Package ‘AMARETTO’

April 29, 2026

Type Package

Title Regulatory Network Inference and Driver Gene Evaluation using Integrative Multi-Omics Analysis and Penalized Regression

Version 1.29.0

Date 2016-06-06

Author Jayendra Shinde, Celine Everaert, Shaimaa Bakr, Mohsen Nabian, Jishu Xu, Vincent Carey, Nathalie Pochet and Olivier Gevaert

Maintainer Olivier Gevaert `<olivier.gevaert@gmail.com>`

Depends R (>= 3.6), impute, doParallel, grDevices, dplyr, methods, ComplexHeatmap

Description Integrating an increasing number of available multi-omics cancer data remains one of the main challenges to improve our understanding of cancer. One of the main challenges is using multi-omics data for identifying novel cancer driver genes. We have developed an algorithm, called AMARETTO, that integrates copy number, DNA methylation and gene expression data to identify a set of driver genes by analyzing cancer samples and connects them to clusters of co-expressed genes, which we define as modules. We applied AMARETTO in a pancancer setting to identify cancer driver genes and their modules on multiple cancer sites. AMARETTO captures modules enriched in angiogenesis, cell cycle and EMT, and modules that accurately predict survival and molecular subtypes. This allows AMARETTO to identify novel cancer driver genes directing canonical cancer pathways.

License Apache License (== 2.0) + file LICENSE

LazyLoad yes

LazyData true

Encoding UTF-8

biocViews
StatisticalMethod,DifferentialMethylation,GeneRegulation,GeneExpression,MethylationArray,Transcription,Preproce

Suggests testthat, MASS, knitr, BiocStyle

NeedsCompilation no

Imports callr (>= 3.0.0.9001), Matrix, Rcpp, BiocFileCache, DT, MultiAssayExperiment, circlize, curatedTCGAData, foreach, glmnet, httr, limma, matrixStats, readr, reshape2, tibble, rmarkdown, graphics, grid, parallel, stats, knitr, ggplot2, gridExtra, utils

Roxygen list(markdown = TRUE)

RoxygenNote 6.1.1.9000

LinkingTo Rcpp

VignetteBuilder knitr

git_url https://git.bioconductor.org/packages/AMARETTO

git_branch devel

git_last_commit 05ae352

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-04-29

Contents

`AMARETTO_CreateModuleData`

AMARETTO_CreateModuleData

Description

AMARETTO_CreateModuleData

Usage

```
AMARETTO_CreateModuleData(AMARETTOinit, AMARETTOresults)
```

Arguments

`AMARETTOinit`	List output from AMARETTO_Initialize().
`AMARETTOresults`	List output from AMARETTO_Run()

Value

result

Examples

```
data('ProcessedDataLIHC')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)
AMARETTOresults <- AMARETTO_Run(AMARETTOinit)
AMARETTO_MD <- AMARETTO_CreateModuleData(AMARETTOinit, AMARETTOresults)
```

`AMARETTO_CreateRegulatorPrograms`

AMARETTO_CreateRegulatorPrograms

Description

AMARETTO_CreateRegulatorPrograms

Usage

```
AMARETTO_CreateRegulatorPrograms(AMARETTOinit, AMARETTOresults)
```

Arguments

`AMARETTOinit`	List output from AMARETTO_Initialize().
`AMARETTOresults`	List output from AMARETTO_Run()

Value

result

Examples

```
data('ProcessedDataLIHC')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)
AMARETTOresults <- AMARETTO_Run(AMARETTOinit)
AMARETTO_RP <- AMARETTO_CreateRegulatorPrograms(AMARETTOinit,AMARETTOresults)
```

`AMARETTO_Download`

AMARETTO_Download

Description

Downloading TCGA dataset for AMARETTO analysis

Usage

```
AMARETTO_Download(CancerSite = "CHOL",
  TargetDirectory = TargetDirectory)
```

Arguments

`CancerSite`	TCGA cancer code for data download
`TargetDirectory`	Directory path to download data

Value

result

Examples

```
TargetDirectory <- file.path(getwd(),"Downloads/");dir.create(TargetDirectory)
CancerSite <- 'CHOL'
DataSetDirectories <- AMARETTO_Download(CancerSite,TargetDirectory = TargetDirectory)
```

`AMARETTO_EvaluateTestSet` *AMARETTO_EvaluateTestSet*

Description

Code to evaluate AMARETTO on a new gene expression test set. Uses output from AMARETTO_Run() and CreateRegulatorData().

