Package: OpenStats (via r-universe)

November 19, 2026

Type Package

Title A Robust and Scalable Software Package for Reproducible Analysis of High-Throughput genotype-phenotype association

Version 1.24.0

Author Hamed Haseli Mashhadi

Maintainer Marina Kan <marinak@ebi.ac.uk>

Description Package contains several methods for statistical analysis of genotype to phenotype association in high-throughput screening pipelines.

License GPL (>= 2)

Imports MASS, jsonlite, Hmisc, methods, knitr, AICcmodavg, car, rlist, summarytools, graphics, stats, utils

Depends nlme

Encoding UTF-8

RoxygenNote 6.0.1

BugReports https://git.io/Jv5wg

URL https://git.io/Jv5w0

biocViews StatisticalMethod, BatchEffect, Bayesian

Suggests rmarkdown

VignetteBuilder knitr

Config/pak/sysreqs cmake make libmagick++-dev gsfonts libicu-dev libuv1-dev libxml2-dev libssl-dev

Repository Bioconductor 3.23

Date/Publication 2026-04-28 12:52:31 UTC

RemoteUrl https://github.com/bioc/OpenStats

RemoteRef RELEASE_3_23

RemoteSha 225404e2f87436fa91d32d7699e4183c4b08bca3

Contents

OpenStatsAnalysis 2
OpenStatsComplementarySplit 8
OpenStatsList 10
OpenStatsListBuilder 12
OpenStatsReport 14
plot.OpenStatsComplementarySplit 15
plot.OpenStatsFE 16
plot.OpenStatsList 17
plot.OpenStatsMM 18
plot.OpenStatsRR 20
print.OpenStatsComplementarySplit 21
print.OpenStatsFE 22
print.OpenStatsList 23
print.OpenStatsMM 24
print.OpenStatsRR 26
summary.OpenStatsComplementarySplit 27
summary.OpenStatsFE 28
summary.OpenStatsList 30
summary.OpenStatsMM 30
summary.OpenStatsRR 32

Index **34**

`OpenStatsAnalysis` *Method "OpenStatsAnalysis"*

Description

The driver function in the OpenStats package for running statisical analysis on phenotypic data.

- It performs several checks on the data and the input model before performing the analysis. This function supports three main analysis frameworks precisely, Linear Mix model (MM), Fisher's Exact test (FE) and Reference Range plus (RR).
- It further monitors the process for failures and errors and applies some runtime patches/fixes.
- The function parameters are designed to be human-friendly by initialising the inputs by the model that will be applied to the data.

Usage

```
OpenStatsAnalysis(
 OpenStatsListObject    = NULL,
 method     = NULL,
 MM_fixed   = TypicalModel(
  depVariable = "data_point",
  withWeight  = MM_BodyWeightIncluded,
  Sex = TRUE,
```

```
 LifeStage = TRUE,
 data = OpenStatsListObject@datasetPL,
 others = NULL,
 debug = debug
),
MM_random = rndProce("TYPICAL"),
MM_BodyWeightIncluded = TRUE,
MM_lower  = ~ Genotype + 1,
MM_weight = if (
 TermInModelAndnLevels(
    model = MM_fixed,
    data = OpenStatsListObject@datasetPL
    )
  ){
 varIdent(form = ~ 1 |  LifeStage)
}else{
 varIdent(form = ~ 1 |  Genotype)
 },
MM_direction = "both",
MM_checks    = c(TRUE, TRUE, TRUE, TRUE),
MM_optimise  = c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE),
FE_formula   = category ~   Genotype + Sex + LifeStage,
RR_formula   = data_point ~ Genotype + Sex + LifeStage,
RRrefLevel = 'control',
RR_prop      = 0.95,
FERR_rep     = 1500,
FERR_FullComparisions = c(TRUE, FALSE),
MMFERR_conf.level = 0.95,
debug        = TRUE,
...
)
```

Arguments

OpenStatsListObject	mandatory argument. An instance of the 'OpenStatsList' or 'PhenList' (former from the Bioconductor 'PhenStat' package).
method	Must be specified by the user. A character string ("MM", "FE" or "RR") defining the method to use for model building.
MM_fixed	Only applies to the "MM" framework. A formula that specifies the fixed effect in the linear mix model. The default is data_point~Genotype+Sex+LifeStage +/- BodyWeight. Note that the algorithm checks and removes the formula terms that do not exist in data.
MM_random	Only applies to the "MM" framework. The random effect in the linear mixed model. See the lme() function for more details about the formula. The default is ~1|Batch.
MM_BodyWeightIncluded	Only applies to the 'default' MM_fixed in the "MM" framework. If TRUE then

the default model includes the body weight and the model would be: data_point~Genotype+Sex+LifeStage + Weight.

`MM_lower` Only applies to the "MM" framework. A right-sided formula, for example ~Genotype+1 or ~Sex+Genotype+1 or ~Sex+Genotype+Sex:Genotype. The lowest model that must not be included in the model optimisation. In other words, the terms in this model won't be removed during the optimisation process. The default is ~ Genotype + 1 that is the genotype effect and the intercept will be kept in the model during the optimisation process.

`MM_weight` Only applies to the "MM" framework. From weight in the lme() manual:
"an optional varFunc object or one-sided formula describing the within-group heteroscedasticity structure. If given as a formula, it is used as the argument to varFixed, corresponding to fixed variance weights. See the documentation on varClasses for a description of the available varFunc classes. Defaults to NULL, corresponding to homoscedastic within-group errors".

