

# Package ‘microbiomeExplorer’

April 12, 2022

**Type** Package

**Title** Microbiome Exploration App

**Version** 1.4.0

**Date** 2021-01-04

**Description** The MicrobiomeExplorer R package is designed to facilitate the analysis and visualization of marker-gene survey feature data.

It allows a user to perform and visualize typical microbiome analytical workflows either through the command line or an interactive

Shiny application included with the package. In addition to applying common analytical workflows the application enables automated analysis report generation.

**License** MIT + file LICENSE

**Imports** shinyjs (>= 2.0.0), shinydashboard, shinycssloaders, shinyWidgets, rmarkdown (>= 1.9.0), DESeq2, RColorBrewer, dplyr, tidyr, purrr, rlang, knitr, readr, DT (>= 0.12.0), biomformat, tools, stringr, vegan, matrixStats, heatmaply, car, broom, limma, reshape2, tibble, forcats, lubridate, methods, plotly (>= 4.9.1)

**Depends** shiny, magrittr, metagenomeSeq, Biobase

**Suggests** V8, testthat (>= 2.1.0)

**DeploySubPath** microbiomeExplorer

**biocViews** Classification, Clustering, GeneticVariability, DifferentialExpression, Microbiome, Metagenomics, Normalization, Visualization, MultipleComparison, Sequencing, Software, ImmunoOncology

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**VignetteBuilder** knitr

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

**git\_branch** RELEASE\_3\_14

**git\_last\_commit** 556ec83

**git\_last\_commit\_date** 2021-10-26

**Date/Publication** 2022-04-12

**Author** Joseph Paulson [aut],  
Janina Reeder [aut, cre],  
Mo Huang [aut],  
Genentech [cph, fnd]

**Maintainer** Janina Reeder <reederj1@gene.com>

## R topics documented:

abundanceHeatmap . . . . .	4
abundanceHeatmapUI . . . . .	5
addFeatData . . . . .	6
addPhenoData . . . . .	6
add_plotly_config . . . . .	7
add_plotly_layout . . . . .	7
aggFeatures . . . . .	8
aggregationTab . . . . .	8
aggregationTabUI . . . . .	9
alphaDiversity . . . . .	10
alphaDiversityUI . . . . .	11
avgAbundance . . . . .	11
avgAbundanceUI . . . . .	12
betaDiversity . . . . .	13
betaDiversityUI . . . . .	14
betaInput . . . . .	14
betaInputUI . . . . .	15
buildEmptyPlotlyPlot . . . . .	15
buildPlottingDF . . . . .	16
calculatePCAs . . . . .	17
computeCI_Interval . . . . .	17
computeDistMat . . . . .	18
corrAnalysis . . . . .	18
corrAnalysisUI . . . . .	19
corrFeature . . . . .	20
corrInput . . . . .	21
corrInputUI . . . . .	22
corrPhenotype . . . . .	23
createHeader . . . . .	24
dataInput . . . . .	25
dataInputUI . . . . .	26
designPairs . . . . .	26
diffAnalysis . . . . .	27
diffAnalysisUI . . . . .	28
diffInput . . . . .	28
diffInputUI . . . . .	29
diffTable . . . . .	29

diffTableUI . . . . .	30
extendPhenoData . . . . .	31
featAbundance . . . . .	31
featAbundanceUI . . . . .	32
featureAnalysis . . . . .	33
featureAnalysisUI . . . . .	34
featureCorr . . . . .	34
featureCorrUI . . . . .	35
featureInput . . . . .	36
featureInputUI . . . . .	37
featureTable . . . . .	37
featureTableUI . . . . .	38
fileUpload . . . . .	39
fileUploadUI . . . . .	40
filterByPheno . . . . .	40
filterMEData . . . . .	41
generateReport . . . . .	42
getFeatModCode . . . . .	43
getFile Type . . . . .	43
getFilterChoices . . . . .	44
getLegendLevel . . . . .	44
getPhenoChanges . . . . .	45
getPhenoModCode . . . . .	45
getWidths . . . . .	46
heatmapInput . . . . .	46
heatmapInputUI . . . . .	47
interAnalysis . . . . .	47
interAnalysisUI . . . . .	48
intraAnalysis . . . . .	49
intraAnalysisUI . . . . .	50
intraInput . . . . .	50
intraInputUI . . . . .	51
longAnalysis . . . . .	52
longAnalysisUI . . . . .	53
longInput . . . . .	53
longInputUI . . . . .	54
longResults . . . . .	55
longResultsUI . . . . .	56
makeQCPlot . . . . .	56
normalizeData . . . . .	57
parseInteractionName . . . . .	58
phenotypeCorr . . . . .	58
phenotypeCorrUI . . . . .	60
phenotypeTable . . . . .	60
phenotypeTableUI . . . . .	61
plotAbundance . . . . .	62
plotAlpha . . . . .	63
plotAvgAbundance . . . . .	64

plotBeta . . . . .	65
plotHeatmap . . . . .	66
plotLongFeature . . . . .	67
plotlyHistogram . . . . .	69
plotlySampleBarplot . . . . .	70
plotSingleFeature . . . . .	71
readData . . . . .	72
relAbundance . . . . .	73
relAbundanceUI . . . . .	74
replaceWithUnknown . . . . .	74
reportList . . . . .	75
reportListUI . . . . .	76
reportRow . . . . .	76
reportRowUI . . . . .	77
rollDownFeatures . . . . .	77
runDiffTest . . . . .	78
runMicrobiomeExplorer . . . . .	79

<b>Index</b>	<b>80</b>
--------------	-----------

---

abundanceHeatmap	<i>Abundance Heatmap module - server</i>
------------------	--

---

## Description

Abundance Heatmap module - server

## Usage

```
abundanceHeatmap(
  input,
  output,
  session,
  aggDat,
  featLevel,
  colorOptions,
  levelOpts,
  hmSort,
  hmFeatList,
  reset
)
```

## Arguments

input	shiny input
output	shiny output
session	shiny session

aggDat	aggregated MRExperiment
featLevel	chosen feature level (aggregation level)
colorOptions	reactive storing filters selected via data input
levelOpts	all available level choices for this dataset
hmSort	reactive storing sorting method for heatmap
hmFeatList	reactive storing list of features to include in heatmap
reset	boolean reactive which resets the module if TRUE

**Value**

R code needed to generate the heatmap

**Author(s)**

Janina Reeder

---

abundanceHeatmapUI      *Abundance Heatmap module - UI*

---

**Description**

Abundance Heatmap module - UI

**Usage**

abundanceHeatmapUI(id)

**Arguments**

id                    namespace identifier

**Value**

box holding the UI code

**Author(s)**

Janina Reeder

---

addFeatData	<i>Add feature data to MRobj.</i>
-------------	-----------------------------------

---

**Description**

This function adds feature data to the featureData slot in an MRExperiment object.

**Usage**

```
addFeatData(MRobj, featdata = NULL)
```

**Arguments**

MRobj	An MRExperiment object.
featdata	Feature data frame or file path.

**Value**

An updated MRExperiment object.

---

addPhenoData	<i>Add phenotype data to object.</i>
--------------	--------------------------------------

---

**Description**

This function adds phenotype data to the phenoData slot in an MRExperiment object.

**Usage**

```
addPhenoData(MRobj, phenodata = NULL)
```

**Arguments**

MRobj	An MRExperiment object.
phenodata	Phenotype data frame or file path.

**Value**

An updated MRExperiment object.

---

add\_plotly\_config      *Adds a config call based on plotly::config*

---

**Description**

Adds a config call based on plotly::config

**Usage**

```
add_plotly_config(.data)
```

**Arguments**

.data                  plotly data object to apply the config call to

**Value**

plotly::config call

---

add\_plotly\_layout      *Adds a layout call based on plotly::layout*

---

**Description**

Adds a layout call based on plotly::layout

**Usage**

```
add_plotly_layout(.data, plotTitle, xaxis_text, ylab)
```

**Arguments**

.data                  plotly data object to apply the layout call to  
plotTitle              plot title to use  
xaxis\_text             x axis label to use  
ylab                    y axis label to use

**Value**

plotly::layout call

aggFeatures *Aggregates counts by level*

---

### Description

This function aggregates counts by a level specified in the featureData slot of the MRExperiment object.

### Usage

```
aggFeatures(MRobj, level = NULL, sort = TRUE)
```

### Arguments

MRobj	An MRExperiment object.
level	Level to aggregate over. If NULL, no aggregation occurs.
sort	boolean determining if resulting aggregated MRExperiment should be sorted based on rowSums; default is TRUE

### Value

Aggregated MRExperiment object or matrix depending on out.

