Package 'BERGMets'

July 15, 2019

Type Package

Title Metabolomics data and analysis in the BERG

Version 0.1.0

Author Brian Carter

Maintainer Brian Carter <brian.carter@cancer.org>

Description
This package provides analytic datasets for all of the BERG metabolomics data as well as several useful functions for doing some initial analyses.

License None

Depends dplyr, FitAR, parallel, ppcor, magrittr, tibble, SASxport, foreign, haven, ggcorrplot, pROC, plotROC

Imports lme4

Remotes marklhc/bootmlm

Encoding UTF-8

LazyData true

R topics documented:

`breast_metabolomics` *Breast cancer metabolomics file*

Description

These data are a subset of our original breast cancer metabolomics NCIA-03-16-16ML+. It includes the cleaned analytic metabolite file, a data set of metadata describing each metabolite, and an ID conversion file.

Usage

```
getMetabolites("breast_metabolomics")
```

Format

A list with 3 data frames

`metabolites` A data frame of 1,547 observations and 1,385 variables. These metabolites were scaled to the daily median by Metabolon and the missings were imputed to the minimum. QC duplicates were averaged to provide only a single sample. No exclusions have been made: there are 1,547 unique samples in the data, and 1,384 metabolites (plus an ID variable)

`ID` A data frame of 1,564 observations and 5 variables. This is an ID conversion file for the breast cancer metabolomics and includes duplicate IDs indicating the QC samples. The original data from Metabolon had several IDs, this file will help you merge if you need to go back to the original. Also includes the case-control and Matching variables.

`biochem` A data frame of 1,385 observations and 18 variables. Each metabolite has a set of metadata from Metabolon, including a biochemical name, pathway information, and various platform information. This dataset also includes derived metadata including the number/proportion of samples missing a given metabolite and ICC_TECH / CV quality scores.

Details

The breast_metabolomics file included in the package should include everything an analyst requires to complete a metabolomics project with these data. A link to the original Rdata file is listed in the references. If you need more information on the breast cancer metabolomics data, please see the documentation file at that file path.

References

S:\CPS\BIOSPECIMENS\Metabolomics\Breast cancer - NCIA-03-16ML\

Examples

```
breast_metabolomics <- getMetabolites("breast_metabolomics")
metabolites <- breast_metabolomics$metabolites
ID <- breast_metabolomics$ID
biochem <- breast_metabolomics$biochem
```

`corMatrix` *Calculates a correlation matrix adjusting for covariates*

Description

The corMatrix() will calculate correlation matrices between source and target variables. If users include a vector of covariates, corMatrix() will calculate partial correlations adjusted for these covariates. See details.

Usage

```
corMatrix(dat, target, source, covariates = NULL, method="pearson")
```

Arguments

`dat`	A data frame containing your metabolites and covariates
`source`	A vector of variable names for your metabolites (see details)
`target`	A vector of variable names for your metabolites (see details)
`covariates`	Optional vector of names for your adjustment variables (see details). All these variables must be numeric.
`method`	a character string indicating which correlation coefficient (or covariance) is to be computed: "pearson" (default), "kendall", or "spearman"

Details

The function will produce simple pairwise correlations or adjusted partial correlations.

Full correlation matrices are computationally intensive because they must be calculated iteratively. For example, computing pairwise correlations for 100 metabolites equals 100^2 calculations; 500 metabolites equals 500^2 calculations.

The corMatrix() function utilizes the parallel package to split these calculations across logical CPU cores and run them in parallel. This cuts down the time it takes to calculate these correlations considerably. For example, including 800 metabolites on a dataset of 1500 samples will take about 8 minutes of processing time.

Value

Correlation matrix: Rows=target; Columns=source. Diagonal==1.0

Author(s)

Brian Carter and Becky Hodge

See Also

ppcor::pcor.test cor

Examples

```
# covariate data

data(survey)

# Metabolite data

breast_metabolomics <- getMetabolites("breast_metabolomics")

# vector of metabolites

comp.id <- names(breast_metabolomics$metabolites)[-1]

# Merge metabolites with survey data

df <- dplyr::left_join(survey, breast_metabolomics$metabolites, "ID")

# Run a simple correlation matrix - no covariates

mat <- corMatrix(dat=df, source=comp.id, target=comp.id, method="pearson")

# Run the full partial correlation matrix

mat <- corMatrix(dat=df,
                     source=comp.id,
                     target=comp.id,
                     covariates="AGE_INT")
```

---

getAUC *Calculates AUC for a logistic regression.*

Description

getAUC() takes a series of outcomes, metabolites, and covariates and calculates AUC for each model. You can input as many outcomes and metabolites as you want.

Usage

```
getAUC(dat, outcome, compid, covariates=NULL, normalize=T)
```

Arguments

dat	Data frame containing your metabolites, outcome, and covariates
outcome	Numeric vector coded [0,1] for your outcome.
compid	Vector of COMP_IDs
covariates	Vector of covariate variable names (optional)
normalize	Normalize the data prior to modeling? (default is TRUE)

Value

If a single outcome is provided, getAUC() will return a data frame with the following variables

COMP_ID	Each COMP_ID listed in compid argument
OUTCOME	Outcome
AUC	

AUC calculation

If multiple outcomes are requested, getAUC() will return a list. Each data frame of the list corresponds to each of the outcomes.

