

# Package ‘pathwayPCA’

October 18, 2022

**Type** Package

**Title** Integrative Pathway Analysis with Modern PCA Methodology and Gene Selection

**Version** 1.12.0

**Description** pathwayPCA is an integrative analysis tool that implements the principal component analysis (PCA) based pathway analysis approaches described in Chen et al. (2008), Chen et al. (2010), and Chen (2011). pathwayPCA allows users to: (1) Test pathway association with binary, continuous, or survival phenotypes. (2) Extract relevant genes in the pathways using the SuperPCA and AES-PCA approaches. (3) Compute principal components (PCs) based on the selected genes. These estimated latent variables represent pathway activities for individual subjects, which can then be used to perform integrative pathway analysis, such as multi-omics analysis. (4) Extract relevant genes that drive pathway significance as well as data corresponding to these relevant genes for additional in-depth analysis. (5) Perform analyses with enhanced computational efficiency with parallel computing and enhanced data safety with S4-class data objects. (6) Analyze studies with complex experimental designs, with multiple covariates, and with interaction effects, e.g., testing whether pathway association with clinical phenotype is different between male and female subjects.

Citations: Chen et al. (2008) <<https://doi.org/10.1093/bioinformatics/btn458>>; Chen et al. (2010) <<https://doi.org/10.1002/gepi.20532>>; and Chen (2011) <<https://doi.org/10.2202/1544-6115.1697>>.

**License** GPL-3

**Depends** R (>= 3.1)

**Imports** lars, methods, parallel, stats, survival, utils

**Suggests** airway, circlize, grDevices, knitr, RCurl, reshape2, rmarkdown, SummarizedExperiment, survminer, testthat, tidyverse

**biocViews** CopyNumberVariation, DNAMethylation, GeneExpression, SNP, Transcription, GenePrediction, GeneSetEnrichment, GeneSignaling, GeneTarget, GenomeWideAssociation, GenomicVariation, CellBiology, Epigenetics, FunctionalGenomics, Genetics, Lipidomics, Metabolomics, Proteomics, SystemsBiology, Transcriptomics, Classification, DimensionReduction,

FeatureExtraction, PrincipalComponent, Regression, Survival,  
MultipleComparison, Pathways

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.1.2

**Collate** 'CreatePathwayCollection.R' 'createClass\_OmicsPath.R'  
'createClass\_validOmics.R' 'accessClass\_OmicsPath.R'  
'createClass\_OmicsSurv.R' 'accessClass\_OmicsSurv.R'  
'accessClass\_OmicsRegCateg.R' 'createClass\_OmicsCateg.R'  
'createClass\_OmicsReg.R' 'accessClass\_OmicsPathData.R'  
'accessClass\_pathwayCollection.R'  
'accessClass\_pathwayCollection\_which.R' 'accessClass\_pcOut.R'  
'accessClass\_pcOutpVals.R' 'aesPC\_calculate\_AESPCA.R'  
'aesPC\_calculate\_LARS.R' 'aesPC\_extract\_OmicsPath\_PCs.R'  
'aesPC\_permtest\_CoxPH.R' 'aesPC\_permtest\_GLM.R'  
'aesPC\_permtest\_LM.R' 'aesPC\_unknown\_matrixNorm.R'  
'aesPC\_wrapper.R' 'createOmics\_All.R'  
'createOmics\_CheckAssay.R'  
'createOmics\_CheckPathwayCollection.R'  
'createOmics\_CheckSampleIDs.R' 'createOmics\_JoinPhenoAssay.R'  
'createOmics\_TrimPathwayCollection.R' 'createOmics\_Wrapper.R'  
'data\_colonSubset.R' 'data\_genesetSubset.R'  
'data\_wikipathways.R' 'data\_wikipathways\_symbols.R'  
'pathwayPCA.R' 'printClass\_Omics\_All.R'  
'printClass\_pathwayCollection.R' 'superPC\_model\_CoxPH.R'  
'superPC\_model\_GLM.R' 'superPC\_model\_LS.R'  
'superPC\_model\_tStats.R' 'superPC\_model\_train.R'  
'superPC\_modifiedSVD.R' 'superPC\_optimWeibullParams.R'  
'superPC\_optimWeibull\_pValues.R' 'superPC\_pathway\_tControl.R'  
'superPC\_pathway\_tScores.R' 'superPC\_pathway\_tValues.R'  
'superPC\_permuteSamples.R' 'superPC\_wrapper.R'  
'utils\_Contains.R' 'utils\_adjust\_and\_sort\_pValues.R'  
'utils\_load\_test\_data\_onto\_PCs.R' 'utils\_multtest\_pvalues.R'  
'utils\_read\_gmt.R' 'utils\_stdExpr\_2\_tidyAssay.R'  
'utils\_transpose\_assay.R' 'utils\_write\_gmt.R'