The default is varIdent(form = ~ 1 | LifeStage) if the LifeStage included in the input data. Otherwise, varIdent(form = ~ 1 | Genotype).

`MM_direction` Only applies to the "MM" framework. Select from "both" (for stepwise optimisation), "backward" (for backward elimination) or "forward" (for forward selection) for the optimisation algorithm. The default is "both".

`MM_checks` Only applies to the "MM" framework. A vector of four 1/0 or TRUE/FALSE values such as c(TRUE, TRUE, TRUE, TRUE)[default]. Performing pre checks on the input model for some known scenarios. The first element of the vector activates checks on the model terms (See MM_fixed) to be existed in data. The second term removes any single level -factor- from the model (in MM_fixed). The third term removes the single value (such as a column of constants/no variation) from the -continuous- terms in the model (in MM_fixed). The Fourth element checks the interaction term to make sure all interactions have some data attached. Caution is needed for this check as it may take longer than usual if the formula in MM_fixed contains many factors. The default is c(TRUE, TRUE, TRUE, TRUE) that is all checks perform.

* Note that the function always removes duplicated columns in the dataset prior to applying the lme/gls.
* Regardless of the 'check' settings, the function always checks for the existence of the 'MM_random' terms (given it is not set to NULL) in the input data

`MM_optimise` Only applies to the "MM" framework. A vector of six binary values such as c(1,1,1,1,1,1) or c(TRUE, TRUE, TRUE, TRUE, TRUE, TRUE) (default). The first element of the vector activates the fixed effect optimisation. The algorithm uses AICc to optimise the fixed effects (Check 'AICcmodavg' package for more details about AICc). The second and third elements of the vector activate optimisation on 'weight' and 'random effects' respectively. The optimisation of weight and random effects refers to comparing the AICc between a model with and without those effects. The fourth element activates the Split model effects (for example, separate male and female effects) (see 'SplitModels' in the output object). The fifth effect activates the effect size estimation (see 'Effect sizes' in

the output object). The sixth element activates the normality tests on the residuals (see 'ResidualNormalityTests' in the output object).

`FE_formula` Only applies to the "FE" framework. The model for analysing the categorical data. The default is: category ~ Genotype + Sex + LifeStage.
Note that similar to MM_fixed, the terms that do not exist in data or the interaction terms will be dismissed from the model.

`RR_formula` Only applies to the "RR" framework. The model for analysing the RR+ compatible data.
** Important. The first term on the right hand side of the formula specifies the variable for discritising the response.
The default is: data_point ~ Genotype + Sex + LifeStage.
Note that similar to MM_fixed, the terms that do not exist in data or the interaction terms will be dismissed from the model.

`RRrefLevel` Only applies to the "RR" framework. A single term for the 'reference level' in the 'reference variable' (the first term on the right hand side of the 'RR_formula') used for discritising the response. If left blank then the a level with more observations will be considered as the reference level. The default is 'control'.

`RR_prop` Only applies to the "RR" framework. A single value between (0.5,1) not including the boundaries. The threshold for the variation ranges in the RR framework. The default value is 0.95.

`FERR_rep` Only applies to the "RR" or "FE" frameworks. The number of iteration for the Monte Carlo Fisher's Exact test. See "B" parameter in 'fisher.test()' function. Set to 0 for non-bayesian results (not recommended). The default is 1500.

`FERR_FullComparisions`
Only applies to the "RR" or "FE" frameworks. A vector of two logical flags, default c(TRUE,FALSE). Setting the first value to TRUE, then all combinations of the effects (all levels of factors in the input model - for example Male_LifeStage, Male_Genotype, Male_Mutant, Male_control, Female_control, Female_Mutant, Female_LifeStage and so on) will be tested. Otherwise only main effects (no sub levels - for example Sex_LifeStage [not for instance Male_LifeStage]) will be tested. Setting the second element of the vector to TRUE (default FALSE) will force Fisher's Exact test to do all comparisions between different levels of the RESPONSE variable. For example, if the respose has three levels such as 1.positive, 2.negative and 3.neutral then the comparisions will be between 1\&2, 1\&3, 2\&3 and 1\&2\&3 (obviously this is the full table).

`MMFERR_conf.level`
Applies to all frameworks (MM, FE, RR). Single numeric value for the interval confidence level. Default is 0.95

`debug` A logical flag. Set to TRUE to see more details about the progress of the function. Default TRUE

`...` Other parameters that can be passed to:
~> If the model is set to Linear Mixed Model (MM) then the parameters that can be passed to 'lme' function. See ?lme() manual page
~> If the model is either Fisher's Exact test (FE) or Reference Range + (RR) then the parameters that can be passed to the 'fisher.test' function. See ?fisher.test() manual page

Details

OpenStatsReport function can be used to extract the key elements of the analysis from the OpenStatsMM/FE/RR objects. The output from OpenStatsReport has schemed that makes it easy to be populated to the downstream processes such as storing and accessing results from a database.