Author(s)

Brian Carter and Becky Hodge

See Also

roc, normalizeMets,

Examples

```
# load metabolite data

metabolites <- getMetabolites("breast_metabolomics")
df <- left_join(survey,metabolites$metabolites,"ID")

comp.id <- metabolites$biochem$COMP_ID[1:10]

# code BMI categories as [0,1] - obese vs normal
df$bmi_binary <- ifelse(df$BMI < 25,0, ifelse(df$BMI >= 30,1,NA))

myAUC <- getAUC(df,
                outcome="bmi_binary",
                compid = comp.id,
                covariates="AGE_INT",
                normalize=T)
```

---

getID *Pulls IDs from the metabolite files and optionally write to a SAS dataset for later use.*

Description

Each metabolite has an ID file. The getID() function will pull these IDs. By default, getID() will create a data frame in your global environment. You can also request export to SAS via the foreign:: package. See details.

Usage

```
getID(dat,filepath=NULL)
```

Arguments

dat	Character vector of the dataset you are requesting (see details below). Options are NOT case sensitive.
filepath	Optional file path to save the ID file. Remember that R requires forward slashes "/" in file paths. Example: "S:/user/bcarter/" NOT "S:\user\bcarter". If you leave filepath=NULL, getID() will simply create an ID file within your global environment.

Details

Users can pull IDs from any of 7 metabolomics projects using the following character strings. For more information, you can find documentation by searching the BERGMets documentation

SAS foreign files: The foreign package does two things. First, it creates a .txt file of the data you are exporting from R. Second, it writes a SAS program that will import this .txt file into SAS and format everything correctly. Selecting this option will write a data file and a .SAS program to your filepath. Simply run the SAS program and your data will be imported.

Value

getID() will only return a data frame named "Metabolomics_IDs" if you leave filepath=NULL. Otherwise, it will save an .xpt or .txt/.sas files to your filepath. You can move on to SAS to finish your analysis.

Author(s)

Brian Carter

See Also

getMetabolites

Examples

```
# Pull the IDs for use in R

# Will return a data frame named "Metabolomics_IDs"

getID("breast_metabolomics")


# Export to SAS using the foreign package

# Creates 2 files:

  # "Import ID_Breast_metabolomics.SAS"

  # "ID_breast_metabolomics"

# Simply run the "Import ID_Breast_metabolomics.SAS" and continue in SAS

  getID(dat="breast_metabolomics",
        filepath="S:/user/bcarter/metabolites")
```

`getMetabolites` *Loads and/or subsets metabolite data.*

Description

Each metabolomics dataset has multiple versions based on how the lab chose to scale them. This function will pull the correct data.

Usage

```
getMetabolites(dat, IDlist=NULL, compid=NULL)
```

Arguments

`dat`	Character vector of the dataset you are requesting (see details below). Options are NOT case sensitive.
`IDlist`	Optional: Character vector of a subset of IDs. If you have already prepared your covariate dataset, you can include a vector of IDs for subsetting the metabolite data. If you leave it NULL, all data will be returned.
`compid`	Optional: Character vector of a subset of COMP_IDs. If you know what metabolites you are looking for, you can enter a vector of these COMP_IDs and only pull these data.

Details

Users can pull data from any of 7 metabolomics projects using the following character strings. For more information, you can find documentation by searching the BERGMets documentation:

1. "breast_metabolomics"
2. "Parkinsons_C8"
3. "Parkinsons_HILICNEG"
4. "Parkinsons_HILICPOS"
5. "prostate_metabolomics"
6. "Validation_Blood"
7. "Validation_Urine"

Value

getMetabolites will return a list of 3 objects:

$metabolites - Data frame of the metabolite data organized by ID

$ID - and ID conversion file, useful if you need to pull lifelink data or refer to the original data files

$biochem - Biochemical metadata: includes biochemical names, pathway information, platform/batch information, QC metrics (CV, ICC, N-missing, etc)

Author(s)

Brian Carter

Examples

```
# Access documentation for the breast_metabolomics file:

?breast_metabolomics

# Pull all data

breast_metabolites <- getMetabolites("breast_metabolomics")

names(breast_metabolites)

# Pull just some IDs

ID <- c("42101011001022",
        "55010005007081",
        "43017011009062",
        "56017043004052",
        "06026241020062")

breast_metabolites <- getMetabolites("breast_metabolomics", IDlist=ID)

nrow(breast_metabolites$metabolites)

# Pull just some metabolites

mets <- c("X34404","X21132",
          "X57463","X37183",
          "X39346")

breast_metabolites <- getMetabolites("breast_metabolomics", compid=mets)

ncol(breast$metabolites$metabolites)

# Pull just some metabolites for just some IDs

breast_metabolites <- getMetabolites("breast_metabolomics",
                                     IDlist=ID, =compid=mets)

dim(breast_metabolites$metabolites)
```

`icc9` *R Version of the ICC9 SAS Macro*

Description

The icc9() function adapts the ICC9 macro published by Ellen Hertzmark and Donna Spiegelman (2010). It computes intraclass correlation coefficients (ICC) and coefficients of variation (CV) and their confidence intervals. These metrics can be calculated with and without adjustment for fixed effects.

Usage

```
icc9(y,subject,fixed=NULL,dat,method="ML")
```

Arguments

y	Name of the continuous response vector (REQUIRED)
subject	Name of the subject vector (REQUIRED)
fixed	Vector of fixed effects one or more variable names (OPTIONAL)
dat	Data frame containing variables used in analysis (REQUIRED)
method	Users can indicate "ML" for maximum likelihood method or "REML" for restricted maximum likelihood. By default, the lmer() function calculates estimates based on restricted maximum likelihood (REML). The original SAS ICC9 macro defaults to a maximum likelihood (ML) method, and so that is the default for this R function.

Details

Users familiar with the icc9 SAS macro will notice a few differences with the R function. Notably, the macro options for subsetting the data or including BY= and WHERE= options have been removed. Users should prepare their data prior to running the function.