### Examples

```
data("mouseData", package = "metagenomeSeq")
aggFeatures(mouseData, level = "genus")
```

---

aggregationTab *Aggregation module server function*

---

### Description

Aggregation module server function

### Usage

```
aggregationTab(
  input,
  output,
  session,
  resetInput,
  levelOpts,
  chosenLevel,
  meData
)
```



**Arguments**

input	shiny input
output	shiny output
session	shiny session
resetInput	boolean updated to TRUE if new data is available
levelOpts	available levels to aggregate on (depends on input data)
chosenLevel	previously selected level (passed from different instance)
meData	the main MExperiment object

**Value**

reactive list holding aggregated object, aggregation code and boolean on normalization

**Author(s)**

Janina Reeder

---

aggregationTabUI      *Aggregation module ui function*

---

**Description**

Aggregation module ui function

**Usage**

```
aggregationTabUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

box holding aggregation input elements

**Author(s)**

Janina Reeder

**Examples**

```
aggregationTabUI("atu_id")
```

---

`alphaDiversity`*Alpha Diversity module - server*

---

**Description**

Alpha Diversity module - server

**Usage**

```
alphaDiversity(  
  input,  
  output,  
  session,  
  aggDat,  
  featLevel,  
  intraSettings,  
  colorOptions,  
  reset  
)
```

**Arguments**

<code>input</code>	shiny input
<code>output</code>	shiny output
<code>session</code>	shiny session
<code>aggDat</code>	aggregated MRExperiment
<code>featLevel</code>	chosen feature level (aggregation level)
<code>intraSettings</code>	analysis settings as passed over from analysis input module
<code>colorOptions</code>	phenotype selections: used for color choices
<code>reset</code>	boolean reactive which resets the module if TRUE

**Value**

R code used to make the alpha diversity plot

**Author(s)**

Janina Reeder

---

alphaDiversityUI      *Alpha Diversity module - UI*

---

**Description**

Alpha Diversity module - UI

**Usage**

```
alphaDiversityUI(id)
```

**Arguments**

id                    namespace identifier

**Value**

box holding the UI code

**Author(s)**

Janina Reeder

---

avgAbundance              *Relative abundance plot module - server*

---

**Description**

Relative abundance plot module - server

**Usage**

```
avgAbundance(  
  input,  
  output,  
  session,  
  aggDat,  
  featLevel,  
  featureSettings,  
  normalizedData,  
  reset  
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
aggDat	aggregated MRExperiment
featLevel	chosen feature level (aggregation level)
featureSettings	analysis input settings passed over to this module
normalizedData	boolean indicating whether data has been normalized
reset	boolean reactive which resets the module if TRUE

**Value**

list storing plot clicks and number of features displayed (passed to feature plot module) as well as the R code to make plot

---

avgAbundanceUI

*Relative abundance plot module - UI*

---

**Description**

Relative abundance plot module - UI

**Usage**

```
avgAbundanceUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

box containing the ui code

**Author(s)**

Janina Reeder

---

betaDiversity      *Beta Diversity module - server*

---

## Description

Beta Diversity module - server

## Usage

```
betaDiversity(  
  input,  
  output,  
  session,  
  aggDat,  
  aggLevel,  
  colorOptions,  
  shapeOptions,  
  betadistance,  
  betaSettings,  
  reset  
)
```

## Arguments

input	shiny input
output	shiny output
session	shiny session
aggDat	MRExperiment storing data
aggLevel	aggregation level
colorOptions	phenotype selection options for color
shapeOptions	phenotype selection options for shape
betadistance	distance measured used for beta diversity analysis
betaSettings	input choices for beta diversity
reset	boolean reactive which resets the module if TRUE

## Value

R code needed to generate the beta diversity plot

## Author(s)

Janina Reeder

---

betaDiversityUI      *Beta Diversity module - UI*

---

**Description**

Beta Diversity module - UI

**Usage**

```
betaDiversityUI(id)
```

**Arguments**

id                    namespace identifier

**Value**

box holding the ui code

**Author(s)**

Janina Reeder

---

betaInput              *Server side for the analysis input module handling analysis control*

---

**Description**

Server side for the analysis input module handling analysis control

**Usage**

```
betaInput(input, output, session, meData, adonisOptions, reset)
```

**Arguments**

input                shiny input  
output               shiny output  
session               shiny session  
meData               MRExperiment object storing all data  
adonisOptions       phenodata columns ready for adonis analysis  
reset                reactive boolean determining if all inputs should be reset

**Value**

list holding all chosen values and the selected feature

**Author(s)**

Janina Reeder

---

betaInputUI	<i>Main beta analysis input module. Set up to handle all analysis tabs in the app depending on given parameters</i>
-------------	---

---

**Description**

Main beta analysis input module. Set up to handle all analysis tabs in the app depending on given parameters

**Usage**

```
betaInputUI(id)
```

**Arguments**

id	element identifier - namespace
----	--------------------------------

**Value**

box containing ui element

**Author(s)**

Janina Reeder

---

buildEmptyPlotlyPlot	<i>Creates an empty plotly plot using the given labels on the x and y axis</i>
----------------------	--

---

**Description**

Creates an empty plotly plot using the given labels on the x and y axis

**Usage**

```
buildEmptyPlotlyPlot(xaxis_text, ylab)
```

**Arguments**

xaxis_text	x axis label
ylab	y axis label

**Value**

call to plotly\_empty

---

buildPlottingDF	<i>Sets up a dataframe used by several plotting functions by joining the required data with relevant phenotype data</i>
-----------------	---

---

**Description**

Sets up a dataframe used by several plotting functions by joining the required data with relevant phenotype data

**Usage**

```
buildPlottingDF(
  df,
  phenoTable,
  x_var = NULL,
  facet1 = NULL,
  facet2 = NULL,
  col_by = NULL,
  col_name = col_by,
  id_var = NULL
)
```

**Arguments**

df	dataframe storing plotting data values
phenoTable	pData of the MRExperiment; all following parameters must be a column of the phenoTable
x_var	main plotting variable
facet1	column-based faceting (can be NULL)
facet2	row-based faceting (can be NULL)
col_by	coloring factor (can be NULL)
col_name	character to be used as name for col_by
id_var	variable used to connect samples longitudinally (can be NULL)

**Value**

dataframe obtained by joining df and relevant columns of phenoTable



---

calculatePCAs	<i>Function to compute the PCAs for a given distance matrix</i>
---------------	---

---

**Description**

Function to compute the PCAs for a given distance matrix

**Usage**

```
calculatePCAs(distmat, pcas)
```

**Arguments**

distmat	the distance matrix
pcas	2-element vector of PCAs to include in results

**Value**

the x slot limited to pcas after calling stats::prcomp on distmat

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
distmat <- computeDistMat(aggdat, dist_method = "bray")
calculatePCAs(distmat, c(1,2))
```

---

computeCI_Interval	<i>Helper function to calculate the confidence interval for a cor.test</i>
--------------------	--

---

**Description**

Helper function to calculate the confidence interval for a cor.test

**Usage**

```
computeCI_Interval(num, mS, method)
```

**Arguments**

num	number of samples
mS	results of cor.test
method	statistical method used for cor.test

**Value**

named vector holding lower and upper thresholds

computeDistMat      *Function to compute the distance matrix using vegdist from the vegan package*

---

**Description**

Function to compute the distance matrix using vegdist from the vegan package

**Usage**

```
computeDistMat(aggdat, dist_method, log = TRUE, nfeatures = nrow(aggmat))
```

**Arguments**

aggdat	aggregated MRExperiment
dist_method	distance method from vegan package (See ?vegan::vegdist for details)
log	transform count matrix to log2; default is TRUE
nfeatures	number of features to use; default is all

**Value**

distance as dist

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
computeDistMat(aggdat, dist_method = "bray")
```

---

corrAnalysis      *corr Analysis Module - server*

---

**Description**

corr Analysis Module - server

**Usage**

```
corrAnalysis(
  input,
  output,
  session,
  data,
  levelOpts,
  chosenLevel,
  resetInput,
  aggData
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
data	the main data object returned from data_input_module
levelOpts	available levels to aggregate on (depends on input data)
chosenLevel	previously selected level (passed from different instance)
resetInput	reactive boolean determining if reset is required
aggData	the aggregated MRExperiment object

**Value**

reactive holding code to be used in reports

---

corrAnalysisUI      *corr Analysis Module - UI*

---

**Description**

corr Analysis Module - UI

**Usage**

```
corrAnalysisUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

fluidRow containing the ui code

**Author(s)**

Janina Reeder

**Examples**

```
corrAnalysisUI("coranalysis_id")
```

---

 corrFeature

*Scatterplot of two features*


---

**Description**

This function plots a scatterplot of two features along with sample correlation statistics.

**Usage**

