEMiTool'
fit_data(cem)
```

**Arguments**

cem                    Object of class CEMiTool

**Value**

Object of class data.frame

**Examples**

```
# Get example CEMiTool object
data(cem)
# Get modules and beta data
cem <- find_modules(cem)
# Get fit data
fit_data(cem)
```

---

generate_report	<i>CEMiTool report</i>
-----------------	------------------------

---

**Description**

Creates report for CEMiTool results

**Usage**

```
generate_report(cem, ...)

## S4 method for signature 'CEMiTool'
generate_report(
  cem,
  max_rows_ora = 50,
  title = "Report",
  directory = "./Reports/Report",
  force = FALSE,
  ...
)
```

**Arguments**

cem	Object of class CEMiTool.
...	parameters to rmarkdown::render
max_rows_ora	maximum number of rows in Over Representation Analysis table results
title	Character string with the title of the report.
directory	Directory name for results.
force	If the directory exists, execution will not stop.

**Value**

An HTML file with an interactive report of CEMiTool analyses.

**Examples**

```
## Not run:
# Get example CEMiTool object
data(cem)
generate_report(cem, output_format=c("pdf_document", "html_document"))

## End(Not run)
```

---

get\_adj

*Calculate adjacency matrix*

---

**Description**

This function takes a CEMiTool object and returns an adjacency matrix.

**Usage**

```
get_adj(cem, ...)

## S4 method for signature 'CEMiTool'
get_adj(
  cem,
  beta,
  network_type = "unsigned",
  cor_function = "cor",
  cor_method = "pearson"
)
```

**Arguments**

cem	Object of class CEMiTool
...	Optional parameters.
beta	Selected soft-threshold value
network_type	A character string indicating to use either "unsigned" (default) or "signed" networks. Default "unsigned".
cor_function	A character string indicating the correlation function to be used. Default 'cor'.
cor_method	A character string indicating which correlation coefficient is to be computed. Default "pearson".

**Value**

Object of class CEMiTool with adjacency data

**Examples**

```
# Get example expression data
data(expr0)
# Initialize new CEMiTool object with expression data
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)
# Calculate adjacency matrix with example beta value 8
cem <- get_adj(cem, beta=8)
# Check results
adj <- adj_data(cem)
adj[1:5, 1:5]
```

---

get\_beta\_data

*Soft-threshold beta data*


---

**Description**

This function takes the input parameters from find\_modules and calculates the WGCNA soft-threshold parameters and returns them.



**Usage**

```
get_beta_data(cem, ...)

## S4 method for signature 'CEMiTool'
get_beta_data(
  cem,
  network_type = "unsigned",
  cor_function = "cor",
  cor_method = "pearson",
  verbose = FALSE
)
```

**Arguments**

cem	A CEMiTool object containing expression data
...	Optional parameters.
network_type	A character string indicating to use either "unsigned" (default) or "signed" networks. Default "unsigned".
cor_function	A character string indicating the correlation function to be used. Default 'cor'.
cor_method	A character string indicating which correlation coefficient is to be computed. Default "pearson"
verbose	Logical. If TRUE, reports analysis steps. Default FALSE

**Value**

A list containing the soft-threshold selected by WGCNA and scale-free model parameters

**Examples**

```
# Get example expression data
data(expr0)
# Initialize new CEMiTool object with expression data
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)
# Get beta data
beta_data <- get_beta_data(cem)
```

---

get\_cemitoool\_r2\_beta *Calculate CEMiTool beta and R2 values*

---

**Description**

This function takes a CEMiTool object with beta data and returns a vector containing the chosen beta and corresponding R squared value.

**Usage**

```
get_cemitool_r2_beta(cem, ...)

## S4 method for signature 'CEMiTool'
get_cemitool_r2_beta(cem, eps = 0.1)
```

**Arguments**

cem	A CEMiTool object containing the fit_indices slot
...	Optional parameters.
eps	A value indicating the accepted interval between successive values of R squared to use to calculate the selected beta. Default: 0.1.

**Value**

A vector containing R squared value and the chosen beta parameter.

**Examples**

```
# Get example expression data
data(expr0)
# Initialize new CEMiTool object with expression data
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)
# Get modules and beta data
cem <- find_modules(cem)
# Get CEMiTool R2 and beta values
get_cemitool_r2_beta(cem)
```

---

get_connectivity	<i>Calculate network connectivity</i>
------------------	---------------------------------------

---

**Description**

This function takes a CEMiTool object and returns the network connectivity.

**Usage**

```
get_connectivity(cem, ...)

## S4 method for signature 'CEMiTool'
get_connectivity(cem, beta)
```

**Arguments**

cem	Object of class CEMiTool containing the fit_indices slot
...	Optional parameters.
beta	A soft-thresholding value to be used for the network.

**Value**

The value of the network's connectivity.

**Examples**

```
# Get example expression data
data(expr0)
# Initialize new CEMiTool object with expression data
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)
# Get modules and beta data
cem <- find_modules(cem)
# Get network connectivity with example beta value 8
get_connectivity(cem, beta=8)
```

---

get\_hubs

*Get hubs*


---

**Description**

Returns n genes in each module with high connectivity.

**Usage**

```
get_hubs(cem, ...)