**VignetteBuilder** knitr

**URL** <<https://gabrielodom.github.io/pathwayPCA/>>

**BugReports** <https://github.com/gabrielodom/pathwayPCA/issues>

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

**git\_branch** RELEASE\_3\_15

**git\_last\_commit** e3fc7f5

**git\_last\_commit\_date** 2022-04-26

**Date/Publication** 2022-10-18

**Author** Gabriel Odom [aut, cre],  
 James Ban [aut],  
 Lizhong Liu [aut],  
 Lily Wang [aut],  
 Steven Chen [aut]

**Maintainer** Gabriel Odom <gabriel.odom@med.miami.edu>

## R topics documented:

aespcapca	4
AESPCA_pVals	5
colonSurv_df	7
colon_pathwayCollection	8
Contains	9
CreateOmics	10
CreateOmicsPath	12
CreatePathwayCollection	15
getPathPCLs	16
getPathpVals	18
LoadOntoPCs	20
OmicsCateg-class	21
OmicsPathway-class	22
OmicsReg-class	23
OmicsSurv-class	23
pathwayPCA	24
read_gmt	25
SE2Tidy	26
SubsetOmicsPath	27
SubsetOmicsResponse	29
SubsetOmicsSurv	31
SubsetPathwayCollection	32
SubsetPathwayData	33
SuperPCA_pVals	34
TransposeAssay	36
WhichPathways	38
wikipwsHS_Entrez_pathwayCollection	39
wikipwsHS_Symbol_pathwayCollection	39
write_gmt	40
<b>Index</b>	<b>42</b>

aespca

*Adaptive, elastic-net, sparse principal component analysis***Description**

A function to perform adaptive, elastic-net, sparse principal component analysis (AES-PCA).

**Usage**

```
aespca(X, d = 1, max.iter = 10, eps.conv = 0.001, adaptive = TRUE, para = NULL)
```

**Arguments**

<code>X</code>	A pathway design matrix: the data matrix should be $n \times p$ , where $n$ is the sample size and $p$ is the number of variables included in the pathway.
<code>d</code>	The number of principal components (PCs) to extract from the pathway. Defaults to 1.
<code>max.iter</code>	The maximum number of times an internal <code>while()</code> loop can make calls to the <code>lars.lsa()</code> function. Defaults to 10.
<code>eps.conv</code>	A numerical convergence threshold for the same <code>while()</code> loop. Defaults to 0.001.
<code>adaptive</code>	Internal argument of the <code>lars.lsa()</code> function. Defaults to TRUE.
<code>para</code>	Internal argument of the <code>lars.lsa()</code> function. Defaults to NULL.

**Details**

This function calculates the loadings and reduced-dimension predictor matrix using both the Singular Value Decomposition and AES-PCA Decomposition (as described in Efron et al (2003)) of the data matrix.

See [https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle\\_2002.pdf](https://web.stanford.edu/~hastie/Papers/LARS/LeastAngle_2002.pdf).

For potential enhancement details, see the comment in the "Details" section of [normalize](#).

**Value**

A list of four elements containing the loadings and projected predictors:

- `aesLoad` : A  $d \times p$  projection matrix of the  $d$  AES-PCs.
- `oldLoad` : A  $d \times p$  projection matrix of the  $d$  PCs from the singular value decomposition (SVD).
- `aesScore` : An  $n \times d$  predictor matrix: the original  $n$  observations loaded onto the  $d$  AES-PCs.
- `oldScore` : An  $n \times d$  predictor matrix: the original  $n$  observations loaded onto the  $d$  SVD-PCs.

**See Also**

[normalize](#); [lars.lsa](#); [ExtractAESPCs](#); [AESPCA\\_pVals](#)

**Examples**

```
# DO NOT CALL THIS FUNCTION DIRECTLY.
# Call this function through AESPCA_pVals() instead.

## Not run:
data("colonSurv_df")
aes pca(as.matrix(colonSurv_df[, 5:50]))

## End(Not run)
```

AESPCA\_pVals

*Test pathway association with AES-PCA***Description**

Given a supervised OmicsPath object (one of OmicsSurv, OmicsReg, or OmicsCateg), extract the first  $k$  adaptive, elastic-net, sparse principal components (PCs) from each pathway-subset of the features in the -Omics assay design matrix, test their association with the response matrix, and return a data frame of the adjusted  $p$ -values for each pathway.

**Usage**

```
AESPCA_pVals(
  object,
  numPCs = 1,
  numReps = 0L,
  parallel = FALSE,
  numCores = NULL,
  asPCA = FALSE,
  adjustpValues = TRUE,
  adjustment = c("Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BH", "BY",
    "ABH", "TSBH"),
  ...
)

## S4 method for signature 'OmicsPathway'
AESPCA_pVals(
  object,
  numPCs = 1,
  numReps = 1000,
  parallel = FALSE,
  numCores = NULL,
  asPCA = FALSE,
  adjustpValues = TRUE,
  adjustment = c("Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BH", "BY",
    "ABH", "TSBH"),
```

```
    ...
  )
```