```
corrFeature(
  aggdat,
  feat1,
  feat2,
  log = TRUE,
  method = c("spearman", "pearson", "kendall"),
  addRegression = TRUE,
  col_by = NULL,
  facet1 = NULL,
  facet2 = NULL,
  plotTitle = "",
  xlab = NULL,
  ylab = NULL,
  allowWebGL = TRUE,
  pwidth = 550,
  pheight = 200
)
```

**Arguments**

aggdat	aggregated MRExperiment
feat1	Feature 1.
feat2	Feature 2.
log	Log2 transform data. Default is TRUE.
method	Correlation coefficient. One of "spearman" (default), "pearson", or "kendall".
addRegression	boolean parameter indicating whether linear regression line should be drawn; default: TRUE
col_by	Phenotype for coloring.

facet1	Phenotype for facet 1.
facet2	Phenotype for facet 2.
plotTitle	Plot title. Default is no title.
xlab	X-axis label. Default is feat1.
ylab	Y-axis label. Default is feat2.
allowWebGL	boolean indicating if WebGL should be used for large data
pwidth	overall plot width; default is 550
pheight	overall plot height; default is 200

**Value**

list holding plotly plot and lm fit

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
corrFeature(aggdat, feat1 = "Bacteroides", feat2 = "Prevotella")
```

---

corrInput

*Server side for the analysis input module handling analysis control*

---

**Description**

Server side for the analysis input module handling analysis control

**Usage**

```
corrInput(  
  input,  
  output,  
  session,  
  type,  
  meData,  
  facetOptions = NULL,  
  reset,  
  aggDat = reactive(NULL)  
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
type	of the correlation (feature vs phenotype)
meData	MRExperiment object storing all data
facetOptions	named vector of available facet choices
reset	reactive boolean determining if all inputs should be reset
aggDat	aggregated MRExperiment object (default is NULL)

**Value**

list holding all chosen values and the selected feature

**Author(s)**

Janina Reeder

---

corrInputUI	<i>Main correlation analysis input module. Handles correlation analysis tab in the app</i>
-------------	--

---

**Description**

Main correlation analysis input module. Handles correlation analysis tab in the app

**Usage**

```
corrInputUI(id, type)
```

**Arguments**

id	element identifier - namespace
type	determines if 'feature' or 'pheno' correlation

**Value**

box containing ui element

**Author(s)**

Janina Reeder

---

corrPhenotype	<i>Scatterplot of a feature and a phenotype</i>
---------------	---

---

### Description

This function plots a scatterplot of a feature and a phenotype along with sample correlation statistics.

### Usage

```
corrPhenotype(
  aggdat,
  feature,
  phenotype,
  log = TRUE,
  method = c("spearman", "pearson", "kendall"),
  addRegression = TRUE,
  col_by = NULL,
  facet1 = NULL,
  facet2 = NULL,
  plotTitle = "",
  xlab = NULL,
  ylab = NULL,
  allowWebGL = TRUE,
  pwidth = 550,
  pheight = 200
)
```

### Arguments

aggdat	aggregated MRExperiment
feature	Feature input.
phenotype	Phenotype input (must be numeric)
log	Log2 transform data. Default is TRUE.
method	Correlation coefficient. One of "spearman" (default), "pearson", or "kendall".
addRegression	boolean parameter indicating whether linear regression line should be drawn; default: TRUE
col_by	Phenotype for coloring.
facet1	Phenotype for facet 1.
facet2	Phenotype for facet 2. (WIP/TODO)
plotTitle	Plot title. Default is no title.
xlab	X-axis label. Default is feat1.
ylab	Y-axis label. Default is feat2.
allowWebGL	boolean indicating if WebGL should be used for large data
pwidth	overall plot width; default is 550
pheight	overall plot height; default is 200

**Value**

list holding plotly plot and lm fit

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
corrPhenotype(aggdat, feature = "Bacteroides", phenotype = "relativeTime")
```

---

createHeader

*Makes header for R script*

---

**Description**

This function makes the header for the report R script to be rendered by knitr into Rmarkdown and rendered into HTML, PDF, or Word.

**Usage**

```
createHeader(  
  title = "MicrobiomeExplorer Report",  
  author = "",  
  date = "",  
  data.source = "",  
  output = getOption("me.reportformat"),  
  toc = TRUE  
)
```

**Arguments**

title	Title of the report.
author	Author of the report.
date	Date of the report.
data.source	R code used to obtain the dataset
output	Output of Rmarkdown file.
toc	Table of contents. Default is TRUE.

**Details**

This was adapted from <https://yihui.name/knitr/demo/stitch/>

**Value**

A character vector where each element is a line in the R script.



---

dataInput	<i>Main Data input server where the user selects files to upload to the app or connects to database</i>
-----------	---

---

**Description**

Main Data input server where the user selects files to upload to the app or connects to database

**Usage**

```
dataInput(  
  input,  
  output,  
  session,  
  dataSource,  
  dataFilterRep,  
  qcRep,  
  addPheno,  
  resetReports  
)
```

**Arguments**

input	module input
output	module output
session	app session
dataSource	reactive Value storing commands for loading data
dataFilterRep	reactive Value storing commands for filtering data
qcRep	reactive Value storing commands for producing qc plots
addPheno	reactive boolean keeping track of phenodata changes
resetReports	reactive boolean indicating whether reports need to be reset

**Value**

list of reactives containing the uploaded and filtered data as well as the filterChoices on phenotypes

**Author(s)**

Janina Reeder

---

dataInputUI	<i>Main Data input UI where the user selects files to upload to the app or connects to database</i>
-------------	---

---

**Description**

Main Data input UI where the user selects files to upload to the app or connects to database

**Usage**

```
dataInputUI(id)
```

**Arguments**

id	module identifier
----	-------------------

**Value**

fluidRow holding UI interface

**Author(s)**

Janina Reeder

**Examples**

```
dataInputUI("datainput_id")
```

---

designPairs	<i>Produce design matrix of pairwise comparisons</i>
-------------	--

---

**Description**

This function takes in the levels of a factor phenotype and outputs a design matrix of all pairwise comparisons.

**Usage**

```
designPairs(levels)
```

**Arguments**

levels	Character vector of the levels of a factor phenotype
--------	--

**Value**

A model matrix

---

`diffAnalysis`*diff Analysis Module - server*

---

**Description**

diff Analysis Module - server

**Usage**

```
diffAnalysis(  
  input,  
  output,  
  session,  
  data,  
  levelOpts,  
  chosenLevel,  
  resetInput,  
  aggData,  
  normalizedData  
)
```

**Arguments**

<code>input</code>	shiny input
<code>output</code>	shiny output
<code>session</code>	shiny session
<code>data</code>	the main data object returned from <code>data_input_module</code>
<code>levelOpts</code>	available levels to aggregate on (depends on input data)
<code>chosenLevel</code>	previously selected level (passed from different instance)
<code>resetInput</code>	reactive boolean determining if reset is required
<code>aggData</code>	the aggregated MRExperiment object
<code>normalizedData</code>	boolean indicating if normalization was done

**Value**

reactive holding code to be used in reports

**Author(s)**

Janina Reeder

---

diffAnalysisUI	<i>Diff Analysis Module - UI</i>
----------------	----------------------------------

---

**Description**

Diff Analysis Module - UI

**Usage**

```
diffAnalysisUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

fluidRow containing the ui code

**Author(s)**

Janina Reeder

**Examples**

```
diffAnalysisUI("diffanalysis_id")
```

---

diffInput	<i>Server side for the analysis input module handling analysis control</i>
-----------	--

---

**Description**

Server side for the analysis input module handling analysis control

**Usage**

```
diffInput(input, output, session, meData, facetOptions = NULL, reset)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
meData	MRExperiment object storing all data
facetOptions	named vector of available facet choices
reset	reactive boolean determining if all inputs should be reset

**Value**

list holding all chosen values and the selected feature

**Author(s)**

Janina Reeder

---

diffInputUI	<i>Main diffanalysis input module. Set up to handle diff analysis tabs in the app depending on given parameters</i>
-------------	---

---

**Description**

Main diffanalysis input module. Set up to handle diff analysis tabs in the app depending on given parameters

**Usage**

diffInputUI(id)

**Arguments**

id                    element identifier - namespace

**Value**

box containing ui element

**Author(s)**

Janina Reeder

---

diffTable	<i>Differential analysis module server code</i>
-----------	---

---

**Description**

Differential analysis module server code

**Usage**

```
diffTable(  
  input,  
  output,  
  session,  
  aggDat,  
  featLevel,  
  diffSettings,  
  reset,  
  normalized  
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
aggDat	aggregated MRExperiment
featLevel	chosen feature level (aggregation level)
diffSettings	reactive storing values selected in analysis input interface
reset	boolean reactive which resets the module if TRUE
normalized	boolean reactive indicating if data has been normalized

**Value**

list containing R code for analysis and for feature plots

**Author(s)**

Janina Reeder

---

diffTableUI

*Differential Analysis module UI*

---

**Description**

Differential Analysis module UI

**Usage**

```
diffTableUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

row containing the UI elements

**Author(s)**

Janina Reeder

---

extendPhenoData	<i>Extends existing phenodata for an object</i>
-----------------	---

---

**Description**

This function adds phenotype data to the phenoData slot in an MRExperiment object.