Usage

```
AMARETTO_EvaluateTestSet(AMARETTOresults = AMARETTOresults,
  MA_Data_TestSet = MA_Data_TestSet,
  RegulatorData_TestSet = RegulatorData_TestSet)
```

Arguments

`AMARETTOresults`	AMARETTO output from AMARETTO_Run().
`MA_Data_TestSet`	Gene expression matrix from a test set (that was not used in AMARETTO_Run()).
`RegulatorData_TestSet`	Test regulator data from CreateRegulatorData().

Value

result

Examples

```
data('ProcessedDataLIHC')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)

AMARETTOresults <- AMARETTO_Run(AMARETTOinit)
AMARETTOtestReport <- AMARETTO_EvaluateTestSet(AMARETTOresults = AMARETTOresults,
                                             MA_Data_TestSet = AMARETTOinit$MA_matrix_Var,
                                      RegulatorData_TestSet = AMARETTOinit$RegulatorData)
```

`AMARETTO_ExportResults` *AMARETTO_ExportResults*

Description

Retrieve a download of all the data linked with the run (including heatmaps)

Usage

```
AMARETTO_ExportResults(AMARETTOinit, AMARETTOresults, data_address,
  Heatmaps = TRUE, CNV_matrix = NULL, MET_matrix = NULL)
```

Arguments

`AMARETTOinit`	AMARETTO initialize output
`AMARETTOresults`	AMARETTO results output
`data_address`	Directory to save data folder
`Heatmaps`	Output heatmaps as pdf
`CNV_matrix`	CNV_matrix
`MET_matrix`	MET_matrix

Value

result

Examples

```
data('ProcessedDataLIHC')
TargetDirectory <- file.path(getwd(),"Downloads/");dir.create(TargetDirectory)
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)

AMARETTOresults <- AMARETTO_Run(AMARETTOinit)
AMARETTO_ExportResults(AMARETTOinit,AMARETTOresults,TargetDirectory,Heatmaps = FALSE)
```

`AMARETTO_HTMLreport` *AMARETTO_HTMLreport*

Description

Retrieve an interactive html report, including gene set enrichment analysis if asked for.

Usage

```
AMARETTO_HTMLreport(AMARETTOinit, AMARETTOresults, ProcessedData,
  show_row_names = FALSE, SAMPLE_annotation = NULL, ID = NULL,
  hyper_geo_test_bool = FALSE, hyper_geo_reference = NULL,
  output_address = "./", MSIGDB = TRUE, driverGSEA = TRUE,
  phenotype_association_table = NULL)
```

Arguments

`AMARETTOinit`	AMARETTO initialize output
`AMARETTOresults`	AMARETTO results output
`ProcessedData`	List of processed input data
`show_row_names`	if True, sample names will appear in the heatmap
`SAMPLE_annotation`	SAMPLE annotation will be added to heatmap
`ID`	ID column of the SAMPLE annotation data frame
`hyper_geo_test_bool`	Boolean if a hyper geometric test needs to be performed. If TRUE provide a GMT file in the hyper_geo_reference parameter.
`hyper_geo_reference`	GMT file with gene sets to compare with.
`output_address`	Output directory for the html files.
`MSIGDB`	TRUE if gene sets were retrieved from MSIGDB. Links will be created in the report.
`driverGSEA`	if TRUE, module drivers will also be included in the hypergeometric test.
`phenotype_association_table`	a Data Frame, containing all modules phenotype association data. Optional.

Value

result

Examples

```
## Not run:
data('ProcessedDataLIHC')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)

AMARETTOresults <- AMARETTO_Run(AMARETTOinit)

AMARETTO_HTMLreport(AMARETTOinit= AMARETTOinit,AMARETTOresults= AMARETTOresults,
                    ProcessedData = ProcessedDataLIHC,
                    hyper_geo_test_bool=FALSE,
                    output_address='./')

## End(Not run)
```

`AMARETTO_Initialize` *AMARETTO_Initialize (version: reorder and filter MA_Matrix)*

Description

Code used to initialize the seed clusters for an AMARETTO run. Requires processed gene expresiosn (rna-seq or microarray), CNV (usually from a GISTIC run), and methylation (from MethylMix, provided in this package) data. Uses the function CreateRegulatorData() and results are fed into the function AMARETTO_Run().