## S4 method for signature 'CEMiTool'
get_hubs(cem, n = 5, method = "adjacency")
```

**Arguments**

cem	Object of class CEMiTool.
...	Optional parameters.
n	Number of hubs to return from each module. If "all", returns all genes in decreasing order of connectivity. Default: 5.
method	Method for hub calculation. Either "adjacency" or "kME". Default: "adjacency"

**Value**

A list containing hub genes for each module and the value of the calculated method.

**Examples**

```
# Get example CEMiTool object
data(cem)
# Get module hubs
hubs <- get_hubs(cem, n=10, "adjacency")
```

---

get\_merged\_mods      *Merge similar modules*

---

### Description

This function takes a CEMiTool object with expression and a vector of numeric labels to merge similar modules.

### Usage

```
get_merged_mods(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
get_merged_mods(cem, mods, diss_thresh = 0.8, verbose = FALSE)
```

### Arguments

cem	Object of class CEMiTool.
...	Optional parameters.
mods	A vector containing numeric labels for each module gene
diss_thresh	A threshold of dissimilarity for modules. Default is 0.8.
verbose	Logical. If TRUE, reports analysis steps. Default FALSE

### Value

Numeric labels assigning genes to modules.

### Examples

```
# Get example expression data  
data(expr0)  
# Initialize new CEMiTool object with expression data  
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)  
# Calculate adjacency matrix with example beta value 8  
cem <- get_adj(cem, beta=8)  
# Get modules  
mods <- get_mods(cem)  
# Merge similar modules  
merged_mods <- get_merged_mods(cem, mods)
```

---

get_mods	<i>Calculate co-expression modules</i>
----------	----------------------------------------

---

### Description

This function takes a CEMiTool object containing an adjacency matrix together with the given network parameters, and returns the given co-expression modules.

### Usage

```
get_mods(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
get_mods(  
  cem,  
  cor_function = "cor",  
  cor_method = "pearson",  
  tom_type = "signed",  
  min_nngen = 20  
)
```

### Arguments

cem	Object of class CEMiTool.
...	Optional parameters.
cor_function	A character string indicating the correlation function to be used. Default 'cor'.
cor_method	A character string indicating which correlation coefficient is to be computed. Default "pearson".
tom_type	A character string indicating to use either "unsigned" or "signed" (default) TOM similarity measure.
min_nngen	Minimal number of genes per module (Default: 20).

### Value

Numeric labels assigning genes to modules.

### Examples

```
# Get example expression data  
data(expr0)  
# Initialize new CEMiTool object with expression data  
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)  
# Calculate adjacency matrix with example beta value 8  
cem <- get_adj(cem, beta=8)  
# Get module labels  
mods <- get_mods(cem)
```

---

get_phi	<i>Calculate phi</i>
---------	----------------------

---

**Description**

This function takes a CEMiTool object and returns the phi parameter.

**Usage**

```
get_phi(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
get_phi(cem)
```

**Arguments**

cem	A CEMiTool object containing the fit_indices slot
...	Optional parameters.

**Value**

The phi parameter

**Examples**

```
# Get example expression data  
data(expr0)  
# Initialize new CEMiTool object with expression data  
cem <- new_cem(expr0, filter=TRUE, apply_vst=FALSE)  
# Get modules and beta data  
cem <- find_modules(cem)  
# Get phi  
get_phi(cem)
```

---

gsea_data	<i>Retrieve Gene Set Enrichment Analysis (GSEA) results</i>
-----------	-------------------------------------------------------------

---

**Description**

Retrieve Gene Set Enrichment Analysis (GSEA) results

**Usage**

```
gsea_data(cem)  
  
## S4 method for signature 'CEMiTool'  
gsea_data(cem)
```

**Arguments**

cem                    Object of class CEMiTool

**Value**

Object of class list with GSEA data

**Examples**

```
# Get example CEMiTool object
data(cem)
# Look at example annotation file
sample_annotation(cem)
# Run GSEA on network modules
cem <- mod_gsea(cem)
# Check results
gsea_data(cem)
```

---

interactions\_data        *Retrieve and set interaction data to CEMiTool object*

---

**Description**

Retrieve and set interaction data to CEMiTool object

**Usage**

```
interactions_data(cem, ...)
```

```
## S4 method for signature 'CEMiTool'
interactions_data(cem)
```

```
interactions_data(cem) <- value
```

```
## S4 replacement method for signature 'CEMiTool'
interactions_data(cem) <- value
```

**Arguments**

cem                    Object of class CEMiTool.  
...                    parameters for `igraph::graph_from_data_frame`  
value                  a data.frame or matrix containing two columns

**Value**

Object of class CEMiTool

## Examples

```
# Get example CEMiTool object
data(cem)
# Read example interactions data
int_df <- read.delim(system.file("extdata", "interactions.tsv",
  package = "CEMiTool"))
# Insert interactions data
interactions_data(cem) <- int_df
# Check interactions data
interactions_data(cem)
```

---

module\_genes

*Get the module genes in a CEMiTool object*

---

## Description

Get the module genes in a CEMiTool object

## Usage

```
module_genes(cem, module = NULL)