**Usage**

```
extendPhenoData(MRobj, phenodata = NULL)
```

**Arguments**

MRobj	An MRExperiment object.
phenodata	Phenotype data frame or file path.

**Value**

An updated MRExperiment object.

---

featAbundance	<i>Feature plot module - server</i>
---------------	-------------------------------------

---

**Description**

Feature plot module - server

**Usage**

```
featAbundance(  
  input,  
  output,  
  session,  
  aggDat,  
  featLevel,  
  intraSettings,  
  selectedFeat,  
  featName,
```

```

    numOfFeats,
    ylabMode,
    normalizedData,
    reset
  )

```

### Arguments

input	shiny input
output	shiny output
session	shiny session
aggDat	aggregated MRExperiment
featLevel	chosen feature level (aggregation level)
intraSettings	analysis settings passed over from analysis input module
selectedFeat	feature selected via drop down element of analysis input
featName	plotly click event passed via relative abundance
numOfFeats	number of features shown in relative abundance plot (affects plotly click data)
ylabMode	character indication if raw \"Reads\" or \"Percentage\" should be shown
normalizedData	boolean indicating whether data has been normalized
reset	boolean reactive which resets the module if TRUE

### Value

R code needed to build the feature plot

### Author(s)

Janina Reeder

---

featAbundanceUI	<i>Feature plot module - UI</i>
-----------------	---------------------------------

---

### Description

Feature plot module - UI

### Usage

```
featAbundanceUI(id)
```

### Arguments

id	namespace identifier
----	----------------------

### Value

box holding the UI code



---

featureAnalysis      *feature Analysis Module - server*

---

**Description**

feature Analysis Module - server

**Usage**

```
featureAnalysis(  
  input,  
  output,  
  session,  
  data,  
  resetInput,  
  aggData,  
  normalizedData  
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
data	the main data object returned from data_input_module
resetInput	reactive boolean determining if reset is required
aggData	the aggregated MRExperiment object
normalizedData	boolean indicating if normalization was done

**Value**

reactive holding code to be used in reports

**Author(s)**

Janina Reeder

---

featureAnalysisUI      *feature Analysis Module - UI*

---

**Description**

feature Analysis Module - UI

**Usage**

```
featureAnalysisUI(id)
```

**Arguments**

id                      namespace identifier

**Value**

fluidRow containing the ui code

**Author(s)**

Janina Reeder

**Examples**

```
featureAnalysisUI("featureanalysis_id")
```

---

featureCorr                      *Feature correlation analysis server module*

---

**Description**

Feature correlation analysis server module

**Usage**

```
featureCorr(  
  input,  
  output,  
  session,  
  aggDat,  
  colorOptions,  
  corFeatBase,  
  corFeat2,  
  corFacet1,
```

```

    corFacet2,
    corMethod,
    reset
  )

```

### Arguments

input	module input
output	module output
session	app session
aggDat	aggregated MRExperiment
colorOptions	reactive storing filters available via data input
corFeatBase	first correlation feature
corFeat2	second correlation feature
corFacet1	first correlation facet
corFacet2	second correlation facet
corMethod	correlation method to use
reset	boolean reactive which resets the module if TRUE

### Value

R code used to do the correlation analysis (character)

### Author(s)

Janina Reeder

---

featureCorrUI	<i>Feature correlation analysis module UI</i>
---------------	---

---

### Description

Feature correlation analysis module UI

### Usage

```
featureCorrUI(id)
```

### Arguments

id	namespace identifier
----	----------------------

### Value

box containing the UI elements

**Author(s)**

Janina Reeder

---

`featureInput`*Server side for the feature analysis input module*

---

**Description**

Server side for the feature analysis input module

**Usage**

```
featureInput(  
  input,  
  output,  
  session,  
  meData,  
  facetOptions = NULL,  
  reset,  
  aggDat = reactive(NULL)  
)
```

**Arguments**

<code>input</code>	shiny input
<code>output</code>	shiny output
<code>session</code>	shiny session
<code>meData</code>	MRExperiment object storing all data
<code>facetOptions</code>	named vector of available facet choices
<code>reset</code>	reactive boolean determining if all inputs should be reset
<code>aggDat</code>	aggregated MRExperiment object (default is NULL)

**Value**

list holding all chosen values and the selected feature

**Author(s)**

Janina Reeder

---

featureInputUI	<i>Main feature analysis input module. Set up to handle all analysis tabs in the app depending on given parameters</i>
----------------	--

---

**Description**

Main feature analysis input module. Set up to handle all analysis tabs in the app depending on given parameters

**Usage**

```
featureInputUI(id)
```

**Arguments**

id	element identifier - namespace
----	--------------------------------

**Value**

box containing ui element

**Author(s)**

Janina Reeder

---

featureTable	<i>Feature table module server code</i>
--------------	---

---

**Description**

Feature table module server code

**Usage**

```
featureTable(input, output, session, meData, featureModRep)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
meData	MRExperiment storing the data
featureModRep	reactiveValue storing modifications performed on fData

**Value**

feature table server fragment - no return value

**Author(s)**

Janina Reeder

---

featureTableUI	<i>Feature table UI module</i>
----------------	--------------------------------

---

**Description**

Feature table UI module

**Usage**

```
featureTableUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

fluidRow containing the UI code for feature tables

**Author(s)**

Janina Reeder

**Examples**

```
featureTableUI("feature_id")
```

---

`fileUpload`*Module handling file upload for the application: server*

---

**Description**

Module handling file upload for the application: server

**Usage**

```
fileUpload(  
    input,  
    output,  
    session,  
    meData,  
    meName,  
    initializeData,  
    addPheno,  
    dataSource,  
    resetFile = reactive(NULL)  
)
```

**Arguments**

<code>input</code>	module input
<code>output</code>	module output
<code>session</code>	app session
<code>meData</code>	main reactive storing the MRexperiment data
<code>meName</code>	main reactive storing the filename uploaded
<code>initializeData</code>	reactiveVal keeping track of new uploads to reset data
<code>addPheno</code>	reactiveVal keeping track of phenodata changes
<code>dataSource</code>	reactive Value storing commands for loading data
<code>resetFile</code>	indicating if module should be reset

**Value**

boolean denoting successful upload of a file

**Author(s)**

Janina Reeder

---

fileUploadUI	<i>Module handling file upload for the application: UI In a deployed version this module should be replaced with database access</i>
--------------	--

---

**Description**

Module handling file upload for the application: UI In a deployed version this module should be replaced with database access

**Usage**

```
fileUploadUI(id)
```

**Arguments**

id	module identifier
----	-------------------

**Value**

div holding ui elements

**Author(s)**

Janina Reeder

---

filterByPheno	<i>Function to filter the MRExperiment by certain phenotype values</i>
---------------	--

---

**Description**

Function to filter the MRExperiment by certain phenotype values

**Usage**

```
filterByPheno(MRobj, rm_phenovalues)
```

**Arguments**

MRobj	the MRExperiment to subset
rm_phenovalues	list of named vectors with names corresponding to column names in pData and values representing phenotypes within the column

**Value**

the filtered MRobj



**Author(s)**

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
filterByPheno(MRobj = mouseData,
  rm_phenovalues = list("diet" = c("BK"), "mouseID" = c("PM1", "PM10")))
```

---

filterMEData

*Function to filter the MRExperiment data by numerical parameters*

---

**Description**

Function to filter the MRExperiment data by numerical parameters

**Usage**

```
filterMEData(MRobj, minpresence = 1, minfeats = 2, minreads = 2)
```

**Arguments**

MRobj	MRExperiment object to filter
minpresence	minimum sample presence per feature
minfeats	minimum number of features per sample
minreads	minimum number of reads per sample

**Value**

the filtered MRobj

**Author(s)**

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
filterMEData(MRobj = mouseData, minpresence = 4, minfeats = 300)
```

---

generateReport	<i>Generates report</i>
----------------	-------------------------

---

### Description

This function generates the pieces of the report, which includes the R script, Rmarkdown file, and any Rmarkdown outputs.

### Usage

```
generateReport(
  rcode,
  filename = "report",
  dir = "out",
  title = "MicrobiomeExplorer Report",
  author = "",
  date = "`r format(Sys.time(), '%d %B, %Y')`",
  data.source = "",
  output = c("html_document"),
  toc = TRUE,
  intro_text = NULL
)
```

### Arguments

rcode	A named list where each element corresponds to a different analysis (Alpha diversity, Beta diversity). The name of the list is used to denote the first part of the code chunks in each analysis section (alpha, beta). Each element is itself a list of R commands corresponding to a code chunk.
filename	Name of output files. Default is "report".
dir	Directory of output. Default is "out".
title	Title of the report.
author	Author of the report.
date	Date of the report.
data.source	R code used to obtain the dataset
output	Output of Rmarkdown file. Options defined in global.R
toc	Table of contents. Default is TRUE.
intro_text	Introductory text to include with the report (optional)

### Details

Adapted from <https://yihui.name/knitr/demo/stitch/>

### Value

A character vector where each element is a line in the R script.

---

getFeatModCode	<i>Helper function returning the fData modifications as strings for report generation</i>
----------------	---

---

**Description**

Helper function returning the fData modifications as strings for report generation

**Usage**