### Arguments

object	An object of class <code>OmicPathway</code> with a response matrix or vector.
numPCs	The number of PCs to extract from each pathway. Defaults to 1.
numReps	How many permutations to estimate the $p$ -value? Defaults to 0 (that is, to estimate the $p$ -value parametrically). If <code>numReps &gt; 0</code> , then the non-parametric, permutation $p$ -value will be returned based on the number of random samples specified.
parallel	Should the computation be completed in parallel? Defaults to <code>FALSE</code> .
numCores	If <code>parallel = TRUE</code> , how many cores should be used for computation? Internally defaults to the number of available cores minus 1.
asPCA	Should the computation return the eigenvectors and eigenvalues instead of the adaptive, elastic-net, sparse principal components and their corresponding loadings. Defaults to <code>FALSE</code> ; this should be used for diagnostic or comparative purposes only.
adjustpValues	Should you adjust the $p$ -values for multiple comparisons? Defaults to <code>TRUE</code> .
adjustment	Character vector of procedures. The returned data frame will be sorted in ascending order by the first procedure in this vector, with ties broken by the unadjusted $p$ -value. If only one procedure is selected, then it is necessarily the first procedure. See the documentation for the <code>ControlFDR</code> function for the adjustment procedure definitions and citations.
...	Dots for additional internal arguments.

### Details

This is a wrapper function for the `ExtractAESPCs`, `PermTestSurv`, `PermTestReg`, and `PermTestCateg` functions.

Please see our Quickstart Guide for this package: [https://gabrielodom.github.io/pathwayPCA/articles/Supplement1-Quickstart\\_Guide.html](https://gabrielodom.github.io/pathwayPCA/articles/Supplement1-Quickstart_Guide.html)

### Value

A results list with class `aespcOut`. This list has three components: a data frame of pathway details, pathway  $p$ -values, and potential adjustments to those values (`pVals_df`); a list of the first `numPCs` *score* vectors for each pathway (`PCs_ls`); and a list of the first `numPCs` feature loading vectors for each pathway (`loadings_ls`). The  $p$ -value data frame has columns:

- `pathways` : The names of the pathways in the `Omic*` object (given in `object@trimPathwayCollection$pathways`.)
- `setsize` : The number of genes in each of the original pathways (given in the `object@trimPathwayCollection$setsize` object).
- `n_tested` : The number of genes in each of the trimmed pathways (given in the `object@trimPathwayCollection$n_tested` object).

- `terms` : The pathway description, as given in the `object@trimPathwayCollection$TERMS` object.
- `rawp` : The unadjusted  $p$ -values of each pathway.
- `...` : Additional columns of adjusted  $p$ -values as specified through the adjustment argument.

The data frame will be sorted in ascending order by the method specified first in the adjustment argument. If `adjustpValues = FALSE`, then the data frame will be sorted by the raw  $p$ -values. If you have the suggested tidyverse package suite loaded, then this data frame will print as a [tibble](#). Otherwise, it will print as a data frame.

### See Also

[CreateOmics](#); [ExtractAESPCs](#); [PermTestSurv](#); [PermTestReg](#); [PermTestCateg](#); [TabulatepValues](#); [clusterApply](#)

### Examples

```
### Load the Example Data ###
data("colonSurv_df")
data("colon_pathwayCollection")

### Create an OmicsSurv Object ###
colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:3],
  respType = "surv"
)

### Calculate Pathway p-Values ###
colonSurv_aespc <- AESPCA_pVals(
  object = colon_Omics,
  numReps = 0,
  parallel = TRUE,
  numCores = 2,
  adjustpValues = TRUE,
  adjustment = c("Hoch", "SidakSD")
)
```

---

colonSurv\_df

*Colon Cancer -Omics Data*

---

### Description

Subset of a colon cancer survival data set, with subject response and assay values.

### Usage

```
data(colonSurv_df)
```

**Format**

A subset of a data frame containing 656 of 2022 genes measured on 250 subjects. The first two columns are the Overall Survival time (OS\_time) and death indicator (OS\_event).

**Source**

GEO GSE17538 <https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE17538>

---

colon\_pathwayCollection

*Gene Pathway Subset*

---

**Description**

An example Canonical Pathways Gene Subset from the Broad Institute: File: c2.cp.v6.0.symbols.gmt.

**Usage**

```
data(colon_pathwayCollection)
```

**Format**

A pathwayCollection list of two elements:

- pathways : A list of 15 character vectors. Each vector contains the names of the individual genes within that pathway as a vector of character strings.
- TERMS : A character vector of length 15 containing the names of the gene pathways.

**Details**

This is a subset of 15 pathways from the Broad Institute pathways list. This subset contains seven pathways which are related to the response information in the [colonSurv\\_df](#) data file.

**Source**

<http://software.broadinstitute.org/gsea/msigdb/collections.jsp>



---

Contains	<i>Check if a long atomic vector contains a short atomic vector</i>
----------	---

---

### Description

Check if any or all of the elements of a short atomic vector are contained within a supplied long atomic vector.

### Usage

```
Contains(long, short, matches = c("any", "all"), partial = FALSE)
```

### Arguments

long	A vector to possibly containing any or all elements of short
short	A short vector or scalar, some elements of which may be contained in long
matches	Should partial set matching of short be allowed? Defaults to "any", signifying that the function should return TRUE if any of the elements of short are contained in long. The other option is "all".
partial	Should partial string matching be allowed? Defaults to FALSE. Partial string matching means that the character string <b>starts with</b> the supplied value.