## S4 method for signature 'CEMiTool'
module_genes(cem, module = NULL)
```

## Arguments

cem	Object of class CEMiTool
module	A character string with the name of the module of which genes are to be returned. Defaults to NULL, which returns the full list of genes and modules.

## Value

Object of class data.frame containing genes and their respective module

## Examples

```
# Get example CEMiTool object
data(cem)
# Get the module genes
module_genes(cem)
# Get genes for module M1
module_genes(cem, module="M1")
```



---

module_to_gmt	<i>Transform module genes list to a gmt file</i>
---------------	--------------------------------------------------

---

**Description**

Transform module genes list to a gmt file

**Usage**

```
module_to_gmt(cem, directory = "../Tables")
```

**Arguments**

cem

**Value**

A .gmt file containing module genes in each row

---

mod_colors	<i>Retrieve and set mod_colors attribute</i>
------------	----------------------------------------------

---

**Description**

Retrieve and set mod\_colors attribute

**Usage**

```
mod_colors(cem)

## S4 method for signature 'CEMiTool'
mod_colors(cem)

mod_colors(cem) <- value

## S4 replacement method for signature 'CEMiTool,character'
mod_colors(cem) <- value
```

**Arguments**

cem	Object of class CEMiTool
value	a character vector containing colors for each module. Names should match with module names

**Value**

A vector with color names.

**Examples**

```
# Get example CEMiTool object
data(cem)
# See module colors
mod_colors(cem)
```

---

mod_gene_num	<i>Get the number of genes in modules in a CEMiTool object</i>
--------------	----------------------------------------------------------------

---

**Description**

Get the number of genes in modules in a CEMiTool object

**Usage**

```
mod_gene_num(cem, module = NULL)

## S4 method for signature 'CEMiTool'
mod_gene_num(cem, module = NULL)
```

**Arguments**

cem	Object of class CEMiTool
module	Default is NULL. If a character string designating a module is given, the number of genes in that module is returned instead.

**Value**

The number of genes in module(s)

**Examples**

```
# Get example CEMiTool object
data(cem)
# Get the number of genes in modules
mod_gene_num(cem)
# Get the number of genes in module M1
mod_gene_num(cem, "M1")
```

---

mod_gsea	<i>Module Gene Set Enrichment Analysis</i>
----------	--------------------------------------------

---

### Description

Performs Gene Set Enrichment Analysis (GSEA) for each co-expression module found.

### Usage

```
mod_gsea(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
mod_gsea(  
  cem,  
  gsea_scale = TRUE,  
  rank_method = "mean",  
  gsea_min_size = 15,  
  gsea_max_size = 1000,  
  verbose = FALSE  
)
```

### Arguments

cem	Object of class CEMiTool.
...	Optional parameters.
gsea_scale	If TRUE, transform data using z-score transformation. Default: TRUE
rank_method	Character string indicating how to rank genes. Either "mean" (the default) or "median".
gsea_min_size	Minimum gene set size (Default: 15).
gsea_max_size	Maximum gene set size (Default: 1000).
verbose	logical. Report analysis steps.

### Value

GSEA results.

### See Also

[plot\\_gsea](#)

**Examples**

```
# Get example CEMiTool object
data(cem)
# Look at example annotation file
sample_annotation(cem)
# Run GSEA on network modules
cem <- mod_gsea(cem)
# Check results
gsea_data(cem)
```

---

mod\_names

*Get module names in a CEMiTool object*

---

**Description**

Get module names in a CEMiTool object

**Usage**

```
mod_names(cem, include_NC = TRUE)

## S4 method for signature 'CEMiTool'
mod_names(cem, include_NC = TRUE)
```

**Arguments**

cem	Object of class CEMiTool
include_NC	Logical. Whether or not to include "Not.Correlated" module. Defaults to TRUE.

**Value**

Module names

**Examples**

```
# Get example CEMiTool object
data(cem)
# Get module names
mod_names(cem)
```

---

`mod_ora`*Module Overrepresentation Analysis*

---

**Description**

Performs overrepresentation analysis for each co-expression module found.

**Usage**

```
mod_ora(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
mod_ora(cem, gmt, verbose = FALSE)
```

**Arguments**

<code>cem</code>	Object of class CEMiTool.
<code>...</code>	Optional parameters.
<code>gmt</code>	Object of class <code>data.frame</code> with 2 columns, one with pathways and one with genes
<code>verbose</code>	logical. Report analysis steps.

**Value**

Object of class CEMiTool

**See Also**

[ora\\_data](#)

**Examples**

```
# Get example CEMiTool object  
data(cem)  
# Read gmt file  
gmt <- read_gmt(system.file('extdata', 'pathways.gmt',  
                           package='CEMiTool'))  
# Run module overrepresentation analysis  
cem <- mod_ora(cem, gmt)  
# Check results  
head(ora_data(cem))
```

---

`mod_summary`*Co-expression module summarization*

---

**Description**

Summarizes modules using mean or eigengene expression.

**Usage**

```
mod_summary(cem, ...)
```

```
## S4 method for signature 'CEMiTool'  
mod_summary(cem, method = c("mean", "median", "eigengene"), verbose = FALSE)
```

**Arguments**

<code>cem</code>	Object of class CEMiTool.
<code>...</code>	Optional parameters.
<code>method</code>	A character string indicating which summarization method is to be used. Can be 'eigengene', 'mean' or 'median'. Default is 'mean'.
<code>verbose</code>	Logical. If TRUE, reports analysis steps.

**Value**

A data.frame with summarized values.

**Examples**