This function has been tested on a limmited set of data, but has so far replicated the SAS ICC and CV estimates with only an occasional rounding difference. If the user identifies major differences in their dataset, please contact the author of this function with details.

Value

Data frame containing ICC and CV estimates

Response_Variable	Reproduces the response variable for the model (the "y" argument in the function)
N_subjects	Number of unique "subject" values in the input dataset
N_Observations	Number of observations in the input dataset
Average_Measurements	Average number of observations per unique subject
ICC	Intraclass correlation coefficient with confidence intervals
CV	Coefficient of variation with confidence intervals

Author(s)

Brian Carter (brian.carter@cancer.org)

References

Users can refer to the documentation for the original SAS icc9 macro at the following web address: https://cdn1.sph.harvard.edu/wp-content/uploads/sites/271/2012/09/icc9.pdf

Examples

```
# Example 1 - basic run
icc9(y="X48761", subject="subjid", dat=qc)

# Example 2 - using BATCH1 as a fixed effect
icc9(y="X48761", subject="subjid", fixed=BATCH1, dat=qc)
```

```
# Example 3 - Using multiple fixed effects
icc9(y="X48761", subject="subjid", fixed=c(BATCH1,BATCH2), dat=qc)

# Exmample 4 - Using the function for multiple response variables
library(dplyr)
lapply(c("X48761","X19130","X53174"), function(response) {
     icc9(y=response,subject="subjid",dat=qc)
     }) %>% do.call("rbind",.)
```

---

makeSAS *Creates a SAS datafile from metabolomics data*

Description

makeSAS() will import metabolomics data and convert it to a SAS file. You can subset the metabolomics files based on IDs or COMP_IDs. See details how to finish the import in SAS

Usage

```
makeSAS(dat,IDlist=NULL,compid=NULL,filepath)
```

Arguments

dat	Character vector of the dataset you are requesting (see details below). Options are NOT case sensitive.
IDlist	Optional: Character vector of a subset of IDs. If you have already prepared your covariate dataset, you can include a vector of IDs for subsetting the metabolite data. If you leave it NULL, all data will be returned.
compid	Optional: Character vector of a subset of COMP_IDs. If you know what metabolites you are looking for, you can enter a vector of these COMP_IDs and only pull these data.
filepath	File path to save the files. Remember that R requires forward slashes "/" in file paths. Example: "S:/user/bcarter/" NOT "S:\user\bcarter"

Details

Users can pull data from any of 7 metabolomics projects using the following character strings. For more information, you can find documentation by searching the BERGMets documentation:

makeSAS() will create 4 files at the output location. Two .txt files will be fore the actual metabolites files and a metadata file listing all the relevant data describing each metabolite including biochemical name, pathway information, platform, and some QC metrics.

The other two files are .SAS programs. In order to get the metabolite data into SAS, simply run these SAS programs. This will create a dataset in the WORK library that you can work with in whatever way you want.

Value

Four files will be created in your output folder. Naming conventions based on the specific metabolomics project you are working on (see examples)

1. breast_metabolomics_date.txt 2. breast_metabolomics_metadata_date.txt 3. import Breast_metabolomics_date.SAS 4. import breast_metabolomics_metadata_date.SAS

Author(s)

Brian Carter

See Also

getMetabolites

Examples

```
makeSAS(dat="breast_metabolomics",
        filepath="s:/users/bcarter/")
```

---

metsAssoc *Basic metabolomics association analysis.*

Description

This function will run a basic association analysis adjusted for covariates for a given list of metabolites. Depending on the outcome, the function will choose a linear or logistic regression model. It will calculate the associations between outcome and each metabolite, merge with the metadata files, and return an organized table.

Usage

```
metsAssoc(dat, biochem, outcome, metabolites, covariates=NULL, normalize=T)
```

Arguments

dat	data frame containing your metabolites and covariates
biochem	The biochemical metadata file. Typically this is in the $biochem element of the metabolites list. You can subset this data frame any way you wish, some variables are probably unneeded. Recommend that you subset to only include COMP_ID, BIOCHEMICAL, SUPER_PATHWAY, SUB_PATHWAY.
outcome	Outcome for the analysis. Logistic regression must be coded [0,1]; linear regression must be continuous numeric. Function will identify which is which and will provide an error message if you code it wrong.
compid	Vector of COMP_IDs used for the models
covariates	Optional. Vector of covariate names from the dat data frame.
normalize	Logical. Set to TRUE if you haven't already mean-centered and glog transformed your metabolite data. Default is TRUE.

Details

This function will run simple association models (either linear or logistic regressions). I recommend that you use the metsModels() function which is a wrapper for all sorts of metabolomics models. It will either run metsAssoc() or metsInt() depending on the inputs.

Value

Data frame organizing the associations and p-values for all metabolites.

Author(s)

Brian Carter

See Also

normalizeMets, metsInt, metsModels,

Examples

```
breast_metabolomics <- getMetabolites("breast_metabolomics")
data(survey)
df <- left_join(survey,breast_metabolomics$metabolites,"ID")
covars<- c("AGE_INT","LASTATE")
comp.id <- names(breast_metabolomics$metabolites)[-1]
biochem <- breast_metabolomics$biochem[,c("COMP_ID","BIOCHEMICAL")]

# Linear regression models
out <- metsAssoc(dat=df,
                        biochem=biochem,
                        outcome="BMI",
                        compid=comp.id[1:10],
                        covariates=covars,
                        normalize=T)  # normalizing data

# Logistic regression models
out <- metsAssoc(dat=df,
                        biochem=biochem,
                        outcome="BMIBIN",
                        compid=comp.id[1:10],
                        covariates=covars,
                        normalize=T)  # normalizing data
```

---

metsBoxplot *Creates layered boxplots of metabolite values across a class variable.*

Description

Boxplots are often useful for illustrating the distribution of metabolite values across a categorical outcome variable. metsBoxplot() will create these boxplots using ggplot2. The plots are somewhat customizable, and can be further customizable by including a custom theme object.