Value

1. Successful execution of the function will return a list of three elements:

`input`	This contains the list of inputs
`output`	A list of outputs
`extra`	A placeholder for extra information if exists

2. If the function fails:

`messages`	A placeholder for the errors/warnings in the case of failure

See Also

OpenStatsListOpenStatsComplementarySplit, plot.OpenStatsMM,plot.OpenStatsFE,plot.OpenStatsRR, summary.OpenStatsMM, summary.OpenStatsFE,summary.OpenStatsRR, print.OpenStatsMM, print.OpenStatsFE, print.OpenStatsRR

Examples

```
######################################################################
# 1 Data preparation
######################################################################
#################
# 1.1 Continuous data - Creating OpenStatsList object
#################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)
#################
# 1.2 Categorical data - Creating OpenStatsList object
#################
fileCat <- system.file("extdata", "test_categorical.csv", package = "OpenStats")
test_Cat <- OpenStatsList(
  dataset = read.csv(fileCat, na.strings = "-"),
  testGenotype = "Aff3/Aff3",
  refGenotype = "+/+",
  dataset.colname.genotype = "Genotype",
```

```
  dataset.colname.batch = "Assay.Date",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "Weight",
  dataset.colname.sex = "Sex"
)
######################################################################
# 2 Testing frameworks
######################################################################

#################
# 2.1 Optimised Linear Mixed model (MM) framework
#################
MM1_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "MM",
  MM_fixed = data_point ~ Genotype + Weight
)
VO_MM1 <- OpenStatsReport(MM1_result)
plot(MM1_result, col = 2, main = "Optimised model")
summary(MM1_result)


#################
# 2.2 Linear Mixed model (MM) with NO optimisation
# for the fixed effects but random/weight effects
#################
MM2_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "MM",
  MM_fixed = data_point ~ Genotype + Weight,
  MM_lower = ~ Genotype + Weight + 1
  # Or simply MM_optimise = c(0, 1, 1, 1, 1, 1)
)
VO_MM2 <- OpenStatsReport(MM2_result)
plot(MM2_result, col = 8, main = "No optimisation on the fixed effects")
summary(MM2_result)

#################
# 2.3 Linear Mixed model (MM) with NO optimisation on the model
#################
MM3_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "MM",
  MM_fixed = data_point ~ Genotype + Weight,
  MM_optimise = c(0, 0, 0, 1, 1, 1)
)
VO_MM3 <- OpenStatsReport(MM3_result)
plot(MM3_result, col = 3, main = "Not optimised model")
summary(MM3_result)


#################
# 2.4 Reference range framework
```

```
#################
RR_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "RR",
  RR_formula = data_point ~ Genotype + Sex
)
VO_RR <- OpenStatsReport(RR_result)
plot(RR_result, col = 3:4)
summary(RR_result)


#################
# 2.5 Fisher's exact test framework
#################
FE_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cat,
  method = "FE",
  FE_formula = Thoracic.Processes ~ Genotype + Sex
)
VO_FE <- OpenStatsReport(FE_result)
plot(FE_result, col = 1:2)
summary(FE_result)
```

`OpenStatsComplementarySplit` *Method "OpenStatsComplementarySplit"*

Description

This function splits the input data according to the defined values in the ‘variables‘ parameter and runs separate analyses on the split datasets. For example, the default split, c("Sex", "LifeStage"), creates independent input data for Males (only), Females, Early, Late, Male.Early, Males.Late, Females.Early, Females.Late and analyses these datasets separately.

Usage

```
OpenStatsComplementarySplit(
 object    = NULL,
 variables = c("Sex", "LifeStage"),
 debug     = FALSE
)
```

Arguments

`object`	Mandatory argument. An instance of the 'OpenStatsAnalysis' object under the MM (linear mixed model) framework.
`variables`	Vector of names. A vector of variable names that will be fed into the split engine. The default is ‘c("Sex", "LifeStage")‘ that should report the results for the following categories: Males, Females, Early, Late, Male.Early, Males.Late, Females.Early, Females.Late.

debug Logical flag. Set to TRUE to see the analysis log. Default FALSE.

Value

List of splits and the analysis outputs. The splits contain the name of the partitioning levels (for example Female or Female.Age_15_weeks), and an 'OpenStatsAnalysis' object including the input data, outputs etc. See the examples for a general view of the output object.

See Also

OpenStatsAnalysis, plot.OpenStatsComplementarySplit, summary.OpenStatsComplementarySplit

Examples

```
######################################################################
# Data preparation
# - Continuous data - Creating OpenStatsList object
######################################################################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)
######################################################################
# Analysis
# -  Optimised Linear Mixed Model (MM) framework
######################################################################
MM_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "MM",
  MM_fixed = data_point ~ Genotype + Weight
)
######################################################################
# Split on Sex
# ** This split is already available from the normal running of OpenStatsAnalysis
######################################################################
output <- OpenStatsComplementarySplit(object = MM_result, variables = "Sex")
# Structure of the output object
lapply(output, names)
# Summaries
summary(output, format = "pandoc") # See knitr:kable function for more formats
# Plots
plot(output, ask = TRUE)
```

`OpenStatsList` *Method "OpenStatsList"*

Description

The driver function to create 'OpenStatsList' object from a data frame.

- The mandatory variable for creating a 'standard' OpenStatsList objects is 'Genotype'. Having two levels in the 'Genotype' field is mandatory. The function further checks for the optional 'Sex' with two levels (Male/Female), LifeStage' with two levels (Early/Late), 'Batch' (defined as date_of experiment in the IMPC) and 'Weight' (defined as animal body weight in the IMPC) and reports any abnormality in the data.