```
getFeatModCode(featureanno)
```

**Arguments**

featureanno      type of feature annotation; values are "Mark unknown" or "Roll down"

**Value**

String containing R code performing the modification

---

getFiletype	<i>Helper function assigning different file extensions to specific short texts identifying the types</i>
-------------	--

---

**Description**

Helper function assigning different file extensions to specific short texts identifying the types

**Usage**

```
getFiletype(fileext)
```

**Arguments**

fileext            the file extension found after '.'

**Value**

character string for the filetype

**Author(s)**

Janina Reeder

---

getFilterChoices	<i>Helper function to filter phenodata for interesting phenotypes to be used for filtering or subsetting</i>
------------------	--

---

**Description**

Helper function to filter phenodata for interesting phenotypes to be used for filtering or subsetting

**Usage**

```
getFilterChoices(MRobj)
```

**Arguments**

MRobj            the MRexperiment storing the data

**Value**

list of named vectors with names being pData column headers and values being unique entries; columns with only one entry or those with different values for each samples are omitted

**Author(s)**

Janina Reeder

---

getLegendLevel	<i>Function to find a non-empty facet in the last row. This will be the one to be connected to the plot legend to avoid duplicates within</i>
----------------	---

---

**Description**

Function to find a non-empty facet in the last row. This will be the one to be connected to the plot legend to avoid duplicates within

**Usage**

```
getLegendLevel(df2, facets, facet2s)
```

**Arguments**

df2            plotting data frame  
 facets        column facets  
 facet2s       row facets

**Value**

the name of the column-based facet which can be used as legend

---

getPhenoChanges	<i>Helper function returning the code used to modify the data types of the pheno table</i>
-----------------	--

---

**Description**

Helper function returning the code used to modify the data types of the pheno table

**Usage**

```
getPhenoChanges(phenotype, datatype)
```

**Arguments**

phenotype	name of the phenotype column header
datatype	variable type to assign to column

**Value**

String storing code to perform modification

**Author(s)**

Janina Reeder

---

getPhenoModCode	<i>Helper function returning the code used to modify the phenotable as a string</i>
-----------------	---

---

**Description**

Helper function returning the code used to modify the phenotable as a string

**Usage**

```
getPhenoModCode(name, pheno1, pheno2)
```

**Arguments**

name	interaction name
pheno1	first interaction phenotype
pheno2	second interaction phenotype

**Value**

String storing code to perform modification

**Author(s)**

Janina Reeder

---

getWidths	<i>Helper function to account for issues plotly has with very small widths (these end up being 1 and cover the entire plotting area)</i>
-----------	--

---

**Description**

Helper function to account for issues plotly has with very small widths (these end up being 1 and cover the entire plotting area)

**Usage**

```
getWidths(df2, facets, x_var, drop = TRUE)
```

**Arguments**

df2	dataframe storing plotting data
facets	column facets
x_var	x variable
drop	passed on as .drop to dplyr::group_by

**Value**

widths for each facet

---

heatmapInput	<i>Server side for the analysis input module handling analysis control</i>
--------------	--

---

**Description**

Server side for the analysis input module handling analysis control

**Usage**

```
heatmapInput(input, output, session, meData, reset, aggDat = reactive(NULL))
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
meData	MRExperiment object storing all data
reset	reactive boolean determining if all inputs should be reset
aggDat	aggregated MRExperiment object (default is NULL)

**Value**

list holding all chosen values and the selected feature

**Author(s)**

Janina Reeder

---

heatmapInputUI	<i>Heatmap analysis input module. Set up to handle all analysis tabs in the app depending on given parameters</i>
----------------	---

---

**Description**

Heatmap analysis input module. Set up to handle all analysis tabs in the app depending on given parameters

**Usage**

```
heatmapInputUI(id)
```

**Arguments**

id	element identifier - namespace
----	--------------------------------

**Value**

box containing ui element

**Author(s)**

Janina Reeder

---

interAnalysis	<i>inter Analysis Module - server</i>
---------------	---------------------------------------

---

**Description**

inter Analysis Module - server

**Usage**

```
interAnalysis(
  input,
  output,
  session,
  data,
  levelOpts,
  chosenLevel,
  resetInput,
  aggData
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
data	the main data object returned from data_input_module
levelOpts	available levels to aggregate on (depends on input data)
chosenLevel	previously selected level (passed from different instance)
resetInput	reactive boolean determining if reset is required
aggData	the aggregated MRExperiment object

**Value**

reactive holding code to be used in reports

---

interAnalysisUI      *inter Analysis Module - UI*

---

**Description**

inter Analysis Module - UI

**Usage**

```
interAnalysisUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

fluidRow containing the ui code



**Author(s)**

Janina Reeder

**Examples**

```
interAnalysisUI("interanalysis_id")
```

---

`intraAnalysis`*Intra Analysis Module - server*

---

**Description**

Intra Analysis Module - server

**Usage**

```
intraAnalysis(  
  input,  
  output,  
  session,  
  data,  
  levelOpts,  
  chosenLevel,  
  resetInput,  
  aggData,  
  normalizedData  
)
```

**Arguments**

<code>input</code>	shiny input
<code>output</code>	shiny output
<code>session</code>	shiny session
<code>data</code>	the main data object returned from <code>data_input_module</code>
<code>levelOpts</code>	available levels to aggregate on (depends on input data)
<code>chosenLevel</code>	previously selected level (passed from different instance)
<code>resetInput</code>	reactive boolean determining if reset is required
<code>aggData</code>	the aggregated MRExperiment object
<code>normalizedData</code>	boolean indicating if normalization was done

**Value**

reactive holding code to be used in reports

**Author(s)**

Janina Reeder

---

intraAnalysisUI      *Intra Analysis Module - UI*

---

**Description**

Intra Analysis Module - UI

**Usage**

```
intraAnalysisUI(id)
```

**Arguments**

id                    namespace identifier

**Value**

fluidRow containing the ui code

**Author(s)**

Janina Reeder

**Examples**

```
intraAnalysisUI("intraanalysis_id")
```

---

intraInput              *Server side for the intra analysis input module*

---

**Description**

Server side for the intra analysis input module

**Usage**

```
intraInput(  
  input,  
  output,  
  session,  
  meData,  
  facetOptions = NULL,  
  reset,  
  aggDat = reactive(NULL)  
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
meData	MRExperiment object storing all data
facetOptions	named vector of available facet choices
reset	reactive boolean determining if all inputs should be reset
aggDat	aggregated MRExperiment object (default is NULL)

**Value**

list holding all chosen values and the selected feature

**Author(s)**

Janina Reeder

---

intraInputUI	<i>Main intra analysis input module. Set up to handle all analysis tabs in the app depending on given parameters</i>
--------------	--

---

**Description**

Main intra analysis input module. Set up to handle all analysis tabs in the app depending on given parameters

**Usage**

```
intraInputUI(id)
```

**Arguments**

id	element identifier - namespace
----	--------------------------------

**Value**

box containing ui element

**Author(s)**

Janina Reeder

---

`longAnalysis`*long Analysis Module - server*

---

**Description**

long Analysis Module - server

**Usage**

```
longAnalysis(  
  input,  
  output,  
  session,  
  data,  
  levelOpts,  
  chosenLevel,  
  resetInput,  
  aggData,  
  normalizedData  
)
```

**Arguments**

<code>input</code>	shiny input
<code>output</code>	shiny output
<code>session</code>	shiny session
<code>data</code>	the main data object returned from <code>data_input_module</code>
<code>levelOpts</code>	available levels to aggregate on (depends on input data)
<code>chosenLevel</code>	previously selected level (passed from longerent instance)
<code>resetInput</code>	reactive boolean determining if reset is required
<code>aggData</code>	the aggregated MRExperiment object
<code>normalizedData</code>	boolean indicating if normalization was done

**Value**

reactive holding code to be used in reports

**Author(s)**

Janina Reeder

---

longAnalysisUI	<i>Long Analysis Module - UI</i>
----------------	----------------------------------

---

**Description**

Long Analysis Module - UI

**Usage**

```
longAnalysisUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

fluidRow containing the ui code

**Author(s)**

Janina Reeder

**Examples**

```
longAnalysisUI("longanalysis_id")
```

---

longInput	<i>Server side for the analysis input module handling analysis control</i>
-----------	--