Usage

```
metsBoxplot(dat, classvar, metabolite, ylog = F, boxcolors = NULL, ref_line = NULL, title, subtitle =
```

Arguments

dat	Data frame used for the analysis
classvar	Factor variable used for the x-axis
metabolite	Character vector (length==1) of the COMP_ID used for the y-axis. May be raw or log transformed
ylog	Logical, Plot y-axis on a log10 scale.
boxcolors	Vector of colors equal to the length of levels(classvar). If you leave this NULL, a black and white image will be returned, but you can color each of the boxes if needed.
ref_line	Numeric vector indicating a horizontal reference line. You may want a line to indicate overall minimum or median, or any other reference you might like to plot.
title	Character vector of a plot title, optional.
subtitle	Character vector of a subtitle, optional
xlab	X-axis label. Defaults to the classvar.
ylab	Y-axis label. Defaults to metabolite COMP_ID
xaxis_size	Font size for x-axis
yaxis_size	Font size for y-axis
titlesize	Font size for title
plot_theme	You can create your own ggplot2 theme object and include it here. If you leave it as NULL, the plot will return with Brian's preferred aesthetic tastes, for better or worse.

Value

ggplot2 object.

Author(s)

Brian Carter

Examples

```
breast <- getMetabolites("breast_metabolomics")

df <- left_join(survey,
                breast$metabolites,
                "ID")

df$BMICAT <- with(df,
                  ifelse(BMI < 18.5, 1, ifelse(
                       BMI < 25, 2, ifelse(
                            BMI < 30, 3, ifelse(
                                 BMI < 35, 4, 5)
                            ))))
     factor(.,1:5, c("<18.5", "18.5-24.9",
                     "25-29.9", "30-34.9",
                     "35+"))
```

```
myplot <- metsBoxplot(dat=df,
            classvar="BMICAT",
            metabolite="X42459",
            ylog = T,
            boxcolors = c("royalblue","darkred","yellow",
                          "orange","forestgreen"),
            ref_line = min(df$X42459),
            title = "Title goes here",
            subtitle = "Subtitle goes here",
            xlab = "BMI Categories",
            ylab = "Sphingomyelin",
            xaxis_size = 12,
            yaxis_size = 12,
            titlesize = 14,
            plot_theme = NULL)


png("Boxplot.png", height=6, width=6, units="in", res=400)
myplot
dev.off()
```

`metsHeatmap` *Creates a correlation heatmap using ggplot2 syntax*

Description

This function will create a heatmap using ggcorrplot(). The function will produce the correlation (or partial correlation) matrix using corMatrix() and then construct the figure. Users are given a limited number of options for adjusting the figures and can supply their own theme().

Usage

```
metsHeatmap(dat, compid, covariates = NULL, biochem = NULL, method = "pearson", hclust = T, title, su
```

Arguments

`dat`	data frame including metabolites and (optionally) covariates
`compid`	Character vector of COMP_IDs included in the heatmap
`covariates`	Optional vector of covariates. If supplied, heatmap will be based on a partial correlation matrix
`biochem`	Data frame of biochemical metadata. At a minimum it must have two variables: COMP_ID and BIOCHEMICAL. The BIOCHEMICAL will be used to label the y-axis. If left NULL, COMP_IDs will be used to label the x- and y-axes.
`method`	Correlation methods: "pearson" (default), "spearman", or "kendall"
`hclust`	Logical: use heirarchical clustering for arranging the heatmap? Defaults to TRUE
`title`	Title for your plot, required.
`subtitle`	Optional subtitle. If left NULL, some summary information about the correlations will be added.

plot_colors	Adjust the color scheme by provided exactly THREE colors. Correlations range from -1.0 to 1.0. The first color wil be -1.0, middle color will be 0, and third color will be 1.0. The function will create gradations between these colors.
xaxis_size	Text size for the x-axis
yaxis_size	Text size for the y-axis
title_size	Text size for the title
subtitle_size	Text size for the subtitle
plot_theme	Optional ggplot2 theme. metsHeatmap() will create a theme based on some of your options. You can create a custom theme using theme() and include the name of this theme object in this argument to apply it.

Details

Building heatmaps from ggcorrplot() is a tedious process, hopefully this makes it a bit easier. Users should be able to edit most of the plot using theme() elements, or they can accept the defaults. The function returns a ggplot2 object, so you can edit the plot after running the function.

Value

ggplot object

Author(s)

Brian Carter

See Also

ggcorrplot, corMatrix, theme

Examples

```
breast_metabolomics <- getMetabolites("breast_metabolomics")
biochem <- b$biochem
metdata <- dplyr::left_join(survey,breast_metabolomics$metabolites,"ID")
comp.id <- sample(biochem$COMP_ID,50,replace=T)


png(file.path("test mets heatmap.png"),height=10,width=10,units="in",res=400)
metsHeatmap(metdata,
            compid=comp.id,
            covariates=NULL,
            biochem=NULL,
            method="pearson",
            hclust=T,
            title="Title to test my new metsHeatmap() function",
            subtitle="here's a subtitle describing something or another",
            plot_colors=NULL,
            xaxis_size=7,
            yaxis_size=7,
            title_size=12,
            subtitle_size=8,
            plot_theme=NULL)
dev.off()
```

metsInt *Stratified and interaction metabolite association models*

Description

This function will run stratified or interaction models for metabolites with a continuous or binary outcome. The choice of stratified vs simple interaction depends on the type of interaction variable. See details.