### Details

This is a helper function to find out if a gene symbol or some similar character string (or character vector) is contained in a pathway. Currently, this function uses base R, but we can write it in a compiled language (such as C++) to increase speed later.

For partial matching (`partial = TRUE`), `long` must be an atomic vector of type character, `short` must be an atomic scalar (a vector with length of 1) of type character, and `matches` should be set to "any". Because this function is designed to match gene symbols or CpG locations, we care if the symbol or location starts with the string supplied. For example, if we set `short = "PIK"`, then we want to find if any of the gene symbols in the supplied long vector belong to the PIK gene family. We don't care if this string appears elsewhere in a gene symbol.

### Value

A logical scalar. If `matches = "any"`, this indicates if any of the elements of `short` are contained in `long`. If `matches = "all"`, this indicates if all of the elements of `short` are contained in `long`. If `partial = TRUE`, the returned logical indicates whether or not any of the character strings in `long` **start with** the character scalar supplied to `short`.

### Examples

```
Contains(1:10, 8)
Contains(LETTERS, c("A", "!"), matches = "any")
Contains(LETTERS, c("A", "!"), matches = "all")
```

```
genesPI <- c(
  "PI4K2A", "PI4K2B", "PI4KA", "PI4KB", "PIK3C2A", "PIK3C2B", "PIK3C2G",
  "PIK3C3", "PIK3CA", "PIK3CB", "PIK3CD", "PIK3CG", "PIK3R1", "PIK3R2",
  "PIK3R3", "PIK3R4", "PIK3R5", "PIK3R6", "PIKFYVE", "PIP4K2A",
  "PIP4K2B", "PIP5K1B", "PIP5K1C", "PITPNB"
)
Contains(genesPI, "PIK3", partial = TRUE)
```

---

 CreateOmics

*Generation Wrapper function for -Omics\*-class objects*


---

## Description

This function calls the [CreateOmicsPath](#), [CreateOmicsSurv](#), [CreateOmicsReg](#), and [CreateOmicsCateg](#) functions to create valid objects of the classes `OmicsPathway`, `OmicsSurv`, `OmicsReg`, or `OmicsCateg`, respectively.

## Usage

```
CreateOmics(
  assayData_df,
  pathwayCollection_ls,
  response = NULL,
  respType = c("none", "survival", "regression", "categorical"),
  centerScale = c(TRUE, TRUE),
  minPathSize = 3,
  ...
)
```

## Arguments

`assayData_df` An  $N \times p$  data frame with named columns.

`pathwayCollection_ls`

A `pathwayCollection` list of known gene pathways with two or three elements:

- `pathways`: A named list of character vectors. Each vector contains the names of the individual genes within that pathway as a vector of character strings. The names contained in these vectors must have non-empty overlap with the *column names* of the `assayData_df` data frame. The names of the pathways (the list elements themselves) should be the a shorthand representation of the full pathway name.
- `TERMS`: A character vector the same length as the `pathways` list with the proper names of the pathways.
- `description`: An optional character vector the same length as the `pathways` list with additional information about the pathways.

If your gene pathways list is stored in a `.gmt` file, use the [read\\_gmt](#) function to import your pathways list as a `pathwayCollection` list object.

response	An optional response object. See "Details" for more information. Defaults to NULL.
respType	What type of response has been supplied. Options are "none", "survival", "regression", and "categorical". Defaults to "none" to match the default response = NULL value.
centerScale	Should the values in assayData_df be centered and scaled? Defaults to TRUE for centering and scaling, respectively. See <a href="#">scale</a> for more information.
minPathSize	What is the smallest number of genes allowed in each pathway? Defaults to 3.
...	Dots for additional arguments passed to the internal <a href="#">CheckAssay</a> function.

### Details

This function is a wrapper around the four CreateOmics\* functions. The values supplied to the response function argument can be in a list, data frame, matrix, vector, [Surv](#) object, or any class which extends these. Because this function makes "best guess" type conversions based on the respType argument, this argument is mandatory if response is non-NULL. Further, it is the responsibility of the user to ensure that the coerced response contained in the resulting Omics object accurately reflects the supplied response.

For respType = "survival", response is assumed to be ordered by event time, then event indicator. For example, if the response is a data frame or matrix, this function assumes that the first column is the time and the second column the death indicator. If the response is a list, then this function assumes that the first entry in the list is the event time and the second entry the death indicator. The death indicator must be a logical or binary (0-1) vector, where 1 or TRUE represents a death and 0 or FALSE represents right-censoring.

Some of the pathways in the supplied pathways list will be removed, or "trimmed", during object creation. For the pathway-testing methods, these trimmed pathways will have *p*-values given as NA. For an explanation of pathway trimming, see the documentation for the [IntersectOmicsPwyCollect](#) function.

### Value

A valid object of class OmicsPathway, OmicsSurv, OmicsReg, or OmicsCateg.