Usage

```
AMARETTO_Initialize(ProcessedData = ProcessedData, Driver_list = NULL,
  NrModules, VarPercentage, PvalueThreshold = 0.001,
  RsquareThreshold = 0.1, pmax = 10, NrCores = 1, OneRunStop = 0,
  method = "union", random_seeds = NULL, convergence_cutoff = 0.01)
```

Arguments

`ProcessedData`	List of Expression, CNV and MethylMix data matrices, with genes in rows and samples in columns.
`Driver_list`	Custom list of driver genes to be considered in analysis
`NrModules`	How many gene co-expression modules should AMARETTO search for? Usually around 100 is acceptable, given the large number of possible driver-passenger gene combinations.
`VarPercentage`	Minimum percentage by variance for filtering of genes; for example, 75% would indicate that the CreateRegulatorData() function only analyses genes that have a variance above the 75th percentile across all samples.
`PvalueThreshold`	Threshold used to find relevant driver genes with CNV alterations: maximal p-value.
`RsquareThreshold`	Threshold used to find relevant driver genes with CNV alterations: minimal R-square value between CNV and gene expression data.
`pmax`	'pmax' variable for glmnet function from glmnet package; the maximum number of variables aver to be nonzero. Should not be changed by user unless she/he fully understands the AMARETTO algorithm and how its parameters choices affect model output.
`NrCores`	A numeric variable indicating the number of computer/server cores to use for paralellelization. Default is 1, i.e. no parallelization. Please check your computer or server's computing capacities before increasing this number. Parallelization is done via the RParallel package. Mac vs. Windows environments may behave differently when using parallelization.
`OneRunStop`	OneRunStop
`method`	Perform union or intersection of the driver genes evaluated from the input data matrices and custom driver gene list provided.
`random_seeds`	A numeric vector of length 2, containing two seed numbers for randomization : 1st for kmeans and 2nd for glmnet
`convergence_cutoff`	A numeric value (E.g. 0.01) representing the fraction of the total number of genes, in which, The algorithm is considered reaching convergence and will stop, if Nr of Gene-replacements in an iteration falls below this threshold * total number of genes.

Value

result

Examples

```
data('ProcessedDataLIHC')
data('Driver_Genes')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)
## Not run:
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    Driver_list = Driver_Genes[['MSigDB']],
                                    NrModules = 2, VarPercentage = 50)

## End(Not run)
```

`AMARETTO_LarsenBased` *AMARETTO_LarsenBased*

Description

AMARETTO_LarsenBased

Usage

```
AMARETTO_LarsenBased(Data, Clusters, RegulatorData, Parameters, NrCores,
  random_seeds, convergence_cutoff)
```

Arguments

`Data`

`Clusters`

`RegulatorData`

`Parameters`

`NrCores`

`random_seeds`

`convergence_cutoff`

Value

result

`AMARETTO_LearnRegulatoryProgramsLarsen` *AMARETTO_LearnRegulatoryProgramsLarsen*

Description

AMARETTO_LearnRegulatoryProgramsLarsen

Usage

```
AMARETTO_LearnRegulatoryProgramsLarsen(Data, Clusters, RegulatorData,
  RegulatorSign, Lambda, AutoRegulation, alpha, pmax, random_seeds)
```

Value

result

`AMARETTO_Preprocess` *AMARETTO_Preprocess*

Description

Wrapper code that analyzes process TCGA GISTIC (CNV) and gene expression (rna-seq or microarray) data via one call

Usage

```
AMARETTO_Preprocess(DataSetDirectories = DataSetDirectories,
  BatchData = BatchData)
```

Arguments

`DataSetDirectories`	DataSetDirectories
`BatchData`	BatchData

Value

result

Examples

```
## Not run:
TargetDirectory <-  "Downloads"  # path to data download directory
CancerSite <- 'CHOL'
DataSetDirectories <- AMARETTO_Download(CancerSite,TargetDirectory)
ProcessedData <- AMARETTO_Preprocess(DataSetDirectories,BatchData)

## End(Not run)
```