Usage

```
metsInt(dat,biochem,outcome,metabolites,intvar,
                covariates=NULL,normalize=T)
```

Arguments

dat	data frame containing your metabolites and covariates
biochem	The biochemical metadata file. Typically this is in the $biochem element of the metabolites list. You can subset this data frame any way you wish, some variables are probably unneeded. Recommend that you subset to only include COMP_ID, BIOCHEMICAL, SUPER_PATHWAY, SUB_PATHWAY.
outcome	Outcome for the analysis. Logistic regression must be coded [0,1]; linear regression must be continuous numeric. Function will identify which is which and will provide an error message if you code it wrong.
compid	Vector of COMP_IDs used for the models
intvar)	Name of interaction variable. If class(intvar)=="numeric", simple interaction models will be fit coded as metabolite*intvar. If class(intvar)=="factor", the function will return estimates for the metabolites stratified by the intvar.
covariates	Optional. Vector of covariate names from the dat data frame.
normalize	Logical. Set to TRUE if you haven't already mean-centered and glog transformed your metabolite data. Default is TRUE.

Details

In order to use metsInt(), users need to take care to code their variables appropriately. The outcome variable determines the type of model fit: a continuous outcome variable will fit a linear model; a binary outcome will fit a logistic model. If the variables are coded incorrectly, an error/warning will be returned.

Users also must code the intvar correctly. If intvar is a factor variable, the function will return metabolite associations stratified across each level of of intvar. If intvar is coded as a numeric variable, the function will return associations for the interaction term. Both models will return a p-value for the interaction based on a likelihood ratio test comparing the interaction model with a reduced model.

Value

A data frame containing the results. Results are merged with the biochem-metadata file and estimates/pvalues are nicely formatted.

Stratified models will return estimates across each level of the interaction variables.

Interaction models will return only the estimate and p-values associated with the interaction term. Both will return an overall p-value for the interaction.

Author(s)

Brian Carter

See Also

normalizeMets, metsAssoc, metsModels,

Examples

```
# Data
breast_metabolomics <- getMetabolites("breast_metabolomics")
data(survey)
df <- left_join(survey,breast_metabolomics$metabolites,"ID")
covars <- c("AGE_INT","LASTATE")
comp.id <- names(breast_metabolomics$metabolites)[-1]
biochem <- breast_metabolomics$biochem[,c("COMP_ID","BIOCHEMICAL")]

#### stratified models

     # categorize my strata variables (quartiles):
     df$agecat <- gtools::quantcut(df$AGE_INT,4)
     metsInt(dat=df,
                    biochem=biochem,
                    outcome="BMI",
                    compid=comp.id,
                    intvar="agecat",
                    covariates=covars,
                    normalize=T)

#### Interaction models
     metsInt(dat=df,
               biochem=biochem,
               outcome="BMI",
               compid=comp.id,
               intvar="AGE_INT",
               covariates=covars,
               normalize=T)
```

`metsModels` *Wrapper function that will run association, interaction, or stratified models*

Description

metsModels should do everything you need it to do. If you include a intvar= variable, the function will run interaction or stratified models (depending on intvar class, see details). If the intvar= argument is left NULL, simple association models will be returned.

Usage

```
metsModels(dat, biochem, outcome, metabolites,intvar=NULL,
           covariates=NULL, normalize=T)
```

Arguments

dat	data frame containing your metabolites and covariates
biochem	The biochemical metadata file. Typically this is in the $biochem element of the metabolites list. You can subset this data frame any way you wish, some variables are probably unneeded. Recommend that you subset to only include COMP_ID, BIOCHEMICAL, SUPER_PATHWAY, SUB_PATHWAY.
outcome	Outcome for the analysis. Logistic regression must be coded [0,1]; linear regression must be continuous numeric. Function will identify which is which and will provide an error message if you code it wrong.
compid	Vector of COMP_IDs used for the models
intvar)	Optional Name of interaction variable. If left NULL, a simple association analysis will be returned. If class(intvar)=="numeric", simple interaction models will be fit coded as metabolite*intvar. If class(intvar)=="factor", the function will return estimates for the metabolites stratified by the intvar.
covariates	Optional. Vector of covariate names from the dat data frame.
normalize	Logical. Set to TRUE if you haven't already mean-centered and glog transformed your metabolite data. Default is TRUE

Details

the metsModels() function is a wrapper function that runs either metsAssocation() for simple association models, or metsInteraction() for interaction/stratified models.

A simple association model includes either linear or logistic regression (depending on the outcome), adjusted for covariates. If the user includes an intvar= variable, the interaction/stratified models will be returned, depending on the intvar class. See documentation files for metsAssociation() and metsInteraction() for details of either sets of models

Value

A data frame containing the results. Results are merged with the biochem-metadata file and estimates/pvalues are nicely formatted.

Simple association models will return estimates, p-values and metabolite metadata

Stratified models will return estimates across each level of the interaction variables.

Interaction models will return only the estimate and p-values associated with the interaction term.

Both will return an overall p-value for the interaction.