- For advance applications, the function is capable of creating a 'OpenStatsList' object without performing checks. To do this, set clean.dataset to FALSE.

Usage

```
OpenStatsList(
 dataset                                                ,
 testGenotype                = 'experimental'           ,
 refGenotype                 = 'control'                ,
 hemiGenotype                = NULL                     ,
 clean.dataset               = TRUE                     ,
 dataset.colname.genotype    = 'biological_sample_group' ,
 dataset.colname.sex         = 'sex'                    ,
 dataset.colname.batch       = 'date_of_experiment'     ,
 dataset.colname.lifestage   = 'LifeStage'              ,
 dataset.colname.weight      = 'weight'                 ,
 dataset.values.missingValue = c(' ','')                ,
 dataset.values.male         = NULL                     ,
 dataset.values.female       = NULL                     ,
 dataset.values.early        = NULL                     ,
 dataset.values.late         = NULL                     ,
 debug                       = TRUE
)
```

Arguments

`dataset`	mandatory argument. data frame created from file or from another source. See notes for more details
`testGenotype`	mandatory argument. Defines the test genotype to be compared to the reference genotype. Default 'experimental'
`refGenotype`	defines the reference genotype; assigned default value is 'control'
`hemiGenotype`	optional argument. defines the genotype value for hemizygous that will be changed to test genotype value

`clean.dataset` logical flag. 'TRUE' activates all checks and modification on the input data. The overview of the checks is, existence of the variables, checking levels, missings and relabeling

`dataset.colname.genotype`
mandatory argument. Column name within dataset for the genotype. Default 'biological_sample_group'

`dataset.colname.sex`
optional argument. column name within dataset for the sex. Default 'sex'

`dataset.colname.batch`
optional argument. column name within dataset for the batch effect. Default 'date_of_experiment'

`dataset.colname.lifestage`
optional argument. column name within dataset for the life stage. Default 'LifeStage'

`dataset.colname.weight`
optional argument. column name within dataset for the body weight. Default 'weight'

`dataset.values.missingValue`
value used as missing value in the dataset. Default '(space)'.

`dataset.values.male`
value used to label "males" in the dataset

`dataset.values.female`
value used to label "females" in the dataset

`dataset.values.early`
value used to label "early life stage" in the dataset

`dataset.values.late`
value used to label "late life stage" in the dataset

`debug` A logical flag. Set to TRUE to see more details about the progress of the function. Default TRUE

Value

an instance of the OpenStatsList class. The S4 object contains:

1. raw data: 'OpenStatsListObject@datasetUNF'
2. polished 'data: OpenStatsListObject@datasetPL'
3. the inputarguments to the 'OpenStatsList' function

Note

OpenStats allows a 'data.frame' for the input data. This data.frame can be formed from csv, tsv, txt etc. files and is organised with rows and columns for samples and features respectively. This allows a wide range of integration with other Bioconductor/CRAN packages, for instance, the output of Bioconductor 'SummarizedExperiment' package can be transformed and fed into OpenStats (note that SummarizedExperiment allows sample in columns and feature in rows that requires at least a transpose operation). Additionally, Bioconductor 'PhenStat' function 'PhenList' produces

very similar results to 'OpenStatsList' that allows direct processing of the 'PhenList' object by downstream OpenStats operational functions.

See Also

OpenStatsAnalysis,plot.OpenStatsList,summary.OpenStatsList, summary.OpenStatsList,

Examples

```
######################################################################
df <- read.csv(system.file("extdata", "test_continuous.csv", package = "OpenStats"))
######################################################################
# OpenStatsList object
######################################################################
OpenStatsList <- OpenStatsList(
  dataset = df,
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.sex = "sex",
  dataset.colname.weight = "weight"
)
p <- plot(OpenStatsList,
  vars = c(
    "Genotype",
    "Sex",
    "data_point",
    "age_in_days"
  )
)
p$Continuous
p$Categorical
summary(OpenStatsList, style = "grid")
class(OpenStatsList)
rm(OpenStatsList)
```

OpenStatsListBuilder *Method "OpenStatsListBuilder"*

Description

Specifying the age in days, this function creates a 'OpenStatsList' object from a 'PhenList' object from Bioconductor PhenStat package.

Usage

```
OpenStatsListBuilder(
  PhenListobject,
  DOE = NULL,
```

```
  DOB = NULL,
  d.threshold = 16 * 7,
  debug = TRUE
)
```

Arguments

PhenListobject	Mandatory argument. Instance of the 'PhenList' object from PhenStat package
DOE	Name of the data column for the 'Batch' in the 'PhenList' object. If left NULL then the input 'PhenList' object will be returned. Default NULL
DOB	Name of the data column for the 'date_of_birth' in the 'PhenList' object. If left NULL then the input 'PhenList' object will be returned. Default NULL
d.threshold	The threshold in age (DOE-DOB) to specify LifeStage early/late levels. The function uses as.Date(DOE)-as.Date(DOB) to calculate the age. The default is 16 weeks (16*7 days)
debug	Logical flag. Set to TRUE to see debug messages. Default TRUE

Value

Provided DOE and DOB are not NULL, a 'OpenStatsList' object that is quite similar to 'PhenList' object with an extra column called 'LifeStage' with two levels 'Early' and 'Late'. Otherwise, the output is similar to the input 'PhenList' object.