`AMARETTO_ReassignGenesToClusters` *AMARETTO_ReassignGenesToClusters*

Description

AMARETTO_ReassignGenesToClusters

Usage

```
AMARETTO_ReassignGenesToClusters(Data, RegulatorData, Beta, Clusters,
  AutoRegulation)
```

Value

result

`AMARETTO_Run` *AMARETTO_Run Function to run AMARETTO, a statistical algorithm to identify cancer drivers by integrating a variety of omics data from cancer and normal tissue.*

Description

AMARETTO_Run Function to run AMARETTO, a statistical algorithm to identify cancer drivers by integrating a variety of omics data from cancer and normal tissue.

Usage

```
AMARETTO_Run(AMARETTOinit)
```

Arguments

`AMARETTOinit`	List output from AMARETTO_Initialize().

Value

result

Examples

```
data('ProcessedDataLIHC')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)
AMARETTOresults <- AMARETTO_Run(AMARETTOinit)
```

`AMARETTO_VisualizeModule`

AMARETTO_VisualizeModule

Description

Function to visualize the gene modules

Usage

```
AMARETTO_VisualizeModule(AMARETTOinit, AMARETTOresults, ProcessedData,
  ModuleNr, show_row_names = FALSE, SAMPLE_annotation = NULL,
  ID = NULL, order_samples = NULL)
```

Arguments

`AMARETTOinit`	List output from AMARETTO_Initialize().
`AMARETTOresults`	List output from AMARETTO_Run().
`ProcessedData`	List of processed input data
`ModuleNr`	Module number to visualize
`show_row_names`	If TRUE, row names will be shown on the plot.
`SAMPLE_annotation`	Matrix or Dataframe with sample annotation
`ID`	Column used as sample name
`order_samples`	Order samples in heatmap by mean or by clustering

Value

result

Examples

```
data('ProcessedDataLIHC')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)

AMARETTOresults <- AMARETTO_Run(AMARETTOinit)

AMARETTO_VisualizeModule(AMARETTOinit = AMARETTOinit,AMARETTOresults = AMARETTOresults,
                         ProcessedData = ProcessedDataLIHC, ModuleNr = 1)
```

`aprior` *aprior*

Description

Following four find empirical hyper-prior values

Usage

```
aprior(gamma.hat)
```

Value

result

`BatchData` *BatchData*

Description

A dataset for conducting batch corerction in TCGA samples

Usage

```
BatchData
```

Format

A data frame with 23263 observations and 3 variables:

Source

`AMARETTO`

`Beta.NA` *Beta.NA*

Description

Beta.NA

Usage

```
Beta.NA(y, X)
```

Value

result

`bprior` *bprior*

Description

bprior

Usage

```
bprior(gamma.hat)
```

Value

result

`build.design` *build.design*

Description

Next two functions make the design matrix (X) from the sample info file

Usage

```
build.design(vec, des = NULL, start = 2)
```

Value

result

`cacheResource` *cacheResource*

Description

cacheResource

Usage

```
cacheResource(TargetDirectory = TargetDirectory, resource = resource)
```

Value

result

`ComBat_NoFiles` *ComBat_NoFiles*

Description

ComBat_NoFiles

Usage

```
ComBat_NoFiles(dat, saminfo, type = "txt", write = FALSE,
  covariates = "all", par.prior = TRUE, filter = FALSE, skip = 0,
  prior.plots = FALSE)
```

Value

result

`computeGisticURL` *computeGisticURL*

Description

computeGisticURL

Usage

```
computeGisticURL(url = NULL, acronym = "CHOL")
```

Value

result

`CreateRegulatorData` *CreateRegulatorData*

Description

Determine potential regulator genes.

Usage

```
CreateRegulatorData(MA_matrix = MA_matrix, CNV_matrix = NULL,
  MET_matrix = NULL, Driver_list = NULL, PvalueThreshold = 0.001,
  RsquareThreshold = 0.1, method = "union")
```

Value

result

`design.mat` *design.mat*

Description

design.mat

Usage

```
design.mat(saminfo)
```

Value

result

`Driver_Genes` *Driver_Genes*

Description

A list of cancer driver genes described in literature.