---

**Description**

Server side for the analysis input module handling analysis control

**Usage**

```
longInput(  
  input,  
  output,  
  session,  
  meData,  
  facetOptions = NULL,  
  reset,  
  aggDat = reactive(NULL)  
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
meData	MExperiment object storing all data
facetOptions	named vector of available facet choices
reset	reactive boolean determining if all inputs should be reset
aggDat	aggregated MExperiment

**Value**

list holding all chosen values and the selected feature

**Author(s)**

Janina Reeder

---

longInputUI	<i>Main diffanalysis input module. Set up to handle diff analysis tabs in the app depending on given parameters</i>
-------------	---

---

**Description**

Main diffanalysis input module. Set up to handle diff analysis tabs in the app depending on given parameters

**Usage**

```
longInputUI(id)
```

**Arguments**

id	element identifier - namespace
----	--------------------------------

**Value**

box containing ui element

**Author(s)**

Janina Reeder

---

`longResults`*Longitudinal analysis module server code*

---

**Description**

Longitudinal analysis module server code

**Usage**

```
longResults(  
  input,  
  output,  
  session,  
  aggDat,  
  featLevel,  
  longSettings,  
  normalizedData,  
  reset  
)
```

**Arguments**

<code>input</code>	shiny input
<code>output</code>	shiny output
<code>session</code>	shiny session
<code>aggDat</code>	aggregated MRExperiment
<code>featLevel</code>	chosen feature level (aggregation level)
<code>longSettings</code>	reactive storing values selected in analysis input interface
<code>normalizedData</code>	reactive boolean indicating if data has been normalized
<code>reset</code>	boolean reactive which resets the module if TRUE

**Value**

list containing R code for analysis and for feature plots

**Author(s)**

Janina Reeder

---

longResultsUI	<i>Longitudinal Analysis module UI</i>
---------------	--

---

**Description**

Longitudinal Analysis module UI

**Usage**

```
longResultsUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

row containing the UI elements

**Author(s)**

Janina Reeder

---

makeQCPlot	<i>Plots sequencing statistics scatterplot</i>
------------	--

---

**Description**

This function makes a scatterplot of read and feature counts for each sample. It was adjusted based on original work by Mo Huang

**Usage**

```
makeQCPlot(  
  MObj,  
  col_by = NULL,  
  log = "none",  
  filter_feat = 0,  
  filter_read = 0,  
  allowWebGL = TRUE,  
  pwidth = 550,  
  pheight = 550  
)
```



**Arguments**

MRobj	metagenomeSeq object to be plotted
col_by	factor by which to color the points
log	character indicating which (if any) axes should be shown as log
filter_feat	Numeric Y-coordinate to draw horizontal dashed line to indicate feature filtering. If 0 (default), no line is drawn.
filter_read	Numeric X-coordinate to draw vertical dashed line to indicate read count filtering. If 0 (default), no line is drawn.
allowWebGL	boolean indicating if webGL should be added
pwidth	overall plot width; default is 550 (125 are added for legend)
pheight	overall plot height; default is 550

**Value**

the plotly QC plot

**Author(s)**

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
makeQCPlot(mouseData)
```

---

normalizeData	<i>Calls appropriate normalization functions depending on input parameter The two available methods included in the package are based on either calculating proportions or by using cumulative sum scaling (CSS), Paulson, et al. Nat Meth 2013.</i>
---------------	--

---

**Description**

Calls appropriate normalization functions depending on input parameter The two available methods included in the package are based on either calculating proportions or by using cumulative sum scaling (CSS), Paulson, et al. Nat Meth 2013.

**Usage**

```
normalizeData(MRobj, norm_method)
```

**Arguments**

MRobj	the MRExperiment
norm_method	method to use for normalization; CSS or Proportional

**Value**

the normalized MRObj

**Examples**

```
data("mouseData", package = "metagenomeSeq")
normalizeData(mouseData, norm_method = "CSS")
```

---

parseInteractionName *Helper function used to build a correct interactionName based on the chosen columns*

---

**Description**

Helper function used to build a correct interactionName based on the chosen columns

**Usage**

```
parseInteractionName(interactionName)
```

**Arguments**

interactionName  
as chosen by user. This may not be good to store internally

**Value**

updated interactionName or warning/error string

---

phenotypeCorr *Phenotype correlation analysis server module*

---

**Description**

Phenotype correlation analysis server module

**Usage**

```
phenotypeCorr(  
  input,  
  output,  
  session,  
  aggDat,  
  colorOptions,  
  corFeatBase,  
  corPheno,  
  corFacet1,  
  corFacet2,  
  corMethod,  
  reset  
)
```

**Arguments**

input	shiny input
output	shiny output
session	shiny session
aggDat	aggregated MRExperiment
colorOptions	reactive storing filters available via data input
corFeatBase	first correlation feature
corPheno	correlation phenotype
corFacet1	first correlation facet
corFacet2	second correlation facet
corMethod	correlation method to use
reset	boolean reactive which resets the module if TRUE

**Value**

R code used to do the correlation analysis (character)

**Author(s)**

Janina Reeder

---

phenotypeCorrUI      *Phenotype correlation analysis module*

---

**Description**

Phenotype correlation analysis module

**Usage**

```
phenotypeCorrUI(id)
```

**Arguments**

id                    namespace identifier

**Value**

box containing the UI element

**Author(s)**

Janina Reeder

---

phenotypeTable      *Phenotype table server module*

---

**Description**

Phenotype table server module

**Usage**

```
phenotypeTable(input, output, session, meData, phenoModRep, addPheno)
```

**Arguments**

input                shiny input  
output                shiny output  
session               shiny session  
meData                MRExperiment storing the data  
phenoModRep        reactive Value storing any phenotable modifications made  
addPheno             reactive boolean keeping track of pheno data modifications

**Value**

phenotype table server fragment - no return value

**Author(s)**

Janina Reeder

---

phenotypeTableUI      *Phenotype table UI module*

---

**Description**

Phenotype table UI module

**Usage**

phenotypeTableUI(id)

**Arguments**

id                    namespace identifier

**Value**

fluidRow holding the ui code

**Author(s)**

Janina Reeder

**Examples**

phenotypeTableUI("phenotype\_id")

---

plotAbundance	<i>Plot relative abundance</i>
---------------	--------------------------------

---

### Description

This function plots the relative abundance of the top abundant features.

### Usage

```
plotAbundance(  
  aggdat,  
  level,  
  x_var = "SAMPLE_ID",  
  ind = seq_len(10),  
  plotTitle = "",  
  ylab = "Reads",  
  facet1 = NULL,  
  facet2 = NULL,  
  source = "A",  
  pwidth = 650,  
  pheight = 150  
)
```

### Arguments

aggdat	aggregated MRExperiment object
level	Feature level.
x_var	Phenotype to aggregate over on X-axis. Default by "SAMPLE_ID".
ind	Indices of top abundant features to plot. Rest of features are aggregated and displayed as "other".
plotTitle	Plot title. Default shows no title.
ylab	Y-axis label. Default is "Reads"
facet1	Phenotype for facet 1.
facet2	Phenotype for facet 2.
source	name of the plot (needed for event handling); default is "A"
pwidth	overall plot width; default is 650
pheight	overall plot height; default is 150

### Value

plotly plot

### Author(s)

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
plotAbundance(aggdat, level = "genus", x_var = "diet")
```

plotAlpha

*Plot alpha diversity***Description**

This function plots the alpha diversity. See `?vegan::diversity` for details on the available index

**Usage**

```
plotAlpha(
  aggdat,
  level,
  index = c("shannon", "simpson", "invsimpson", "richness"),
  x_var = "SAMPLE_ID",
  ylab = index,
  col_by = NULL,
  facet1 = NULL,
  facet2 = NULL,
  plotTitle = "",
  pwidth = 500,
  pheight = 150
)
```

**Arguments**

aggdat	aggregated MRExperiment
level	Feature level
index	Diversity index, one of "shannon", "simpson", "invsimpson" or "richness" (=number of features). Default is "shannon".
x_var	Phenotype to aggregate over on X-axis. Default by "SAMPLE_ID".
ylab	Y-axis label. Default is "Reads".
col_by	Phenotype for coloring.
facet1	Phenotype for facet 1.
facet2	Phenotype for facet 2.
plotTitle	Plot title. By default, no title is used.
pwidth	overall plot width; default is 650
pheight	overall plot height; default is 150

**Value**

plotly plot object

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
plotAlpha(aggdat, level = "genus", index = "shannon", x_var = "diet")
```

---

plotAvgAbundance	<i>Plot average relative abundance</i>
------------------	--

---

**Description**

This function plots the average relative abundance of the top abundant features.

**Usage**

```
plotAvgAbundance(
  aggdat,
  level,
  ind = seq_len(10),
  plotTitle = "",
  ylab = "Reads",
  facet1 = NULL,
  facet2 = NULL,
  source = "A",
  pwidth = 500,
  pheight = 150
)
```

**Arguments**

aggdat	aggregated MRExperiment object
level	Feature level.
ind	Indices of top abundant features to plot. Rest of features are aggregated and displayed as "other".
plotTitle	Plot title. Default shows no title.
ylab	Y-axis label. Default is "Reads"
facet1	Phenotype for facet 1.
facet2	Phenotype for facet 2.
source	name of the plot (needed for event handling); default is "A"
pwidth	overall plot width; default is 500
pheight	overall plot height; default is 150



**Value**

plotly plot

**Author(s)**

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
plotAvgAbundance(aggdat, level = "genus")
```

---

plotBeta

*Plot beta diversity*

---

**Description**

This functions plots the beta diversity as a PCoA plot.