### See Also

[OmicsPathway](#), [CreateOmicsPath](#), [OmicsSurv](#), [CreateOmicsSurv](#), [OmicsCateg](#), [CreateOmicsCateg](#), [OmicsReg](#), [CreateOmicsReg](#), [CheckAssay](#), [CheckPwyColl](#), and [IntersectOmicsPwyCollect](#)

### Examples

```
### Load the Example Data ###
data("colonSurv_df")
data("colon_pathwayCollection")

### Create an OmicsPathway Object ###
colon_OmicsPath <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection
```

```

)

### Create an OmicsSurv Object ###
colon_OmicsSurv <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:3],
  respType = "surv"
)

### Create an OmicsReg Object ###
colon_OmicsReg <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:2],
  respType = "reg"
)

### Create an OmicsCateg Object ###
colon_OmicsCateg <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, c(1,3)],
  respType = "cat"
)

```

---

CreateOmicsPath

*Generation functions for -Omics\*-class objects*


---

## Description

These functions create valid objects of class `OmicsPathway`, `OmicsSurv`, `OmicsReg`, or `OmicsCateg`.

## Usage

```
CreateOmicsPath(assayData_df, sampleIDs_char, pathwayCollection_ls)
```

```
CreateOmicsSurv(
  assayData_df,
  sampleIDs_char,
  pathwayCollection_ls,
  eventTime_num,
  eventObserved_lgl
)
```

```
CreateOmicsReg(
  assayData_df,
  sampleIDs_char,
```

```

    pathwayCollection_ls,
    response_num
)

CreateOmicsCateg(
  assayData_df,
  sampleIDs_char,
  pathwayCollection_ls,
  response_fact
)

```

## Arguments

`assayData_df` An  $N \times p$  data frame with named columns.

`sampleIDs_char` A character vector with the  $N$  sample names.

`pathwayCollection_ls`  
 A `pathwayCollection` list of known gene pathways with two or three elements:
 

- `pathways` : A named list of character vectors. Each vector contains the names of the individual genes within that pathway as a vector of character strings. The names contained in these vectors must have non-empty overlap with the *column names* of the `assayData_df` data frame. The names of the pathways (the list elements themselves) should be the a shorthand representation of the full pathway name.
- `TERMS`: A character vector the same length as the `pathways` list with the proper names of the pathways.
- `description` : An optional character vector the same length as the `pathways` list with additional information about the pathways.

`eventTime_num` A numeric vector with  $N$  observations corresponding to the last observed time of follow up.

`eventObserved_lgl`  
 A logical vector with  $N$  observations indicating right-censoring. The values will be `FALSE` if the observation was censored (i.e., we did not observe an event).

`response_num` A numeric vector of length  $N$ : the dependent variable in an ordinary regression exercise.

`response_fact` A factor vector of length  $N$ : the dependent variable of a generalized linear regression exercise.

## Details

Please note that the classes of the parameters are *not* flexible. The -Omics assay data *must* be or extend the class `data.frame`, and the response values (for a survival-, regression-, or categorical-response object) *must* match their expected classes *exactly*. The reason for this is to encourage the end user to pay attention to the quality and format of their input data. Because the functions internal to this package have only been tested on the classes described in the Arguments section, these class checks prevent unexpected errors (or worse, incorrect computational results without an error). These draconian input class restrictions protect the accuracy of your data analysis.

**Value**

A valid object of class `OmicsPathway`, `OmicsSurv`, `OmicsReg`, or `OmicsCateg`.

**OmicsPathway**

Valid `OmicsPathway` objects will have no response information, just the mass spectrometry or bio-assay ("design") matrix and the pathway list. `OmicsPathway` objects should be created only when unsupervised pathway extraction is needed (not possible with Supervised PCA). Because of the missing response, no pathway testing can be performed on an `OmicsPathway` object.

**OmicsSurv**

Valid `OmicsSurv` objects will have two response vectors: a vector of the most recently recorded follow-up times and a logical vector if that time marks an event (TRUE: observed event; FALSE: right- censored observation).

**OmicsReg and OmicsCateg**

Valid `OmicsReg` and `OmicsCateg` objects will have one response vector of continuous (numeric) or categorical (factor) observations, respectively.

**See Also**

[OmicsPathway](#), [OmicsSurv](#), [OmicsReg](#), and [OmicsCateg](#)

**Examples**

```
# DO NOT CALL THESE FUNCTIONS DIRECTLY. USE CreateOmics() INSTEAD.
```

```
data("colon_pathwayCollection")
data("colonSurv_df")
```

```
## Not run:
```

```
CreateOmicsPath(
  assayData_df = colonSurv_df[, -(1:3)],
  sampleIDs_char = colonSurv_df$sampleID,
  pathwayCollection_ls = colon_pathwayCollection
)
```

```
CreateOmicsSurv(
  assayData_df = colonSurv_df[, -(1:3)],
  sampleIDs_char = colonSurv_df$sampleID,
  pathwayCollection_ls = colon_pathwayCollection,
  eventTime_num = colonSurv_df$OS_time,
  eventObserved_lgl = as.logical(colonSurv_df$OS_event)
)
```

```
CreateOmicsReg(
  assayData_df = colonSurv_df[, -(1:3)],
  sampleIDs_char = colonSurv_df$sampleID,
  pathwayCollection_ls = colon_pathwayCollection,
```

```

    response_num = colonSurv_df$OS_time
  )

  CreateOmicsCateg(
    assayData_df = colonSurv_df[, -(1:3)],
    sampleIDs_char = colonSurv_df$sampleID,
    pathwayCollection_ls = colon_pathwayCollection,
    response_fact = as.factor(colonSurv_df$OS_event)
  )

## End(Not run)

```

---

## CreatePathwayCollection

*Manually Create a pathwayCollection-class Object.*

---

### Description

Manually create a pathwayCollection list similar to the output of the [read\\_gmt](#) function.