See Also

OpenStatsAnalysis, OpenStatsList

Examples

```
## Not run:
 library(PhenStat)
 file <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
 ######################################################################
 # PhenListObject from PhenStat package
 # The R package PhenStats must be installed in prior
 ######################################################################
 PhenListObject <- PhenList(
   dataset = read.csv(file),
   testGenotype = "experimental",
   refGenotype = "control",
   dataset.colname.batch = "date_of_experiment",
   dataset.colname.genotype = "biological_sample_group",
   dataset.colname.sex = "sex",
   dataset.colname.weight = "weight"
 )
 ######################################################################
 # OpenStatsList object
 ######################################################################
 OpenStatsListBuilder <- OpenStats:::OpenStatsListBuilder(
   PhenListobject = PhenListObject,
```

```
  DOE = "Batch",
  DOB = "Birth.Date",
  d.threshold = 99
 )
 plot(OpenStatsListBuilder)
 class(OpenStatsListBuilder)
 rm(OpenStatsListBuilder)

## End(Not run)
```

`OpenStatsReport` *Method "OpenStatsReport"*

Description

Wrapper for the output of 'OpenStatsAnalysis'. Returns model fitting and results in a list or JSON format (StatPacket).

Usage

```
OpenStatsReport(
 object                  ,
 othercolumns = NULL     ,
 JSON = FALSE            ,
 RemoveNullKeys = FALSE  ,
 ReportNullSchema = FALSE ,
 ...
)
```

Arguments

`object`	'Mandatory argument'. An instance of the OpenStatsAnalysis result object
`othercolumns`	A list of column names that must be included in the results. Default NULL
`JSON`	Logical flag. Setting to TRUE for the JSON (StatPacket) output otherwise, the function returns a list
`RemoveNullKeys`	Logical flag. Setting to TRUE will remove all NULL elements from the output. Default is FALSE
`ReportNullSchema`	logical flag. Setting to TRUE forces the function to return results even if the OpenStatsAnalysis returns a failure message
`...`	Other parameters that can be passed to 'toJSON()' function in the "jsonlite" library

Details

OpenStatsReport function can be used to extract the key elements of the analysis from the OpenStatsMM/FE/RR objects (the output from OpenStatsAnalysis function). The output from OpenStatsReport has schemed that makes it easy to be populated to the downstream processes such as storing and accessing results from a database.

Value

A list of values or a JSON object depends on the "JSON" parameter

See Also

OpenStatsAnalysis

Examples

```
example(OpenStatsAnalysis)
```

`plot.OpenStatsComplementarySplit` *plot for an 'OpenStatsComplementarySplit' object*

Description

This function visualises an 'OpenStatsComplementarySplit' object

Usage

```
 ## S3 method for class 'OpenStatsComplementarySplit'
plot(x, main = "Final Model", ask = FALSE, mfrow = c(2, 2), ...)
```

Arguments

`x`	an instance of 'OpenStatsComplementarySplit' result
`main`	a string to be pasted to the title of the plots
`ask`	see 'ask' in 'par()' function. Default FALSE
`mfrow`	the screen partition. see 'mfrow' argument in the 'par' function. Default c(2,2) then all plots display in one screen.
`...`	other parameters that can be passed to the 'plot' function

Details

The plot function creates some visualisations for the split results from the linear mixed model framework. Each level of partitioning variables (see 'variables' in the 'OpenStatsComplementarySplit' function manual) produces a set of plots listed below:

- Residual versus fitted values
- Residual density plot and the normality test p-value
- Residual Q-Q plot
- The density plot of the response variable and the normality test p-value

Value

Not applicable

See Also

OpenStatsComplementarySplit, print.OpenStatsComplementarySplit, summary.OpenStatsComplementarySplit

Examples

```
example(OpenStatsComplementarySplit)
```

`plot.OpenStatsFE` *plot for an 'OpenStatsFE' object*

Description

This function visualises an 'OpenStatsFE' object

Usage

```
 ## S3 method for class 'OpenStatsFE'
plot(x, main = "Mosaic plot", ask = FALSE, mfrow = c(2, 2), ...)
```

Arguments

`x`	an instance of 'OpenStatsFE' result from OpenStatsAnalysis(method = 'FE') function
`main`	a string to be pasted to the title of the plots
`ask`	see 'ask' in 'par()' function. Default FALSE
`mfrow`	the screen partition. see 'mfrow' argument in the 'par' function. Default c(2,2) then all plots display in one screen.
`...`	other parameters that can be passed to the 'plot' function

Details

The plot function creates some visualisations for the Fisher's exact test framework:

- Mosaic plot of the response versus Genotype/Sex/LifeStage (if they exist in the data
- Mosaic plot of the Sex versus Genotype (if they exist in the data)

Value

Not applicable

See Also

OpenStatsAnalysis, plot.OpenStatsRR, plot.OpenStatsMM

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Categorical data - Creating OpenStatsList object
#################
fileCat <- system.file("extdata", "test_categorical.csv", package = "OpenStats")
test_Cat <- OpenStatsList(
  dataset = read.csv(fileCat, na.strings = "-"),
  testGenotype = "Aff3/Aff3",
  refGenotype = "+/+",
  dataset.colname.genotype = "Genotype",
  dataset.colname.batch = "Assay.Date",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "Weight",
  dataset.colname.sex = "Sex"
)
#################
# Fisher's exact test framework
#################
FE_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cat,
  method = "FE",
  FE_formula = Thoracic.Processes ~ Genotype + Sex
)
plot(FE_result, col = 1:2)
```

---

plot.OpenStatsList *plot for an 'OpenStatsList' object*

Description

This function visualises an 'OpenStatsList' object

Usage