```
# Get example CEMiTool object  
data(cem)  
# Summarize results  
mod_summary <- mod_summary(cem)
```

---

`new_cem`*Create a CEMiTool object*

---

**Description**

Create a CEMiTool object

**Usage**

```
new_cem(
  expr = data.frame(),
  sample_annot = data.frame(),
  sample_name_column = "SampleName",
  class_column = "Class",
  filter = TRUE,
  apply_vst = FALSE,
  filter_pval = 0.1
)
```

**Arguments**

<code>expr</code>	Object of class <code>data.frame</code> with gene expression data
<code>sample_annot</code>	Object of <code>data.frame</code> containing the sample annotation. It should have at least two columns containing group <code>Class</code> and the Sample Name that should match with samples in expression file
<code>sample_name_column</code>	A string specifying the column to be used as sample identification. Default: "SampleName".
<code>class_column</code>	A string specifying the column to be used as a grouping factor for samples. Default: "Class"
<code>filter</code>	Logical. Used to define if posterior functions should use filtered expression data or not (Default: TRUE)
<code>apply_vst</code>	Logical. Used to define if posterior functions should use a variance stabilizing transformation on expression data before analyses. Only valid if argument <code>filter</code> is TRUE. (Default: FALSE)
<code>filter_pval</code>	logical. Threshold for filter p-value. Ignored if <code>filter = FALSE</code> (Default: 0.1)

**Value**

Object of class `CEMiTool`

**Examples**

```
# Create new CEMiTool object
cem <- new_cem()
# Create new CEMiTool object with expression and sample_annotation data
data(expr0)
data(sample_annot)
cem <- new_cem(expr0, sample_annot, "SampleName", "Class")
# Equivalent to a call to new()
cem2 <- new("CEMiTool", expression=expr0, sample_annotation=sample_annot)
identical(cem, cem2)
```

---

nmodules	<i>Get the number of modules in a CEMiTool object</i>
----------	-------------------------------------------------------

---

**Description**

Get the number of modules in a CEMiTool object

**Usage**

```
nmodules(cem)

## S4 method for signature 'CEMiTool'
nmodules(cem)
```

**Arguments**

cem                    Object of class CEMiTool

**Value**

number of modules

**Examples**

```
# Get example CEMiTool object
data(cem)
# Get the number of modules
nmodules(cem)
```

---

ora_data	<i>Retrieve over representation analysis (ORA) results</i>
----------	------------------------------------------------------------

---

**Description**

Retrieve over representation analysis (ORA) results

**Usage**

```
ora_data(cem)

## S4 method for signature 'CEMiTool'
ora_data(cem)
```

**Arguments**

cem                    Object of class CEMiTool



## Details

This function returns the results of the `mod_ora` function on the `CEMiTool` object. The ID column corresponds to pathways in the `gmt` file for which genes in the modules were enriched. The Count column shows the number of genes in the module that are enriched for each pathway. The `GeneRatio` column shows the proportion of genes in the module enriched for a given pathway out of all the genes in the module enriched for any given pathway. The `BgRatio` column shows the proportion of genes in a given pathway out of all the genes in the `gmt` file. For more details, please refer to the `clusterProfiler` package documentation.

## Value

Object of class `data.frame` with ORA data

## References

Guangchuang Yu, Li-Gen Wang, Yanyan Han, Qing-Yu He. `clusterProfiler`: an R package for comparing biological themes among gene clusters. *OMICS: A Journal of Integrative Biology*. 2012, 16(5):284-287.

## Examples

```
# Get example CEMiTool object
data(cem)
# Read gmt file
gmt <- read_gmt(system.file('extdata', 'pathways.gmt',
                           package='CEMiTool'))
# Run module overrepresentation analysis
cem <- mod_ora(cem, gmt)
# Check results
head(ora_data(cem))
```

---

plot\_beta\_r2

*Soft-threshold beta selection graph*

---

## Description

Creates a graph showing each possible soft-threshold value and its corresponding R squared value

## Usage

```
plot_beta_r2(cem, ...)
```

```
## S4 method for signature 'CEMiTool'
plot_beta_r2(cem, plot_title = "Scale independence (beta selection)")
```

**Arguments**

cem                    Object of class CEMiTool.  
...                    Optional parameters.  
plot\_title            title of the graph

**Value**

Object of class CEMiTool with beta x R squared plot

**Examples**

```
# Get example CEMiTool object
data(cem)
# Plot scale-free model fit as a function of the soft-thresholding beta parameter choice
cem <- plot_beta_r2(cem)
# Check resulting plot
show_plot(cem, "beta_r2")
```

---

plot\_gsea

*GSEA visualization*

---

**Description**

Creates a heatmap with the results of gene set enrichment analysis (GSEA) of co-expression modules

**Usage**

```
plot_gsea(cem, ...)
```

## S4 method for signature 'CEMiTool'

```
plot_gsea(cem, pv_cut = 0.05)
```

**Arguments**

cem                    Object of class CEMiTool.  
...                    Optional parameters.  
pv\_cut                P-value cut-off. Default 0.05

**Value**

Object of class CEMiTool with GSEA plots

**Examples**