Usage

```
Driver_Genes
```

Format

List

Source

`AMARETTO`

`filter.absent` *filter.absent*

Description

filters data based on presence/absence call

Usage

```
## S3 method for class 'absent'
filter(x, pct)
```

Value

result

`FindTranscriptionallyPredictive_CNV` *FindTranscriptionallyPredictive_CNV*

Description

Function to identify which genes CNV significantly predict expression of that gene.

Usage

```
FindTranscriptionallyPredictive_CNV(MA_matrix, CNV_matrix,
  PvalueThreshold = 0.001, RsquareThreshold = 0.1)
```

Value

result

`geneFiltering` *geneFiltering*

Description

Function to filter gene expression matrix

Usage

```
geneFiltering(Type, MAdata, Percentage)
```

Value

result

`GeneSetDescription` *GeneSetDescription*

Description

GeneSetDescription

Usage

```
GeneSetDescription(filename, MSIGDB)
```

Arguments

`filename`	The name of the gmt file.
`MSIGDB`	If True, the gene set description column will be provided from MSIGDB.

Value

result

`get_firehoseData` *get_firehoseData*

Description

Downloading TCGA dataset via firehose

Usage

```
get_firehoseData(TargetDirectory = "./",
  TCGA_acronym_uppercase = "LUAD", dataType = "stddata",
  dataFileTag = "mRNAseq_Preprocess.Level_3", FFPE = FALSE,
  fileType = "tar.gz",
  gdacURL = "http://gdac.broadinstitute.org/runs/", untarUngzip = TRUE,
  printDisease_abbr = FALSE)
```

Value

result

`GmtFromModules` *GmtFromModules*

Description

GmtFromModules

Usage

```
GmtFromModules(AMARETTOinit, AMARETTOresults, driverGSEA)
```

Arguments

`AMARETTOinit`	List output from AMARETTO_Initialize().
`AMARETTOresults`	List output from AMARETTO_Run().
`driverGSEA`	if TRUE , module driver genes will also be added to module target genes for GSEA.

Value

result

`HyperGTestGeneEnrichment` *Hyper Geometric Geneset Enrichement Test*

Description

Calculates the p-values for unranked gene set enrichment based on two gmt files as input and the hyper geometric test.

Usage

```
HyperGTestGeneEnrichment(gmtfile, testgmtfile, NrCores,
  ref.numb.genes = 45956)
```

Arguments

`gmtfile`	The gmt file with reference gene set.
`testgmtfile`	The gmt file with gene sets to test. In our case, the gmt file of the modules.
`NrCores`	Number of cores used for parallelization.
`ref.numb.genes`	The total number of genes teste, standard equal to 45 956 (MSIGDB standard).

Value

result

`int.eprior` *int.eprior*

Description

Monte Carlo integration function to find the nonparametric adjustments

Usage

```
int.eprior(sdat, g.hat, d.hat)
```

Value

result

`it.sol` *it.sol*

Description

Pass in entire data set, the design matrix for the entire data, the batch means, the batch variances, priors (m, t2, a, b), columns of the data matrix for the batch. Uses the EM to find the parametric batch adjustments

Usage

```
it.sol(sdat, g.hat, d.hat, g.bar, t2, a, b, conv = 1e-04)
```

Value

result

`L` *L*

Description

likelihood function

Usage

```
L(x, g.hat, d.hat)
```

Value

result

`Lambda_Sequence` *Lambda_Sequence*

Description

Lambda_Sequence

Usage

```
Lambda_Sequence(sx, sy)
```

Value

result

`list.batch` *list.batch*

Description

Makes a list with elements pointing to which array belongs to which batch

Usage

```
list.batch(saminfo)
```

Value

result

`MsigdbMapping` *MsigdbMapping*

Description

A dataset containing all MSIGDB pathways and their descriptions. .