```
## S3 method for class 'OpenStatsList'
plot(x, vars = NULL, ...)
```

Arguments

x	OpenStatsList object
vars	Variable(s) of interest. The default is 'Batch', 'Genotype', 'Sex and 'LifeStage' if exists in the data
...	Optional parameters that can be passed to 'Hmisc::plot.describe()'

Details

The plot function produces two sets of plots for:

- categorical data: scatter plot of proportions
- continuous data: histogram

Value

List of two plot objects, Continuous and Categorical

See Also

OpenStatsList

Examples

```
example(OpenStatsList)
```

`plot.OpenStatsMM` *plot for an 'OpenStatsMM' object*

Description

This function visualises an 'OpenStatsMM' object

Usage

```
 ## S3 method for class 'OpenStatsMM'
plot(x, main = "Final Model", ask = FALSE, mfrow = c(2, 2), ...)
```

Arguments

`x`	an instance of 'OpenStatsMM' result from OpenStatsAnalysis(method = 'MM') function
`main`	a string to be pasted to the title of the plots
`ask`	see 'ask' in 'par()' function. Default FALSE
`mfrow`	the screen partition. see 'mfrow' argument in the 'par' function. Default c(2,2) then all plots display in one screen.
`...`	other parameters that can be passed to the 'plot' function

Details

The plot function creates some visualisations for the linear mixed model framework:

- Residual versus fitted values
- Residual density plot and the normality test p-value
- Residual Q-Q plot
- The density plot of the response variable and the normality test p-value

Value

Not applicable

See Also

OpenStatsAnalysis, plot.OpenStatsFE, plot.OpenStatsRR

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Continuous data - Creating OpenStatsList object
#################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)

#################
#  Optimised Linear Mixed model (MM) framework
#################
MM1_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "MM",
  MM_fixed = data_point ~ Genotype + Weight
)
print(MM1_result, col = 2, main = "Optimised model")
```

`plot.OpenStatsRR` *plot for an 'OpenStatsRR' object*

Description

This function visualises an 'OpenStatsRR' object

Usage

```
 ## S3 method for class 'OpenStatsRR'
plot(x, main = "Mosaic plot", ask = FALSE, mfrow = c(2, 2), ...)
```

Arguments

`x`	an instance of 'OpenStatsRR' result from OpenStatsAnalysis(method = 'RR') function
`main`	a string to be pasted to the title of the plots
`ask`	see 'ask' in 'par()' function. Default FALSE
`mfrow`	the screen partition. see 'mfrow' argument in the 'par' function. Default c(2,2) then all plots display in one screen.
`...`	other parameters that can be passed to the 'plot' function

Details

The plot function creates some visualisations for the reference range plus framework

- Mosaic plot of the discretised response versus Genotype/Sex/LifeStage (if they exist in the data
- Mosaic plot of the Sex versus Genotype (if they exist in the data)

Value

Not applicable

See Also

`OpenStatsAnalysis`, `plot.OpenStatsFE`, `plot.OpenStatsMM`

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Continuous data - Creating OpenStatsList object
#################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
```

```
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)

#################
# Reference range framework
#################
RR_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "RR",
  RR_formula = data_point ~ Genotype + Sex
)
plot(RR_result, col = 3:4)
```

`print.OpenStatsComplementarySplit` *Summary for an OpenStatsComplementarySplit object*

Description

This function displays a summary table for an 'OpenStatsComplementarySplit' object

Usage

```
 ## S3 method for class 'OpenStatsComplementarySplit'
print(x, format = "rst", ...)
```

Arguments

`x`	an instance of 'OpenStatsComplementarySplit' result
`format`	See format argument from the knitr::kable function
`...`	Other parameters that can be passed to knitr::kable function

Value

The output consists of the following statistics for levels of partitioning variables (see 'variables' in the 'OpenStatsComplementarySplit' function manual):

- Applied model
- Checked/optimised model
- Treatment group
- Control group

- If possible, whether sexual dimorphism is detected from the analysis
- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

OpenStatsComplementarySplit, OpenStatsAnalysis, plot.OpenStatsComplementarySplit, print.OpenStatsComplementarySplit

Examples

```
example(OpenStatsComplementarySplit)
```

---

print.OpenStatsFE *Print summary table for an OpenStatsFE object*

Description

This function prints summary table for an OpenStatsFE object

Usage