```
# Get example CEMiTool object
data(cem)
# Get example sample annotation file
# Run GSEA on network modules
cem <- mod_gsea(cem)
# Plot GSEA results
cem <- plot_gsea(cem)
# Check resulting plot
show_plot(cem, "gsea")
```

---

plot\_hist

*Plot histogram*

---

**Description**

This function plots a histogram of the distribution of gene expression, to help assess the normality of the data.

**Usage**

```
plot_hist(cem, ...)
```

```
## S4 method for signature 'CEMiTool'
plot_hist(cem, filter = FALSE)
```

**Arguments**

cem	Object of class CEMiTool
...	Optional parameters
filter	Logical. Whether or not to use filtered data for CEMiTool objects (Default: FALSE).

**Value**

Object of class CEMiTool containing expression histogram

**Examples**

```
# Get example CEMiTool object
data(cem)
# Plot histogram
cem <- plot_hist(cem)
# Check results
show_plot(cem, "hist")
```

---

plot\_interactions      *Network visualization*

---

## Description

Creates a graph based on interactions provided

## Usage

```
plot_interactions(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
plot_interactions(cem, n = 10, ...)
```

## Arguments

cem	Object of class CEMiTool.
...	Optional parameters.
n	number of nodes to label

## Value

Object of class CEMiTool with profile plots

## Examples

```
# Get example CEMiTool object  
data(cem)  
# Get example gene interactions data  
int <- system.file("extdata", "interactions.tsv", package = "CEMiTool")  
int_df <- read.delim(int)  
# Include interaction data into CEMiTool object  
interactions_data(cem) <- int_df  
# Plot resulting networks  
cem <- plot_interactions(cem)  
# Check resulting plot  
show_plot(cem, "interaction")
```

---

plot_mean_k	<i>Network mean connectivity</i>
-------------	----------------------------------

---

**Description**

Creates a graph showing the mean connectivity of genes in the network

**Usage**

```
plot_mean_k(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
plot_mean_k(cem, title = "Mean connectivity")
```

**Arguments**

cem	Object of class CEMiTool.
...	Optional parameters.
title	title of the graph

**Value**

Object of class CEMiTool with connectivity plot

**Examples**

```
# Get example CEMiTool object  
data(cem)  
# Plot scale-free model fit as a function of the soft-thresholding beta parameter choice  
cem <- plot_mean_k(cem)  
# Check resulting plot  
show_plot(cem, "mean_k")
```

---

plot_mean_var	<i>Plot mean and variance</i>
---------------	-------------------------------

---

**Description**

This plot returns a scatterplot of the mean by the variance of gene expression. A linear relationship between these values for RNAseq data suggest that an appropriate transformation such as the Variance Stabilizing Transformation should be applied.

**Usage**

```
plot_mean_var(cem, ...)

## S4 method for signature 'CEMiTool'
plot_mean_var(cem, filter = FALSE)
```

**Arguments**

cem	Object of class CEMiTool
...	Optional parameters
filter	Logical. Whether or not to use filtered data for CEMiTool objects (Default: FALSE).

**Value**

Object of class CEMiTool containing a mean and variance plot

**Examples**

```
# Get example CEMiTool object
data(cem)
# Plot mean and variance plot
cem <- plot_mean_var(cem)
# Check results
show_plot(cem, 'mean_var')
```

---

plot\_ora

*ORA visualization*

---

**Description**

Creates a bar plot with the results of module overrepresentation analysis

**Usage**

```
plot_ora(cem, ...)

## S4 method for signature 'CEMiTool'
plot_ora(cem, n = 10, pv_cut = 0.05, ...)
```

**Arguments**

cem	Object of class CEMiTool.
...	parameters to plot_ora_single
n	number of modules to show
pv_cut	p-value significance cutoff. Default is 0.05.

**Value**

Object of class CEMiTool with ORA plots

**Examples**

```
# Get example CEMiTool object
data(cem)
# Read example gmt file
gmt <- read_gmt(system.file('extdata', 'pathways.gmt',
                           package='CEMiTool'))
# Run overrepresentation analysis
cem <- mod_ora(cem, gmt)
# Plot module gene expression profiles
cem <- plot_ora(cem)
# Check resulting plot
show_plot(cem, "ora")
```

---

plot_ora_single	<i>ORA visualization for one module</i>
-----------------	-----------------------------------------

---

**Description**

ORA visualization for one module

**Usage**

```
plot_ora_single(
  es,
  ordr_by = "p.adjust",
  max_length = 50,
  pv_cut = 0.05,
  graph_color = "#4169E1",
  title = "Over Representation Analysis"
)
```

**Arguments**

es	a data.frame from ora function containing only one module
ordr_by	column to order the data.frame
max_length	max length of a gene set name
pv_cut	p-value cutoff
graph_color	color of bars
title	title of the graph

**Value**

a list with ggplot2 object and the number of significant gene sets

---

plot_profile	<i>Expression profile visualization</i>
--------------	-----------------------------------------

---

## Description

Creates a plot with module gene expression profiles along samples

## Usage