Usage

```
MsigdbMapping
```

Format

List

Source

`AMARETTO`

`plot_run_history` *Title plot_run_history*

Description

Title plot_run_history

Usage

```
plot_run_history(AMARETTOinit, AMARETTOresults)
```

Arguments

`AMARETTOinit`	AMARETTO initialize output
`AMARETTOresults`	AMARETTO results output

Value

plot

Examples

```
data('ProcessedDataLIHC')
AMARETTOinit <- AMARETTO_Initialize(ProcessedData = ProcessedDataLIHC,
                                    NrModules = 2, VarPercentage = 50)

AMARETTOresults <- AMARETTO_Run(AMARETTOinit)

plot_run_history(AMARETTOinit,AMARETTOresults)
```

`postmean` *postmean*

Description

postmean

Usage

```
postmean(g.hat, g.bar, n, d.star, t2)
```

Value

result

`postvar` *postvar*

Description

postvar

Usage

```
postvar(sum2, n, a, b)
```

Value

result

`Preprocess_MAdata` *Preprocess_MAdata*

Description

Preprocess_MAdata

Usage

```
Preprocess_MAdata(CancerSite = CancerSite, MAEO_ge = MAEO_ge,
  BatchData = BatchData)
```

Value

result

`printf` *printf*

Description

Wrapper function for C-style formatted output.

Usage

```
printf(...)
```

Value

result

`ProcessedDataLIHC` *ProcessedDataLIHC*

Description

A list of dataframes of processed toy example dataset from TCGA-LIHC.

Usage

```
ProcessedDataLIHC
```

Format

List

Source

`AMARETTO`

`readGMT` *readGMT*

Description

readGMT

Usage

```
readGMT(filename)
```

Arguments

`filename`

Value

result

`read_gct` *read_gct*

Description

Function to turn a .gct data files into a matrix format

Usage

```
read_gct(file_address)
```

Arguments

`file_address`	Address of the input gct file.

Value

result

Examples

```
data_matrix<-read_gct(file_address="")
```

`Save_CancerSite` *Save_CancerSite*

Description

Save_CancerSite

Usage

```
Save_CancerSite(CancerSite, TargetDirectory, DataSetDirectories,
  ProcessedData)
```

Value

result

`TCGA_BatchCorrection_MolecularData` *TCGA_BatchCorrection_MolecularData*

Description

TCGA_BatchCorrection_MolecularData

Usage

```
TCGA_BatchCorrection_MolecularData(GEN_Data = GEN_Data,
  BatchData = BatchData, MinInBatch = MinInBatch)
```

Value

result

`TCGA_GENERIC_BatchCorrection` *TCGA_GENERIC_BatchCorrection*

Description

TCGA_GENERIC_BatchCorrection

Usage

```
TCGA_GENERIC_BatchCorrection(GEN_Data = GEN_Data,
  BatchData = BatchData)
```

Value

result

`TCGA_GENERIC_CheckBatchEffect`

TCGA_GENERIC_CheckBatchEffect

Description

TCGA_GENERIC_CheckBatchEffect

Usage

```
TCGA_GENERIC_CheckBatchEffect(GEN_Data, BatchData)
```

Value

result

`TCGA_GENERIC_CleanUpSampleNames`

TCGA_GENERIC_CleanUpSampleNames

Description

TCGA_GENERIC_CleanUpSampleNames

Usage

```
TCGA_GENERIC_CleanUpSampleNames(GEN_Data = GEN_Data, IDlength = 12)
```

Value

result

`TCGA_GENERIC_GetSampleGroups`

TCGA_GENERIC_GetSampleGroups

Description

TCGA_GENERIC_GetSampleGroups

Usage

```
TCGA_GENERIC_GetSampleGroups(SampleNames)
```

Value

result

`TCGA_GENERIC_MergeData`

TCGA_GENERIC_MergeData

Description

TCGA_GENERIC_MergeData

Usage

```
TCGA_GENERIC_MergeData(NewIDListUnique, DataMatrix, MergeMethod)
```

Value

result

`TCGA_Load_GISTICdata` *TCGA_Load_GISTICdata*

Description

TCGA_Load_GISTICdata

Usage

```
TCGA_Load_GISTICdata(GisticDirectory)
```

Value

result

`TCGA_Load_MolecularData`

TCGA_Load_MolecularData

Description

TCGA_Load_MolecularData

Usage

```
TCGA_Load_MolecularData(MAEO_ge)
```

Value

result

`trim.dat` *trim.dat*

Description

Trims the data of extra columns, note your array names cannot be named 'X' or start with 'X.'

Usage

```
trim.dat(dat)
```

Value

result

`write_gct` *write_gct*

Description

write_gct

Usage

```
write_gct(data_in, file_address)
```

Value

result

Index