```
  ## S3 method for class 'OpenStatsFE'
print(x, format = "rst", ...)
```

Arguments

x	an instance of OpenStatsFE result from OpenStatsAnalysis(method = 'FE') function
format	See format argument from the knitr::kable function
...	Other parameters that can be passed to knitr::kable function

Value

The output consists of the following statistics:

- Applied model
- Checked/optimised model
- Treatment group
- Control group
- If possible, whether sexual dimorphism is detected from the analysis

- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

OpenStatsAnalysis, print.OpenStatsMM, print.OpenStatsRR

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Categorical data - Creating OpenStatsList object
#################
fileCat <- system.file("extdata", "test_categorical.csv", package = "OpenStats")
test_Cat <- OpenStatsList(
  dataset = read.csv(fileCat, na.strings = "-"),
  testGenotype = "Aff3/Aff3",
  refGenotype = "+/+",
  dataset.colname.genotype = "Genotype",
  dataset.colname.batch = "Assay.Date",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "Weight",
  dataset.colname.sex = "Sex"
)
#################
# Fisher's exact test framework
#################
FE_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cat,
  method = "FE",
  FE_formula = Thoracic.Processes ~ Genotype + Sex
)
print(FE_result)
```

---

print.OpenStatsList *Print summary table for an OpenStatsList object*

Description

This function prints a summary table for an OpenStatsList object

Usage

```
 ## S3 method for class 'OpenStatsList'
print(x, vars = NULL, ...)
```

Arguments

x	OpenStatsList object
vars	Variable(s) of interest
...	Optional parameters that can be passed to 'summarytools::dfSummary ()'

Value

Tabe of summary statistics

See Also

OpenStatsList, summary.OpenStatsList, OpenStatsAnalysis

Examples

```
example(OpenStatsList)
```

---

print.OpenStatsMM *Summary for an OpenStatsMM object*

Description

This function prints summary table for an OpenStatsMM object

Usage

```
 ## S3 method for class 'OpenStatsMM'
print(x, format = "rst", ...)
```

Arguments

x	an instance of OpenStatsMM result from OpenStatsAnalysis(method = 'MM') function
format	See format argument from the knitr::kable function
...	Other parameters that can be passed to knitr::kable function

Value

The output consists of the following statistics:

- Applied model
- Checked/optimised model
- Treatment group
- Control group
- If possible, whether sexual dimorphism is detected from the analysis
- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

OpenStatsAnalysis, print.OpenStatsFE, print.OpenStatsRR

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Continuous data - Creating OpenStatsList object
#################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)

#################
#  Optimised Linear Mixed model (MM) framework
#################
MM1_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "MM",
  MM_fixed = data_point ~ Genotype + Weight
)
print(MM1_result)
```

`print.OpenStatsRR` *Summary for an OpenStatsRR object*

Description

This function prints summary table for an OpenStatsRR object

Usage

```
 ## S3 method for class 'OpenStatsRR'
print(x, format = "rst", ...)
```

Arguments

`x`	an instance of OpenStatsRR result from OpenStatsAnalysis(method = 'RR') function
`format`	See format argument from the knitr::kable function
`...`	Other parameters that can be passed to knitr::kable function

Value

The output consists of a pair of values separated by comma, e.g. 1,1, for low and high classes respectively. The following statistics are reported in the summary:

- Applied model
- Checked/optimised model
- Treatment group
- Control group
- If possible, whether sexual dimorphism is detected from the analysis
- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

`OpenStatsAnalysis`,, `print.OpenStatsFE`, `print.OpenStatsMM`

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Continuous data - Creating OpenStatsList object
#################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)

#################
# Reference range framework
#################
RR_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "RR",
  RR_formula = data_point ~ Genotype + Sex
)
print(RR_result)
```

`summary.OpenStatsComplementarySplit` *Summary for an OpenStatsComplementarySplit object*

Description

This function provides summary for an 'OpenStatsComplementarySplit' object

Usage

```
 ## S3 method for class 'OpenStatsComplementarySplit'
summary(object, format = "rst", ...)
```

Arguments

`object`	an instance of 'OpenStatsComplementarySplit' result
`format`	See format argument from the knitr::kable function
`...`	Other parameters that can be passed to knitr::kable function

Value

The output consists of the following statistics for levels of partitioning variables (see 'variables' in the 'OpenStatsComplementarySplit' function manual):

- Applied model
- Checked/optimised model
- Treatment group
- Control group
- If possible, whether sexual dimorphism is detected from the analysis
- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

`OpenStatsComplementarySplit`, `plot.OpenStatsComplementarySplit`, `print.OpenStatsComplementarySplit`, `OpenStatsAnalysis`

Examples

```
example(OpenStatsComplementarySplit)
```

---

summary.OpenStatsFE *Summary for an OpenStatsFE object*

Description

This function provides summary for an OpenStatsFE object

Usage

```
 ## S3 method for class 'OpenStatsFE'
summary(object, format = "rst", ...)
```

Arguments

`object`	an instance of OpenStatsFE result from OpenStatsAnalysis(method = 'FE') function
`format`	See format argument from the knitr::kable function
`...`	Other parameters that can be passed to knitr::kable function

Value

The output consists of the following statistics:

- Applied model
- Checked/optimised model
- Treatment group
- Control group
- If possible, whether sexual dimorphism is detected from the analysis
- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

OpenStatsAnalysis, summary.OpenStatsMM, summary.OpenStatsRR

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Categorical data - Creating OpenStatsList object
#################
fileCat <- system.file("extdata", "test_categorical.csv", package = "OpenStats")
test_Cat <- OpenStatsList(
  dataset = read.csv(fileCat, na.strings = "-"),
  testGenotype = "Aff3/Aff3",
  refGenotype = "+/+",
  dataset.colname.genotype = "Genotype",
  dataset.colname.batch = "Assay.Date",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "Weight",
  dataset.colname.sex = "Sex"
)
#################
# Fisher's exact test framework
#################
FE_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cat,
  method = "FE",
  FE_formula = Thoracic.Processes ~ Genotype + Sex
)
summary(FE_result)
```

summary.OpenStatsList *Summary for an OpenStatsList object*

Description

This function provides a detailed summary of an OpenStatsList object

Usage