```
plot_profile(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
plot_profile(cem, order_by_class = TRUE, center_func = "mean")
```

## Arguments

cem	Object of class CEMiTool.
...	Optional parameters.
order_by_class	Logical. Only used if a sample annotation file is present. Whether or not to order by the class column in the sample annotation file (as defined by the class_column slot in cem).
center_func	Character string indicating the centrality measure to show in the plot. Either 'mean' (the default) or 'median'.

## Value

Object of class CEMiTool with profile plots

## Examples

```
# Get example CEMiTool object  
data(cem)  
# Plot module gene expression profiles  
cem <- plot_profile(cem)  
# Check resulting plot  
show_plot(cem, "profile")
```



---

plot_qq	<i>Plot quantile-quantile plot</i>
---------	------------------------------------

---

**Description**

This function creates a normal QQ plot of the expression values.

**Usage**

```
plot_qq(cem, ...)  
  
## S4 method for signature 'CEMiTool'  
plot_qq(cem, filter = FALSE)
```

**Arguments**

cem	Object of class CEMiTool
...	Optional parameters
filter	Logical. Whether or not to use filtered data for CEMiTool objects (Default: FALSE).

**Value**

Object of class CEMiTool containing qqplot

**Examples**

```
# Get example CEMiTool object  
data(cem)  
# Plot quantile-quantile plot  
cem <- plot_qq(cem)  
# Check results  
show_plot(cem, 'qq')
```

---

plot_sample_tree	<i>Sample clustering</i>
------------------	--------------------------

---

**Description**

Creates a dendrogram showing the similarities between samples in the expression data.

**Usage**

```
plot_sample_tree(cem, ...)

## S4 method for signature 'CEMiTool'
plot_sample_tree(
  cem,
  col_vector = NULL,
  sample_name_column = NULL,
  class_column = NULL,
  filter = FALSE
)
```

**Arguments**

<code>cem</code>	Object of class <code>CEMiTool</code> or <code>data.frame</code> .
<code>...</code>	Optional parameters.
<code>col_vector</code>	A vector of columns to use for visualizing the clustering. See Details.
<code>sample_name_column</code>	A string specifying the column to be used as sample identification. For <code>CEMiTool</code> objects, this will be the string specified in the <code>sample_name_column</code> slot.
<code>class_column</code>	A string specifying the column to be used as sample group identification. For <code>CEMiTool</code> objects, this will be the string specified in the <code>class_column</code> slot.
<code>filter</code>	Logical. Whether or not to use filtered data for <code>CEMiTool</code> objects (Default: <code>FALSE</code> ).

**Value**

Object of class `CEMiTool` with dendrogram or a plot object.

**Examples**

```
# Get example CEMiTool object
data(cem)
# Plot sample dendrogram
cem <- plot_sample_tree(cem)
# Check resulting plot
show_plot(cem, "sample_tree")
```

---

read\_gmt

*Read a GMT file*

---

**Description**

Read a GMT file

**Usage**

```
read_gmt(fname)
```

**Arguments**

fname            GMT file name.

**Value**

A list containing genes and description of each pathway

**Examples**

```
# Read example gmt file
gmt_fname <- system.file("extdata", "pathways.gmt", package = "CEMiTool")
gmt_in <- read_gmt(gmt_fname)
```

---

sample_annot	<i>Yellow Fever Sample Annotation data</i>
--------------	--------------------------------------------

---

**Description**

Modified data from a yellow fever vaccination study by Querec et al, 2009. This dataset, together with expr can be used as input for CEMiTool functions

**Usage**

```
data(sample_annot)
```

**Format**

An object of class data.frame

**Source**

[GEO](#)

**References**

Querec TD, Akondy RS, Lee EK, Cao W et al. Systems biology approach predicts immunogenicity of the yellow fever vaccine in humans. Nat Immunol 2009 Jan;10(1):116-25. PMID: 19029902  
[PubMed](#)

**Examples**

```
data(expr)
data(sample_annot)
# Run CEMiTool analysis
## Not run: cemitool(expr, sample_annot)
```

---

sample\_annotation      *Retrive or set the sample\_annotation attribute*

---

## Description

Retrive or set the sample\_annotation attribute

## Usage

```
sample_annotation(cem)

## S4 method for signature 'CEMiTool'
sample_annotation(cem)

sample_annotation(
  cem,
  sample_name_column = "SampleName",
  class_column = "Class"
) <- value

## S4 replacement method for signature 'CEMiTool'
sample_annotation(
  cem,
  sample_name_column = "SampleName",
  class_column = "Class"
) <- value
```

## Arguments

cem	Object of class CEMiTool
sample_name_column	A string containing the name of a column which should be used as a unique identifier for samples in the file. Only used when assigning a sample annotation data.frame. Default: "SampleName".
class_column	A string containing the name of a column which should be used to identify different sample groups. Only used when assigning a sample annotation data.frame. Default: "Class"
value	A data.frame containing the sample annotation, should have at least two columns containing the Class and the Sample Name that should match with samples in expression

## Value

A data.frame containing characteristics of each sample.

**Examples**

```

# Get example expression data
data(expr0)
# Get example sample_annotation data
data(sample_annot)
# Initialize CEMiTool object with expression
cem <- new_cem(expr0)
# Add sample annotation file to CEMiTool object
sample_annotation(cem,
  sample_name_column="SampleName",
  class_column="Class") <- sample_annot
# Check annotation
head(sample_annotation(cem))

```

---

save\_plots

*Save CEMiTool object plots*


---

**Description**

Save plots into the directory specified by the directory argument.

**Usage**

```

save_plots(cem, ...)