Author(s)

Brian Carter

See Also

metsAssoc, metsInt, normalizeMets

Examples

```
breast_metabolomics <- getMetabolites("breast_metabolomics")
data(survey)
df <- left_join(survey,breast_metabolomics$metabolites,"ID")
covars<- c("AGE_INT","LASTATE")
comp.id <- names(breast_metabolomics$metabolites)[-1]
biochem <- breast_metabolomics$biochem[,c("COMP_ID","BIOCHEMICAL")]

# Association analysis (logistic regression)
out <- metsAssoc(dat=df,
                      biochem=biochem,
                      outcome="BMIBIN",
                      compid=comp.id[1:10],
                      covariates=covars,
                      normalize=T)  # normalizing data



# Stratified analysis (linear regression)
     metsInt(dat=df,
               biochem=biochem,
               outcome="BMI",
               compid=comp.id,
               intvar="AGE_INT",
               covariates=covars,
               normalize=T)

{
  }
```

`metsROC` *Creates ROC curves using ggplot()*

Description

Users provide a binary outcome, metabolites, and covariates. Logistic regression models are fit for each of the metabolites and ROC curves are returned. Users have the choice of plotting single or multiple curves on each figure.

Usage

```
metsROC(dat, outcome, compid, covariates, normalize=T, xlabel = "1 - Specificity",
ylabel = "Sensitivity", title = NULL, colors = NULL, title_size = 10,
subtitle_size = 10, xaxis_size = 9, yaxis_size = 9, plot_theme = NULL)
```

Arguments

`dat`	Data frame containing outcome, metabolites, and covariates data
`outcome`	Binary outcome variable, must be coded [0,1] for a logistic regression.
`compid`	Vector of metabolite COMP_IDs. If you include more than one COMP_ID, all will be plotted on a single figure.
`covariates`	Vector of covariate names.

normalize	Logical. Set to TRUE if you haven't already mean-centered and glog transformed your metabolite data. Default is TRUE.
xlabel	X-axis title - defaults to "1 - Specificity"
ylabel	Y-axis title - defaults to "Sensitivity"
title	Optional title. If you don't include a title, the function will provide one for you.
colors	Optional vector of colors for each curve. length(colors) must equal length(compid)
title_size	Optional text size for title.
subtitle_size	Optional text size for subtitle.
xaxis_size	Optional text size for x-axis.
yaxis_size	Optional text size for y-axis.
plot_theme	Optional ggplot theme object. Users can create their own custom theme. This provides the ability to edit parts of the figure as needed.

Details

the metsROC() function uses ggplot2 to create ROC curves for any number of metabolites. Users should be able to edit the most important aspects of the plot using the arguments in the function; others can be adjusted using theme() elements, or accept the defaults. The function returns a ggplot2 object, so you can edit the plot after running the function.

Value

a ggplot2 object

Author(s)

Brian Carter and Becky Hodge

See Also

ggplot2, plotROC, normalizeMets

Examples

```
breast_metabolomics <- getMetabolites("breast_metabolomics")

biochem <- breast_metabolomics$biochem

metdata <- dplyr::left_join(survey,breast_metabolomics$metabolites,"ID")

comp.id <- sample(biochem$COMP_ID,50,replace=T)

# categorize BMI
metdata$BMIBIN <- metdata$BMIBIN-1



png("test ROC.png",height=6,width=6,units="in",res=400)
metsROC(dat=metdata,
        outcome="BMIBIN",
        compid=comp.id[1:5],
        covariates="AGE_INT",
```

```
        normalize=T,
        xlabel="1 - Specificity",
        ylabel="Sensitivity",
        title=NULL,
        colors=NULL,
        title_size=10,
        #subtitle_size=12,
        #xaxis_size=12,
        #yaxis_size=12,
        plot_theme=NULL)
dev.off()
```

`normalizeMets` *Normalize metabolite data prior to analyses*

Description

This function normalizes metabolite values by creating a log-transformed z-score. This should be done prior to any analysis, although the function can also be done on the fly in the modeling functions by setting the normalize=TRUE argument.

Users have the choice of normalizing an entire dataset or selected metabolites. It is important that the data be subset to your analytic sample prior to running the function. Either merge with your covariate data or subset the IDs.

Usage

```
normalizeMets(dat, metabolites=NULL)
```

Arguments

`dat`	Data frame containing metabolite data.
`metabolites`	Optional. Vector of COMP_IDs of the metabolites in your data.

Details

Users have several options for running this function, but the data must be subset to your analytic sample prior to running. This can be done by merging the metabolites with your survey data or just subsetting the metabolite dataset to some set of IDs.

If you choose to merge the metabolites with your survey data, you MUST include a vector of COMP_IDs in the metabolites argument. See examples below how you might want to do this.

If you choose to normalize the entire dataset, you only need to include the dataset in the 'dat' argument, no vector of metabolites is required. See examples below.

Value

Data frame of metabolites mean centered and log-transformed

Author(s)

Brian Carter

Examples

```
### Example 1 - normalize an entire dataset
     data(breast_metabolomics)
     normalized_data <- normalizeMets(breast_metabolomics$metabolites)

### Example 2 - merge with your survey data first
     data(breast_metabolomics)
     mydata <- left_join(survey_data, breast_metabolomics$metbolites, "ID")

     # create a vector of COMP_IDs
     mets <- c("X101","X102","X103")

     # normalize the data
     mydata <- normalizeMets(mydata,metabolites=mets)
```

Parkinsons_C8 *CPS2 Parkinsons Disease Metabolomics - C8 platform*

Description

These data were collected as part of a collaboration with Harvard on Parkinsons Disease. These data are unlike the other metabolomics data we've collected using Metabolon and are not organized around compound IDs and without the extensive metadata and QC analyses.

Users should check the other Parkinsons data we have collected from HILIC_Pos and HILIC_Neg metabolomics platforms. There are repeats in these data, some metabolites have been collected more than once across the three platforms.