**Usage**

```
plotBeta(
  aggdat,
  dim = c(1, 2),
  log = TRUE,
  dist_method = "bray",
  pcas = NULL,
  nfeatures = nrow(aggdat),
  col_by = NULL,
  shape_by = NULL,
  plotTitle = "",
  xlab = NULL,
  ylab = NULL,
  pt_size = 8,
  plotText = NULL,
  confInterval = NULL,
  allowWebGL = TRUE,
  pwidth = 550,
  pheight = 550
)
```

**Arguments**

aggdat	aggregated MRExperiment
dim	Vector of length 2 specifying which dimensions to plot.
log	Log2 transform data. Default is TRUE.
dist_method	Which distance method to use. See ?vegan::vegdist for more <code>vegdist()</code> for options. Default is "bray".
pcas	precalculated pcas to avoid recalculation via CalcPCs
nfeatures	Number of top features in terms of standard deviation. Default is all.
col_by	Phenotype for coloring.
shape_by	Phenotype for shape.
plotTitle	Plot title. By default, becomes PCoA (codedist.method).
xlab	X-axis label. By default, shows dimension and percent variance explained.
ylab	Y-axis label. By default, shows dimension and percent variance explained.
pt_size	the size of the markers
plotText	adonis text to be added to plot
confInterval	numeric value indicating confidence level for ellipses
allowWebGL	boolean indicating if WebGL should be used
pwidth	overall plot width; default is 550 (125 are added for legend)
pheight	overall plot height; default is 550

**Value**

plotly plot object

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
plotBeta(aggdat)
```

---

plotHeatmap

*Plot heatmap*

---

**Description**

This function plots a heatmap of feature abundance.

**Usage**

```
plotHeatmap(  
  aggdat,  
  features = NULL,  
  log = TRUE,  
  sort_by = c("Fano", "MAD", "Variance"),  
  nfeat = 50,  
  col_by = NULL,  
  row_by = NULL,  
  plotTitle = ""  
)
```

**Arguments**

aggdat	aggregated MRExperiment
features	Vector of features to plot. If NULL, the top 'nfeat' features in terms of 'sort_by' will be plotted.
log	Log2 transform data. Default is TRUE.
sort_by	Dispersion measure to sort features, one of "Fano", "MAD", and "Variance"
nfeat	Number of features to display. Default is 50.
col_by	Vector of phenotypes for coloring.
row_by	Name of feature level for coloring.
plotTitle	Plot title. By default, no title.

**Value**

plotly heatmap

**Examples**

```
data("mouseData", package = "metagenomeSeq")  
aggdat <- aggFeatures(mouseData, level = "genus")  
plotHeatmap(aggdat, sort_by = "Fano")
```

---

plotLongFeature

*Plot longitudinal features*

---

**Description**

This function plots the reads of a particular feature over different time points.

**Usage**

```
plotLongFeature(
  aggdat,
  feature,
  x_var,
  id_var = "SAMPLE_ID",
  plotTitle = NULL,
  ylab = "Reads",
  log = FALSE,
  showLines = TRUE,
  fixedHeight = NULL,
  x_levels = NULL,
  pwidth = 650
)
```

**Arguments**

aggdat	aggregated MRExperiment
feature	Feature to plot.
x_var	Phenotype to show along on X-axis.
id_var	phenotype used to connect data points. Default is "SAMPLE_ID"
plotTitle	Plot title. Default shows no title.
ylab	Y-axis label. Default is "Reads"
log	Log2 transform data. Default is FALSE.
showLines	add lines between the points
fixedHeight	sets a specific plot height (differential analysis)
x_levels	restrict to specific levels of x_var (differential analysis)
pwidth	overall plot width; default is 650

**Value**

plotly object holding long feature plot

**Author(s)**

Janina Reeder, Mo Huang

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
plotLongFeature(aggdat, feature = "Prevotella", x_var = "diet",
  id_var = "mouseID")
```

---

plotlyHistogram      *Function plotting a plotly histogram on the given histvalue*

---

### Description

Function plotting a plotly histogram on the given histvalue

### Usage

```
plotlyHistogram(  
  histvalue,  
  plotTitle,  
  xaxisTitle = "",  
  yaxisTitle = "",  
  pwidth = 200,  
  pheight = 200  
)
```

### Arguments

histvalue	the value to plot as a histogram
plotTitle	title of the plot
xaxisTitle	name of xaxis; default is ""
yaxisTitle	name of yaxis; default is ""
pwidth	overall plot width; default is 200
pheight	overall plot height; default is 200

### Value

plotly plot object

### Examples

```
data("mouseData", package = "metagenomeSeq")  
plotlyHistogram(histvalue = colSums(MRcounts(mouseData) > 0),  
  plotTitle = "Feature distribution",  
  xaxisTitle = "features", yaxisTitle = "frequency")
```

---

plotlySampleBarplot     *Function plotting a barplot showing number of OTUs per samples*

---

### Description

Function plotting a barplot showing number of OTUs per samples

### Usage

```
plotlySampleBarplot(  
  MObj,  
  col_by = NULL,  
  xaxisTitle = "",  
  yaxisTitle = "",  
  pwidth = 600,  
  pheight = 450,  
  sortbyfreq = FALSE,  
  pheno_sort = NULL,  
  x_levels = NULL  
)
```

### Arguments

MObj	containing data to plot
col_by	phenotype to color bars by; default is NULL
xaxisTitle	name of xaxis; default is ""
yaxisTitle	name of yaxis; default is ""
pwidth	overall plot width; default is 600
pheight	overall plot height; default is 450
sortbyfreq	boolean determining if bars should be sorted by frequency; default is FALSE
pheno_sort	order of pheno levels to sort by; ignored if sortbyfreq is TRUE
x_levels	character vector holding x values in order to be shown

### Value

plotly plot object

### Examples

```
data("mouseData", package = "metagenomeSeq")  
plotlySampleBarplot(mouseData)
```

---

plotSingleFeature      *Plot features*

---

## Description

This function plots the reads of a particular feature or set of features.

## Usage

```
plotSingleFeature(  
  aggdat,  
  feature = "other",  
  x_var = "SAMPLE_ID",  
  ind = seq_len(10),  
  plotTitle = NULL,  
  ylab = "Reads",  
  xlab = NULL,  
  facet1 = NULL,  
  facet2 = NULL,  
  log = FALSE,  
  showPoints = FALSE,  
  fixedHeight = NULL,  
  x_levels = NULL,  
  pwidth = 500  
)
```

## Arguments

aggdat	aggregated MRExperiment
feature	Feature to plot.
x_var	Phenotype to aggregate over on X-axis. Default by "SAMPLE_ID".
ind	Indices of top abundant features to plot. Needed to determine appropriate color
plotTitle	Plot title. Default shows no title.
ylab	Y-axis label. Default is "Reads"
xlab	X-axis label. If NULL, x_var will be used as label.
facet1	Phenotype for facet 1.
facet2	Phenotype for facet 2.
log	Log2 transform data. Default is FALSE.
showPoints	add points for each sample on plot
fixedHeight	sets a specific plot height (differential analysis)
x_levels	restrict to specific levels of x_var (differential analysis)
pwidth	overall plot width; default is 650

**Value**

plotly plot object

**Author(s)**

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
plotSingleFeature(aggdat, feature = "Prevotella", x_var = "diet")
```

---

readData

*Reads in data*

---

**Description**

This function reads in an MRexperiment object saved as an RDS file, a Biom file, or a tab - delimited count matrix with features as rows and samples as columns.

**Usage**

```
readData(filepath, type = "RDS")
```

**Arguments**

filepath	Relative or absolute file path of data object.
type	The type of file to be read; default is "RDS", other options are "RDATA", "BIOM", "TAB", "CSV"

**Value**

An MRexperiment object.



---

relAbundance	<i>Relative abundance plot module - server</i>
--------------	--

---

### Description

Relative abundance plot module - server

### Usage