### Usage

```

CreatePathwayCollection(
  sets_ls,
  TERMS,
  setType = c("pathways", "genes", "regions"),
  ...
)

```

### Arguments

sets_ls	A named list of character vectors. Each vector should contain the names of the individual genes, proteins, sits, or CpGs within that set as a vector of character strings. If you create this pathway collection to integrate with data of class Omics*, the names contained in these vectors should have non-empty overlap with the feature names of the assay data frame that will be paired with this list in the subsequent analysis.
TERMS	A character vector the same length as the sets_ls list with the proper names of the sets.
setType	What is the type of the set: pathway set of gene, gene sites in RNA or DNA, or regions of CpGs. Defaults to ' 'pathway' '.
...	Additional vectors or data components related to the sets_ls list. These values should be passed as a name-value pair. See "Details" for more information.

**Details**

This function checks the set list and set term inputs and then creates a pathwayCollection object from them. Pass additional list elements (such as the description of each set) using the form tag = value through the ... argument (as in the [list](#) function). Because some functions in the pathwayPCA package add and edit elements of pathwayCollection objects, please do not create pathwayCollection list items named setsize or n\_tested.

**Value**

A list object with class pathwayCollection.

**See Also**

[read\\_gmt](#)

**Examples**

```
data("colon_pathwayCollection")

CreatePathwayCollection(
  sets_ls = colon_pathwayCollection$pathways,
  TERMS = colon_pathwayCollection$TERMS
)
```

---

getPathPCLs	<i>Extract PCs and Loadings from a superpcOut- or aespcOut-class Object.</i>
-------------	--

---

**Description**

Given an object of class aespcOut or superpcOut, as returned by the functions [AESPCA\\_pVals](#) or [SuperPCA\\_pVals](#), respectively, and the name or unique ID of a pathway, return a data frame of the principal components and a data frame of the loading vectors corresponding to that pathway.

**Usage**

```
getPathPCLs(pcOut, pathway_char, ...)

## S3 method for class 'superpcOut'
getPathPCLs(pcOut, pathway_char, ...)

## S3 method for class 'aespcOut'
getPathPCLs(pcOut, pathway_char, ...)
```



**Arguments**

pcOut	An object of classes superpcOut or aespcOut as returned by the <a href="#">SuperPCA_pVals</a> or <a href="#">AESPCA_pVals</a> functions, respectively.
pathway_char	A character string of the name or unique identifier of a pathway
...	Dots for additional arguments (currently unused).

**Details**

Match the supplied pathway character string to either the pathways or terms columns of the pVals\_df data frame within the pcOut object. Then, subset the loadings\_ls and PCs\_ls lists for their entries which match the supplied pathway. Finally, return a list of the PCs, loadings, and the pathway ID and name.

**Value**

A list of four elements:

- PCs : A data frame of the principal components
- Loadings : A matrix of the loading vectors with features in the row names
- pathway : The unique pathway identifier for the pcOut object
- term : The name of the pathway

NULL

NULL

**Examples**

```
### Load Data ###
data("colonSurv_df")
data("colon_pathwayCollection")

### Create -Omics Container ###
colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:3],
  respType = "survival"
)

### Calculate Supervised PCA Pathway p-Values ###
colon_superpc <- SuperPCA_pVals(
  colon_Omics,
  numPCs = 2,
  parallel = TRUE,
  numCores = 2,
  adjustment = "BH"
)

### Extract PCs and Loadings ###
```

```
getPathPCLs(
  colon_superpc,
  "KEGG_PENTOSE_PHOSPHATE_PATHWAY"
)
```

---

getPathVals	<i>Extract Table of p-values from a superpcOut- or aespcoOut- class Object.</i>
-------------	---

---

### Description

Given an object of class `aespcOut` or `superpcOut`, as returned by the functions [AESPCA\\_pVals](#) or [SuperPCA\\_pVals](#), respectively, return a data frame of the  $p$ -values for the top pathways.