Usage

```
getMetabolites("Parkinsons_C8")
```

Format

A list with 3 data frames

`metabolites` A data frame of 635 observations and 14,624 variables. QC samples have been removed. No scaling has been done to the metabolite data and missing values have NOT been imputed

`ID` A data frame of 635 observations and 3 variables. The original data had several IDs, this file will help you merge if you need to go back to the original.

`biochem` A data frame of 14,623 observations and 12 variables. Each metabolite has a set of metadata from the lab, including a biochemical name, pathway information, and various platform information. This dataset also includes derived metadata including the number/proportion of samples missing a given metabolite and CV quality scores.

Details

The Parkinsons_C8 file included in the package should include everything an analyst requires to complete a metabolomics project with these data. A link to the original Rdata file is listed in the references. If you need more information on these data, please see the documentation file at that file path.

References

S:\CPS\BIOSPECIMENS\Metabolomics\Parkinsons disease\

Examples

```
Parkinsons_C8 <- getMetabolites("Parkinsons_C8")
metabolites <- Parkinsons_C8$metabolites
ID <- Parkinsons_C8$ID
biochem <- Parkinsons_C8$biochem
```

Parkinsons_HILICNEG *CPS2 Parkinsons Disease Metabolomics - HILIC-Pos platform*

Description

These data were collected as part of a collaboration with Harvard on Parkinsons Disease. These data are unlike the other metabolomics data we've collected using Metabolon and are not organized around compound IDs and without the extensive metadata and QC analyses.

Users should check the other Parkinsons data we have collected from C8 and HILIC_Pos metabolomics platforms. There are repeats in these data, some metabolites have been collected more than once across the three platforms.

Usage

```
getMetabolites("Parkinsons_HILICNEG")
```

Format

A list with 3 data frames

`metabolites` A data frame of 636 observations and 88 variables. QC samples have been removed. No scaling has been done to the metabolite data and missing values have NOT been imputed

`ID` A data frame of 636 observations and 3 variables. The original data had several IDs, this file will help you merge if you need to go back to the original.

`biochem` A data frame of 87 observations and 12 variables. Each metabolite has a set of metadata from the lab, including a biochemical name, pathway information, and various platform information. This dataset also includes derived metadata including the number/proportion of samples missing a given metabolite and CV quality scores.

Details

The Parkinsons_HILICNEG file included in the package should include everything an analyst requires to complete a metabolomics project with these data. A link to the original Rdata file is listed in the references. If you need more information on these data, please see the documentation file at that file path.

References

S:\CPS\BIOSPECIMENS\Metabolomics\Parkinsons disease\

Examples

```
Parkinsons_HILICNEG <- getMetabolites("Parkinsons_HILICNEG")
metabolites <- Parkinsons_HILICNEG$metabolites
ID <- Parkinsons_HILICNEG$ID
biochem <- Parkinsons_HILICNEG$biochem
```

Parkinsons_HILICPOS *CPS2 Parkinsons Disease Metabolomics - HILIC-Pos platform*

Description

These data were collected as part of a collaboration with Harvard on Parkinsons Disease. These data are unlike the other metabolomics data we've collected using Metabolon and are not organized around compound IDs and without the extensive metadata and QC analyses.

Users should check the other Parkinsons data we have collected from C8 and HILIC_Neg metabolomics platforms. There are repeats in these data, some metabolites have been collected more than once across the three platforms.

Usage

```
getMetabolites("Parkinsons_HILICPOS")
```

Format

A list with 3 data frames

`metabolites` A data frame of 636 observations and 11,626 variables. QC samples have been removed. No scaling has been done to the metabolite data and missing values have NOT been imputed

`ID` A data frame of 636 observations and 3 variables. The original data had several IDs, this file will help you merge if you need to go back to the original.

`biochem` A data frame of 11,625 observations and 12 variables. Each metabolite has a set of metadata from the lab, including a biochemical name, pathway information, and various platform information. This dataset also includes derived metadata including the number/proportion of samples missing a given metabolite and CV quality scores.

Details

The Parkinsons_HILICPOS file included in the package should include everything an analyst requires to complete a metabolomics project with these data. A link to the original Rdata file is listed in the references. If you need more information on these data, please see the documentation file at that file path.

References

S:\CPS\BIOSPECIMENS\Metabolomics\Parkinsons disease\

Examples

```
Parkinsons_HILICPOS <- getMetabolites("Parkinsons_HILICPOS")
metabolites <- Parkinsons_HILICPOS$metabolites
ID <- Parkinsons_HILICPOS$ID
biochem <- Parkinsons_HILICPOS$biochem
```

`prostate_metabolomics` *Prostate cancer metabolomics file*

Description

These data are a subset of our original prostate cancer metabolomics ACS0-01-16MD+. It includes the cleaned analytic metabolite file, a data set of metadata describing each metabolite, and an ID conversion file. The original data is a case-cohort design.

Usage

```
getMetabolites("prostate_metabolomics")
```

Format

A list with 3 data frames

`metabolites` A data frame of 556 observations and 1,266 variables. These metabolites were scaled to the daily median by Metabolon and the missings were imputed to the minimum. QC duplicates were averaged to provide only a single sample. No exclusions have been made: there are 556 unique samples in the data, and 1,265 metabolites (plus an ID variable)

`ID` A data frame of 556 observations and 5 variables. This is an ID conversion file for the prostate cancer metabolomics. The original data from Metabolon had several IDs, this file will help you merge if you need to go back to the original. Also includes the case-control and a subtype variable.