```
relAbundance(  
  input,  
  output,  
  session,  
  aggDat,  
  featLevel,  
  intraSettings,  
  normalizedData,  
  reset  
)
```

### Arguments

input	shiny input
output	shiny output
session	shiny session
aggDat	aggregated MRExperiment
featLevel	chosen feature level (aggregation level)
intraSettings	analysis input settings passed over to this module
normalizedData	boolean indicating whether data has been normalized
reset	boolean reactive which resets the module if TRUE

### Value

list storing plot clicks and number of features displayed (passed to feature plot module) as well as the R code to make plot

---

relAbundanceUI	<i>Relative abundance plot module - UI</i>
----------------	--

---

**Description**

Relative abundance plot module - UI

**Usage**

```
relAbundanceUI(id)
```

**Arguments**

id                    namespace identifier

**Value**

box containing the ui code

**Author(s)**

Janina Reeder

---

replaceWithUnknown	<i>Helper function to replace any un-annotated features with the term unknown</i>
--------------------	---

---

**Description**

Helper function to replace any un-annotated features with the term unknown

**Usage**

```
replaceWithUnknown(feacol)
```

**Arguments**

feacol                vector of entries to be replaced where needed (fData column)

**Value**

modified feacol

**Author(s)**

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
featcol <- fData(mouseData)[["genus"]]
featcol[featcol == "NA"] <- NA
replaceWithUnknown(featcol)
```

---

reportList

*Report tab module server*

---

**Description**

Report tab module server

**Usage**

```
reportList(
  input,
  output,
  session,
  dataSource,
  preprocessRep,
  qcRep,
  analysisRep,
  aggIndex,
  reset
)
```

**Arguments**

input	module input
output	module output
session	app session
dataSource	R code to obtain data for rendering
preprocessRep	R code containing preprocessing steps of data
qcRep	R Code to generate QC plots
analysisRep	R Code to generate all analyses saved to reports
aggIndex	boolean value representing aggregation steps in analysisRep
reset	boolean reactive which resets the module if TRUE

**Value**

report list server fragment - no return value

**Author(s)**

Janina Reeder

---

reportListUI	<i>report tab ui</i>
--------------	----------------------

---

**Description**

report tab ui

**Usage**

```
reportListUI(id)
```

**Arguments**

id	namespace identifier
----	----------------------

**Value**

fluidRow holding ui elements

**Author(s)**

Janina Reeder

**Examples**

```
reportListUI("reportlist_id")
```

---

reportRow	<i>Report Row</i>
-----------	-------------------

---

**Description**

Report Row

**Usage**

```
reportRow(input, output, session, type, content)
```

**Arguments**

input	module input
output	module output
session	app session
type	boolean indicating whether checkbox should be included
content	R code to show

**Value**

reactive boolean indicating whether row is selected

**Author(s)**

Janina Reeder

---

reportRowUI	<i>Report row module consisting of a checkbox, image and description/R code area</i>
-------------	--

---

**Description**

Report row module consisting of a checkbox, image and description/R code area

**Usage**

```
reportRowUI(id, type)
```

**Arguments**

id	namespace identifier
type	boolean indicating if a selector checkbox should be added

**Value**

div holding the UI code

**Author(s)**

Janina Reeder

---

rollDownFeatures	<i>Helper function which rolls down annotated from closest higher order with annotation</i>
------------------	---

---

**Description**

Helper function which rolls down annotated from closest higher order with annotation

**Usage**

```
rollDownFeatures(featrow)
```

**Arguments**

featrow	vector of entries to be replaced where needed (fData row)
---------	---

**Value**

modified featurerow

**Author(s)**

Janina Reeder

**Examples**

```
data("mouseData", package = "metagenomeSeq")
featrow <- fData(mouseData)[5,]
rollDownFeatures(featrow)
```

---

runDiffTest

*Performs differential abundance testing*

---

**Description**

This function performs differential abundance testing between groups of a specified phenotype. Four methods are available: limma, Kruskal-Wallis, ZILN and DESeq2 (see details).

**Usage**

```
runDiffTest(
  aggdat,
  level,
  phenotype,
  phenolevels = NULL,
  log = TRUE,
  coef = NULL,
  method = c("limma", "Kruskal-Wallis", "ZILN", "DESeq2")
)
```

**Arguments**

aggdat	aggregated MRExperiment
level	Feature level.
phenotype	Phenotype to test.
phenolevels	levels of the phenotype to restrict the comparison to
log	Log2 transform data. Default is TRUE.
coef	Numeric which indicates which pairwise comparison to analyze when there are more than two groups. Corresponds to the column number of the model matrix produced by <code>designPairs()</code> . If NULL, a test of any difference between all groups is performed.
method	Differential testing method. One of "limma" (default), "Kruskal-Wallis", "ZILN", or "DESeq2".

**Details**

limma is a differential expression tool for microarray data using linear models. It can also be applied to microbiome data.

The Kruskal-Wallis test is a non-parametric rank test which examines if groups come from the same distribution. A significant result indicates at least one group is distributionally different than another group.

ZILN is a zero-inflated log-normal model implemented in `fitFeatureModel()` of the `metagenomeSeq` package.

DeSeq2 performs differential gene expression analysis based on the negative binomial distribution

**Value**

data.frame holding results of the differential analysis

**Examples**

```
data("mouseData", package = "metagenomeSeq")
aggdat <- aggFeatures(mouseData, level = "genus")
runDiffTest(aggdat = aggdat, level = "genus",
            phenotype = "diet", method = "Kruskal-Wallis")
```

---

`runMicrobiomeExplorer` *Main function to start the Microbiome Explorer Shiny app via a command line call*

---

**Description**

Main function to start the Microbiome Explorer Shiny app via a command line call

**Usage**

```
runMicrobiomeExplorer()
```

**Value**

the shiny application

# Index

abundanceHeatmap, 4  
abundanceHeatmapUI, 5  
add\_plotly\_config, 7  
add\_plotly\_layout, 7  
addFeatData, 6  
addPhenoData, 6  
aggFeatures, 8  
aggregationTab, 8  
aggregationTabUI, 9  
alphaDiversity, 10  
alphaDiversityUI, 11  
avgAbundance, 11  
avgAbundanceUI, 12

betaDiversity, 13  
betaDiversityUI, 14  
betaInput, 14  
betaInputUI, 15  
buildEmptyPlotlyPlot, 15  
buildPlottingDF, 16

calculatePCAs, 17  
computeCI\_Interval, 17  
computeDistMat, 18  
corrAnalysis, 18  
corrAnalysisUI, 19  
corrFeature, 20  
corrInput, 21  
corrInputUI, 22  
corrPhenotype, 23  
createHeader, 24

dataInput, 25  
dataInputUI, 26  
designPairs, 26, 78  
diffAnalysis, 27  
diffAnalysisUI, 28  
diffInput, 28  
diffInputUI, 29  
diffTable, 29

diffTableUI, 30

extendPhenoData, 31

featAbundance, 31  
featAbundanceUI, 32  
featureAnalysis, 33  
featureAnalysisUI, 34  
featureCorr, 34  
featureCorrUI, 35  
featureInput, 36  
featureInputUI, 37  
featureTable, 37  
featureTableUI, 38  
fileUpload, 39  
fileUploadUI, 40  
filterByPheno, 40  
filterMEData, 41  
fitFeatureModel, 79

generateReport, 42  
getFeatModCode, 43  
getFileType, 43  
getFilterChoices, 44  
getLegendLevel, 44  
getPhenoChanges, 45  
getPhenoModCode, 45  
getWidths, 46

heatmapInput, 46  
heatmapInputUI, 47

interAnalysis, 47  
interAnalysisUI, 48  
intraAnalysis, 49  
intraAnalysisUI, 50  
intraInput, 50  
intraInputUI, 51

longAnalysis, 52  
longAnalysisUI, 53



longInput, [53](#)  
longInputUI, [54](#)  
longResults, [55](#)  
longResultsUI, [56](#)

makeQCPlot, [56](#)

normalizeData, [57](#)

parseInteractionName, [58](#)  
phenotypeCorr, [58](#)  
phenotypeCorrUI, [60](#)  
phenotypeTable, [60](#)  
phenotypeTableUI, [61](#)  
plotAbundance, [62](#)  
plotAlpha, [63](#)  
plotAvgAbundance, [64](#)  
plotBeta, [65](#)  
plotHeatmap, [66](#)  
plotLongFeature, [67](#)  
plotlyHistogram, [69](#)  
plotlySampleBarplot, [70](#)  
plotSingleFeature, [71](#)

readData, [72](#)  
relAbundance, [73](#)  
relAbundanceUI, [74](#)  
replaceWithUnknown, [74](#)  
reportList, [75](#)  
reportListUI, [76](#)  
reportRow, [76](#)  
reportRowUI, [77](#)  
rollDownFeatures, [77](#)  
runDiffTest, [78](#)  
runMicrobiomeExplorer, [79](#)

vegdist, [66](#)