## S4 method for signature 'CEMiTool'
save_plots(
  cem,
  value = c("all", "profile", "gsea", "ora", "interaction", "beta_r2", "mean_k",
    "sample_tree", "mean_var", "hist", "qq"),
  force = FALSE,
  directory = "./Plots"
)

```

**Arguments**

cem	Object of class CEMiTool.
...	Optional parameters One of "all", "profile", "gsea", "ora", "interaction", "beta_r2", "mean_k", "sample_tree", "mean_var", "hist", "qq".
value	A character string containing the name of the plot to be saved.
force	If the directory exists, execution will not stop.
directory	Directory into which the files will be saved.

**Value**

A pdf file or files with the desired plot(s)

**Examples**

```
# Get example CEMiTool object
data(cem)
# Plot beta x R squared graph
cem <- plot_beta_r2(cem)
# Save plot
## Not run: save_plots(cem, value="beta_r2", directory="./Plots")
```

---

select_genes	<i>Select genes based on variance</i>
--------------	---------------------------------------

---

**Description**

Select genes based on variance

**Usage**

```
select_genes(expr, n_genes, filter_pval = 0.1)
```

**Arguments**

expr	A data.frame containing expression values
n_genes	(Optional) Number of genes to be selected
filter_pval	P-value cutoff for gene selection

**Value**

A vector containing the names of selected genes

**Examples**

```
# Get example expression data
data(expr0)
# Filter genes
expr_f <- filter_genes(expr0)
# Check selected genes
expr_f[1:5, 1:5]
# Filter genes and apply variance stabilizing transformation
expr_f2 <- filter_genes(expr0, apply_vst=TRUE)
# Check results
expr_f2[1:5, 1:5]
# Selected genes
selected <- select_genes(expr_f2)
# Get data.frame with only selected genes
expr_s <- expr_f2[selected, ]
# Check results
expr_s[1:5, 1:5]
```

---

show,CEMiTool-method    *Print a cemitool object*

---

### Description

Print a cemitool object

### Usage

```
## S4 method for signature 'CEMiTool'
show(object)
```

### Arguments

object                    Object of class CEMiTool

### Value

A CEMiTool object.

---

show\_plot                    *Retrieve CEMiTool object plots*

---

### Description

Retrieve CEMiTool object plots

### Usage

```
show_plot(cem, value)

## S4 method for signature 'CEMiTool'
show_plot(
  cem,
  value = c("profile", "gsea", "ora", "interaction", "beta_r2", "mean_k",
            "sample_tree", "mean_var", "hist", "qq")
)
```

### Arguments

cem                        Object of class CEMiTool.

value                     A character string containing the name of the plot to be shown. One of "profile", "gsea", "ora", "interaction", "beta\_r2", "mean\_k", "sample\_tree", "mean\_var", "hist", "qq".

**Value**

A plot corresponding to a CEMiTool analysis

**Examples**

```
# Get example CEMiTool object
data(cem)
# Plot beta x R squared graph
cem <- plot_beta_r2(cem)
# Check plot
show_plot(cem, "beta_r2")
```

---

vst

*Perform variance stabilizing transformation on expression file.*


---

**Description**

Perform variance stabilizing transformation on expression file.

**Usage**

```
vst(expr)
```

**Arguments**

expr                    expression file containing genes in the rows and samples in the columns

**Value**

A data.frame containing the results.

---

write\_files

*Save the CEMiTool object in files*


---

**Description**

Save the CEMiTool object in files

**Usage**

```
write_files(cem, ...)
```

```
## S4 method for signature 'CEMiTool'
write_files(cem, directory = "./Tables", force = FALSE)
```



**Arguments**

<code>cem</code>	Object of class <code>CEMiTool</code>
<code>...</code>	Optional parameters
<code>directory</code>	a directory
<code>force</code>	if the directory exists the execution will not stop

**Value**

A directory containing `CEMiTool` results in files.

**Examples**