```
 ## S3 method for class 'OpenStatsList'
summary(object, vars = NULL, ...)
```

Arguments

object	OpenStatsList object
vars	Variable(s) of interest
...	Optional parameters that can be passed to 'summarytools::dfSummary ()'

Value

Tabe of summary statistics

See Also

OpenStatsList, summary.OpenStatsList

Examples

```
example(OpenStatsList)
```

summary.OpenStatsMM *Summary for an OpenStatsMM object*

Description

This function provides summary for an OpenStatsMM object

Usage

```
 ## S3 method for class 'OpenStatsMM'
summary(object, format = "rst", ...)
```

Arguments

object	an instance of OpenStatsMM result from OpenStatsAnalysis(method = 'MM') function
format	See format argument from the knitr::kable function
...	Other parameters that can be passed to knitr::kable function

Value

The output consists of the following statistics:

- Applied model
- Checked/optimised model
- Treatment group
- Control group
- If possible, whether sexual dimorphism is detected from the analysis
- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

OpenStatsAnalysis, summary.OpenStatsFE, summary.OpenStatsRR

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Continuous data - Creating OpenStatsList object
#################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)

#################
```

```
#  Optimised Linear Mixed model (MM) framework
#################
MM1_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "MM",
  MM_fixed = data_point ~ Genotype + Weight
)
summary(MM1_result)
```

---

summary.OpenStatsRR *Summary for an OpenStatsRR object*

Description

This function provides summary for an OpenStatsRR object

Usage

```
 ## S3 method for class 'OpenStatsRR'
summary(object, format = "rst", ...)
```

Arguments

object	an instance of OpenStatsRR result from OpenStatsAnalysis(method = 'RR') function
format	See format argument from the knitr::kable function
...	Other parameters that can be passed to knitr::kable function

Value

The output consists of a pair of values separated by comma, e.g. 1,1, for low and high classes respectively. The following statistics are reported in the summary:

- Applied model
- Checked/optimised model
- Treatment group
- Control group
- If possible, whether sexual dimorphism is detected from the analysis
- Genotype effect p-value
- Genotype effect p-value for females
- Genotype effect p-value for males
- If LifeStage existed in the data, LifeStage p-value
- Genotype effect for early adults
- Genotype effect for late adults
- If Sex existed in the data, Sex p-value
- If bodyweight existed in the data, bodyweight p-value

See Also

OpenStatsAnalysis, summary.OpenStatsFE, summary.OpenStatsMM

Examples

```
######################################################################
# Data preparation
######################################################################
#################
# Continuous data - Creating OpenStatsList object
#################
fileCon <- system.file("extdata", "test_continuous.csv", package = "OpenStats")
test_Cont <- OpenStatsList(
  dataset = read.csv(fileCon),
  testGenotype = "experimental",
  refGenotype = "control",
  dataset.colname.genotype = "biological_sample_group",
  dataset.colname.batch = "date_of_experiment",
  dataset.colname.lifestage = NULL,
  dataset.colname.weight = "weight",
  dataset.colname.sex = "sex"
)

#################
# Reference range framework
#################
RR_result <- OpenStatsAnalysis(
  OpenStatsList = test_Cont,
  method = "RR",
  RR_formula = data_point ~ Genotype + Sex
)
summary(RR_result)
```

Index

∗ **OpenStatsAnalysis**
    OpenStatsList, 10
∗ **OpenStatsListBuilder**
    OpenStatsListBuilder, 12
∗ **OpenStatsList**
    OpenStatsList, 10
∗ **~OpenStats**
    OpenStatsAnalysis, 2
    OpenStatsComplementarySplit, 8
∗ **~SplitEffect**
    OpenStatsComplementarySplit, 8

OpenStatsAnalysis, 2, *9*, *12, 13*, *15*, *17*, *19, 20*, *22–26*, *28, 29*, *31*, *33*
OpenStatsComplementarySplit, *6*, 8, *16*, *22*, *28*
OpenStatsList, *6*, 10, *13*, *18*, *24*, *30*
OpenStatsListBuilder, 12
OpenStatsReport, 14

plot.OpenStatsComplementarySplit, *9*, 15, *22*, *28*
plot.OpenStatsFE, *6*, 16, *19, 20*
plot.OpenStatsList, *12*, 17
plot.OpenStatsMM, *6*, *17*, 18, *20*
plot.OpenStatsRR, *6*, *17*, *19*, 20
print.OpenStatsComplementarySplit, *16*, 21, *22*, *28*
print.OpenStatsFE, *6*, 22, *25, 26*
print.OpenStatsList, 23
print.OpenStatsMM, *6*, *23*, 24, *26*
print.OpenStatsRR, *6*, *23*, *25*, 26

summary.OpenStatsComplementarySplit, *9*, *16*, 27
summary.OpenStatsFE, *6*, 28, *31*, *33*
summary.OpenStatsList, *12*, *24*, 30, *30*
summary.OpenStatsMM, *6*, *29*, 30, *33*
summary.OpenStatsRR, *6*, *29*, *31*, 32