### Usage

```
getPathVals(pcOut, score = FALSE, numPaths = 20L, alpha = NULL, ...)

## S3 method for class 'superpcOut'
getPathVals(pcOut, score = FALSE, numPaths = 20L, alpha = NULL, ...)

## S3 method for class 'aespcOut'
getPathVals(pcOut, score = FALSE, numPaths = 20L, alpha = NULL, ...)
```

### Arguments

<code>pcOut</code>	An object of classes <code>superpcOut</code> or <code>aespcOut</code> as returned by the <a href="#">SuperPCA_pVals</a> or <a href="#">AESPCA_pVals</a> functions, respectively.
<code>score</code>	Should the unadjusted $p$ -values be returned transformed to negative natural logarithm scores or left as is? Defaults to <code>FALSE</code> ; that is, the raw $p$ -values are returned instead of the transformed $p$ -values.
<code>numPaths</code>	The number of top pathways by raw $p$ -value. Defaults to the top 20 pathways. We do not permit users to specify <code>numPaths</code> and <code>alpha</code> concurrently.
<code>alpha</code>	The significance threshold for raw $p$ -values. Defaults to <code>NULL</code> . If <code>alpha</code> is given, then <code>numPaths</code> will be ignored.
<code>...</code>	Dots for additional arguments (currently unused).

### Details

Row-subset the `pVals_df` entry of an object of class `aespcOut` or `superpcOut` by the number of pathways requested (via the `nPaths` argument) or by the unadjusted significance level for each pathway (via the `alpha` argument). Return a data frame of the pathway names, FDR-adjusted significance levels (if available), and the raw score (negative natural logarithm of the  $p$ -values) of each pathway.

**Value**

A data frame with the following columns:

- `terms` : The pathway name, as given in the `object@trimPathwayCollection$TERMS` object.
- `description` : (OPTIONAL) The pathway description, as given in the `object@trimPathwayCollection$description` object, if supplied.
- `rawp` : The unadjusted  $p$ -values of each pathway. Included if `score = FALSE`.
- `...` : Additional columns of FDR-adjusted  $p$ -values as specified through the adjustment argument of the [SuperPCA\\_pVals](#) or [AESPCA\\_pVals](#) functions.
- `score` : The negative natural logarithm of the unadjusted  $p$ -values of each pathway. Included if `score = TRUE`.

NULL

NULL

**Examples**

```
### Load Data ###
data("colonSurv_df")
data("colon_pathwayCollection")

### Create -Omics Container ###
colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:3],
  respType = "survival"
)

### Calculate Supervised PCA Pathway p-Values ###
colon_superpc <- SuperPCA_pVals(
  colon_Omics,
  numPCs = 2,
  parallel = TRUE,
  numCores = 2,
  adjustment = "BH"
)

### Extract Table of p-Values ###
# Top 5 Pathways
getPathpVals(
  colon_superpc,
  numPaths = 5
)

# Pathways with Unadjusted p-Values < 0.01
getPathpVals(
  colon_superpc,
  alpha = 0.01
)
```

---

 LoadOntoPCs

*Calculate Test Data PCs from Training-Data Estimated Loadings*


---

### Description

Given a list of loading vectors from a training data set, calculate the PCs of the test data set.

### Usage

```
LoadOntoPCs(design_df, loadings_ls, sampleID = c("firstCol", "rowNames"))
```

### Arguments

design_df	A test data frame with rows as samples and named features as columns
loadings_ls	A list of $p \times d$ loading vectors or matrices as returned by either the <a href="#">SuperPCA_pVals</a> , <a href="#">AESPCA_pVals</a> , or <a href="#">ExtractAESPCs</a> functions. These lists of loadings will have feature names as their row names. Such feature names must match a subset of the column names of design_df exactly, as pathway-specific test-data subsetting is performed by column name.
sampleID	Are the sample IDs in the first column of design_df or in accessible by rownames(design_df)? Defaults to the first column. If your data does not have sample IDs for some reason, set this to rowNames.

### Details

This function takes in a list of loadings and a training-centered test data set, applies over the list of loadings, subsets the columns of the test data by the row names of the loading vectors, right-multiplies the test-data subset matrix by the loading vector / matrix, and returns a data frame of the test-data PCs for each loading vector.

### Value

A data frame with the PCs from each pathway concatenated by column. If you have the tidyverse loaded, this object will display as a [tibble](#).

### Examples

```
### Load the Data ###
data("colonSurv_df")
data("colon_pathwayCollection")

### Create -Omics Container ###
colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
```

```

    response = colonSurv_df[, 1:3],
    respType = "survival"
  )

  ### Extract AESPCs ###
  colonSurv_aespc <- AESPCA_pVals(
    object = colon_Omics,
    numReps = 0,
    parallel = TRUE,
    numCores = 2,
    adjustpValues = TRUE,
    adjustment = c("Hoch", "SidakSD")
  )

  ### Project Data onto Pathway First PCs ###
  LoadOntoPCs(
    design_df = colonSurv_df,
    loadings_ls = colonSurv_aespc$loadings_ls
  )

```

---

 OmicsCateg-class

 An S4 class for categorical responses within an OmicsPathway object
 

---

## Description

This creates the OmicsCateg class which extends the OmicsPathway master class.

## Slots

`assayData_df` An  $N \times p$  data frame with named columns.

`pathwayCollection` A list of known gene pathways with three or four elements:

- `pathways` : A named list of character vectors. Each vector contains the names of the individual genes within that pathway as a vector of character strings. The names contained in these vectors must have non-empty overlap with the *column names* of the `assayData_df` data frame. The names of the pathways (the list elements themselves) should be the a shorthand representation of the full pathway name.
- `TERMS` : A character vector the same length as the pathways list with the proper names of the pathways.
- `description` : An optional character vector the same length as the pathways list with additional information about the pathways.
- `setsize` : A named integer vector the same length as the pathways list with the number of genes in each pathway. This list item is calculated during the creation step of a CreateOmics function call.