```
# Get example CEMiTool object
data(cem)
# Save CEMiTool results in files
write_files(cem, directory=".", force=TRUE)
```

# Index

- \* **datasets**
  - expr0, [9](#)
  - sample\_annot, [43](#)
- \* **data**
  - cem, [4](#)
- \* **internal**
  - expr\_pct\_filter, [11](#)
  - module\_to\_gmt, [25](#)
  - plot\_ora\_single, [39](#)
  - vst, [48](#)
  
- adj\_data, [3](#)
- adj\_data, CEMiTool-method (adj\_data), [3](#)
- adj\_data<- (adj\_data), [3](#)
- adj\_data<- , CEMiTool-method (adj\_data), [3](#)
  
- cem, [4](#)
- cemitool, [5](#)
- CEMiTool-class, [7](#)
  
- diagnostic\_report, [8](#)
- diagnostic\_report, CEMiTool-method (diagnostic\_report), [8](#)
  
- expr0, [9](#)
- expr\_data, [10](#)
- expr\_data, CEMiTool-method (expr\_data), [10](#)
- expr\_data<- (expr\_data), [10](#)
- expr\_data<- , CEMiTool-method (expr\_data), [10](#)
- expr\_pct\_filter, [11](#)
  
- filter\_genes, [11](#)
- find\_modules, [12](#)
- find\_modules, CEMiTool-method (find\_modules), [12](#)
- fit\_data, [14](#)
- fit\_data, CEMiTool-method (fit\_data), [14](#)
  
- generate\_report, [14](#)
  
- generate\_report, CEMiTool-method (generate\_report), [14](#)
- get\_adj, [15](#)
- get\_adj, CEMiTool-method (get\_adj), [15](#)
- get\_beta\_data, [16](#)
- get\_beta\_data, CEMiTool-method (get\_beta\_data), [16](#)
- get\_cemitool\_r2\_beta, [17](#)
- get\_cemitool\_r2\_beta, CEMiTool-method (get\_cemitool\_r2\_beta), [17](#)
- get\_connectivity, [18](#)
- get\_connectivity, CEMiTool-method (get\_connectivity), [18](#)
- get\_hubs, [19](#)
- get\_hubs, CEMiTool-method (get\_hubs), [19](#)
- get\_merged\_mods, [20](#)
- get\_merged\_mods, CEMiTool-method (get\_merged\_mods), [20](#)
- get\_mods, [21](#)
- get\_mods, CEMiTool-method (get\_mods), [21](#)
- get\_phi, [22](#)
- get\_phi, CEMiTool-method (get\_phi), [22](#)
- gsea\_data, [22](#)
- gsea\_data, CEMiTool-method (gsea\_data), [22](#)
  
- interactions\_data, [23](#)
- interactions\_data, CEMiTool-method (interactions\_data), [23](#)
- interactions\_data<- (interactions\_data), [23](#)
- interactions\_data<- , CEMiTool-method (interactions\_data), [23](#)
  
- mod\_colors, [25](#)
- mod\_colors, CEMiTool-method (mod\_colors), [25](#)
- mod\_colors<- (mod\_colors), [25](#)
- mod\_colors<- , CEMiTool, character-method (mod\_colors), [25](#)

- mod\_gene\_num, [26](#)
- mod\_gene\_num, CEMiTool-method  
(mod\_gene\_num), [26](#)
- mod\_gsea, [27](#)
- mod\_gsea, CEMiTool-method (mod\_gsea), [27](#)
- mod\_names, [28](#)
- mod\_names, CEMiTool-method (mod\_names),  
[28](#)
- mod\_ora, [29](#)
- mod\_ora, CEMiTool-method (mod\_ora), [29](#)
- mod\_summary, [30](#)
- mod\_summary, CEMiTool-method  
(mod\_summary), [30](#)
- module\_genes, [24](#)
- module\_genes, CEMiTool-method  
(module\_genes), [24](#)
- module\_to\_gmt, [25](#)
  
- new\_cem, [30](#)
- nmodules, [32](#)
- nmodules, CEMiTool-method (nmodules), [32](#)
  
- ora\_data, [29](#), [32](#)
- ora\_data, CEMiTool-method (ora\_data), [32](#)
  
- plot\_beta\_r2, [33](#)
- plot\_beta\_r2, CEMiTool-method  
(plot\_beta\_r2), [33](#)
- plot\_gsea, [27](#), [34](#)
- plot\_gsea, CEMiTool-method (plot\_gsea),  
[34](#)
- plot\_hist, [35](#)
- plot\_hist, CEMiTool-method (plot\_hist),  
[35](#)
- plot\_interactions, [36](#)
- plot\_interactions, CEMiTool-method  
(plot\_interactions), [36](#)
- plot\_mean\_k, [37](#)
- plot\_mean\_k, CEMiTool-method  
(plot\_mean\_k), [37](#)
- plot\_mean\_var, [37](#)
- plot\_mean\_var, CEMiTool-method  
(plot\_mean\_var), [37](#)
- plot\_ora, [38](#)
- plot\_ora, CEMiTool-method (plot\_ora), [38](#)
- plot\_ora\_single, [39](#)
- plot\_profile, [40](#)
- plot\_profile, CEMiTool-method  
(plot\_profile), [40](#)
  
- plot\_qq, [41](#)
- plot\_qq, CEMiTool-method (plot\_qq), [41](#)
- plot\_sample\_tree, [41](#)
- plot\_sample\_tree, CEMiTool-method  
(plot\_sample\_tree), [41](#)
  
- read\_gmt, [42](#)
  
- sample\_annot, [43](#)
- sample\_annotation, [44](#)
- sample\_annotation, CEMiTool-method  
(sample\_annotation), [44](#)
- sample\_annotation<-  
(sample\_annotation), [44](#)
- sample\_annotation<-, CEMiTool-method  
(sample\_annotation), [44](#)
- save\_plots, [45](#)
- save\_plots, CEMiTool-method  
(save\_plots), [45](#)
- select\_genes, [46](#)
- show, CEMiTool-method, [47](#)
- show\_plot, [47](#)
- show\_plot, CEMiTool-method (show\_plot),  
[47](#)
  
- vst, [48](#)
  
- write\_files, [48](#)
- write\_files, CEMiTool-method  
(write\_files), [48](#)