`biochem` A data frame of 1,265 observations and 17 variables. Each metabolite has a set of metadata from Metabolon, including a biochemical name, pathway information, and various platform information. This dataset also includes derived metadata including the number/proportion of samples missing a given metabolite and ICC_TECH / CV quality scores.

Details

The prostate_metabolomics file included in the package should include everything an analyst requires to complete a metabolomics project with these data. A link to the original Rdata file is listed in the references. If you need more information on the prostate cancer metabolomics data, please see the documentation file at that file path.

References

S:\CPS\BIOSPECIMENS\Metabolomics\Prostate Cancer - ACS0-01-16MD+\

Examples

```
prostate_metabolomics <- getMetabolites("prostate_metabolomics")
metabolites <- prostate_metabolomics$metabolites
ID <- prostate_metabolomics$ID
biochem <- prostate_metabolomics$biochem
```

qc *Example dataset for documenting the icc9() function*

Description

Sample dataset using scrambled and deidentified metabolomics data from the American Cancer Society's Cancer Prevention Study II.

Usage

```
data("qc")
```

Format

A data frame with 138 observations on the following 6 variables.

`subjid` a character vector of subject IDs, triplicates of 46 unique subjects

`BATCH1` First numeric batch variable for fixed effects adjustment

`BATCH2` Second numeric batch variable for fixed effects adjustment

`X48761` Normalized metabolite measurement

`X19130` Normalized metabolite measurement

`X53174` Normalized metabolite measurement

Examples

```
head(qc)
```

survey *Sample covariate file used in metabolomics analyses*

Description

This is real lifelink survey data, subset to only include a few variables.

Usage

```
data("survey")
```

Format

A data frame with 1534 observations on the following 5 variables.

`ID` Subject identifier

`BMI` BMI at blood draw - continuous numeric

`AGE_INT` Age at blood draw - continuous numeric

`LASTATE` a factor for time since last meal

`BMIBIN` Binary BMI outcome (0= <25; 1=>30)

Examples

```
data(survey)
str(survey)
```

`Validation_Blood` *Metabolomics in the CPS3 validation study - Blood samples*

Description

These data are a subset of our original metabolomics data collected for the CPS3 validation study. It includes the cleaned analytic metabolite file, a data set of metadata describing each metabolite, and an ID conversion file.

The data were collected at two time points, so the metabolite dataset includes a "Round" variable indicating the timing of data collection.

Usage

```
getMetabolites("Validation_Blood")
```

Format

A list with 3 data frames

`metabolites` A data frame of 1,472 observations and 1,370 variables. These metabolites were scaled to the daily median by Metabolon and the missings were imputed to the minimum. Samples were further normalized by sample volume. QC duplicates were averaged to provide only a single sample. No exclusions have been made, but there are duplicates. Duplicate samples were collected over two period, and there is a "Round" indicating the timing of each collection. In total there are 757 unique CPS3 participants in the file.

`ID` A data frame of 1,533 observations and 3 variables. The original data from Metabolon had several IDs, this file will help you merge if you need to go back to the original.

`biochem` A data frame of 1,368 observations and 17 variables. Each metabolite has a set of metadata from Metabolon, including a biochemical name, pathway information, and various platform information. This dataset also includes derived metadata including the number/proportion of samples missing a given metabolite and ICC_TECH / CV quality scores.

Details

The Validation_Blood file included in the package should include everything an analyst requires to complete a metabolomics project with these data. A link to the original Rdata file is listed in the references. If you need more information on these data, please see the documentation file at that file path.

References

S:\CPS3\Biospecimens\Metabolomics\Validation study blood - ACS0-01-18ML\

Examples

```
Validation_Blood <- getMetabolites("Validation_Blood")
metabolites <- Validation_Blood$metabolites
ID <- Validation_Blood$ID
biochem <- Validation_Blood$biochem
```

Validation_Urine *Metabolomics in the CPS3 validation study - Urine samples*

Description

These data are a subset of our original metabolomics data collected for the CPS3 validation study. It includes the cleaned analytic metabolite file, a data set of metadata describing each metabolite, and an ID conversion file.

The data were collected at two time points, so the metabolite dataset includes a "Round" variable indicating the timing of data collection.

Usage

```
getMetabolites("Validation_Urine")
```

Format

A list with 3 data frames

`metabolites` A data frame of 1,473 observations and 1,553 variables. These metabolites were scaled to the daily median by Metabolon and the missings were imputed to the minimum. Samples are further normalized by sample osmolality. QC triplicates were averaged to provide only a single sample. No exclusions have been made, but there are duplicates. Duplicate samples were collected over two periods, and there is a "Round" indicating the timing of each collection. In total there are 757 unique CPS3 participants in the file.

`ID` A data frame of 1,561 observations and 3 variables. The original data from Metabolon had several IDs, this file will help you merge if you need to go back to the original.

`biochem` A data frame of 1,551 observations and 17 variables. Each metabolite has a set of metadata from Metabolon, including a biochemical name, pathway information, and various platform information. This dataset also includes derived metadata including the number/proportion of samples missing a given metabolite and ICC_TECH / CV quality scores.

Details

The Validation_Urine file included in the package should include everything an analyst requires to complete a metabolomics project with these data. A link to the original Rdata file is listed in the references. If you need more information on these data, please see the documentation file at that file path.

References

S:\CPS3\Biospecimens\Metabolomics\Validation study urine - ACS0-02-18ML\

Examples

```
Validation_Urine <- getMetabolites("Validation_Urine")
metabolites <- Validation_Urine$metabolites
ID <- Validation_Urine$ID
biochem <- Validation_Urine$biochem
```

Index