`response` A factor vector of length  $N$ : the dependent variable of a generalized linear regression exercise. Currently, we support binary factors only. We expect to extend support to n-ary responses in the next package version.

**See Also**

[OmicsPathway](#), [CreateOmics](#)

---

OmicsPathway-class	<i>An S4 class for mass spectrometry or bio-assay data and gene pathway lists</i>
--------------------	---

---

**Description**

An S4 class for mass spectrometry or bio-assay data and gene pathway lists

**Slots**

`assayData_df` An  $N \times p$  data frame with named columns.

`sampleIDs_char` A character vector with the N sample names.

`pathwayCollection` A list of known gene pathways with three or four elements:

- `pathways` : A named list of character vectors. Each vector contains the names of the individual genes within that pathway as a vector of character strings. The names contained in these vectors must have non-empty overlap with the *column names* of the `assayData_df` data frame. The names of the pathways (the list elements themselves) should be the a shorthand representation of the full pathway name.
- `TERMS` : A character vector the same length as the pathways list with the proper names of the pathways.
- `description` : An optional character vector the same length as the pathways list with additional information about the pathways.
- `setsize` : A named integer vector the same length as the pathways list with the number of genes in each pathway. This list item is calculated during the creation step of a `CreateOmics` function call.

`trimPathwayCollection` A subset of the list stored in the `pathwayCollection` slot. This list will have pathways that only contain genes that are present in the assay data frame.

**See Also**

[CreateOmics](#)

---

OmicsReg-class	<i>An S4 class for continuous responses within an OmicsPathway object</i>
----------------	---

---

**Description**

This creates the OmicsReg class which extends the OmicsPathway master class.

**Slots**

assayData\_df An  $N \times p$  data frame with named columns.

pathwayCollection A list of known gene pathways with three or four elements:

- pathways : A named list of character vectors. Each vector contains the names of the individual genes within that pathway as a vector of character strings. The names contained in these vectors must have non-empty overlap with the *column names* of the assayData\_df data frame. The names of the pathways (the list elements themselves) should be the shorthand representation of the full pathway name.
- TERMS : A character vector the same length as the pathways list with the proper names of the pathways.
- description : An optional character vector the same length as the pathways list with additional information about the pathways.
- setsize : A named integer vector the same length as the pathways list with the number of genes in each pathway. This list item is calculated during the creation step of a CreateOmics function call.

response A numeric vector of length  $N$ : the dependent variable in a regression exercise.

**See Also**

[OmicsPathway](#), [CreateOmics](#)

---

OmicsSurv-class	<i>An S4 class for survival responses within an OmicsPathway object</i>
-----------------	---

---

**Description**

This creates the OmicsSurv class which extends the OmicsPathway master class.

**Slots**

assayData\_df An  $N \times p$  data frame with named columns.

pathwayCollection A list of known gene pathways with three or four elements:

- pathways : A named list of character vectors. Each vector contains the names of the individual genes within that pathway as a vector of character strings. The names contained in these vectors must have non-empty overlap with the *column names* of the assayData\_df data frame. The names of the pathways (the list elements themselves) should be the shorthand representation of the full pathway name.

- **TERMS** : A character vector the same length as the pathways list with the proper names of the pathways.
- **description** : An optional character vector the same length as the pathways list with additional information about the pathways.
- **setsize** : A named integer vector the same length as the pathways list with the number of genes in each pathway. This list item is calculated during the creation step of a CreateOmics function call.

**eventTime** A numeric vector with  $N$  observations corresponding to the last observed time of follow up.

**eventObserved** A logical vector with  $N$  observations indicating right-censoring. The values will be FALSE if the observation was censored (i.e., we did not observe an event).

### See Also

[OmicsPathway](#), [CreateOmics](#)

---

pathwayPCA

*Extract and Test the Significance of Pathway-Specific Principal Components*

---

### Description

To introduce this package, please see our "Integrative Pathway Analysis" vignette: [https://gabrielodom.github.io/pathwayPCA/articles//Introduction\\_to\\_pathwayPCA.html](https://gabrielodom.github.io/pathwayPCA/articles//Introduction_to_pathwayPCA.html).

The pathwayPCA package has three main components:

- **Import and Tidy Data**: [https://gabrielodom.github.io/pathwayPCA/articles/Supplement2-Importing\\_Data.html](https://gabrielodom.github.io/pathwayPCA/articles/Supplement2-Importing_Data.html)
- **Create Omics Data Objects**: [https://gabrielodom.github.io/pathwayPCA/articles/Supplement3-Create\\_Omics\\_Objects.html](https://gabrielodom.github.io/pathwayPCA/articles/Supplement3-Create_Omics_Objects.html)
- **Test Pathway Significance**: [https://gabrielodom.github.io/pathwayPCA/articles/Supplement4-Methods\\_Walkthrough.html](https://gabrielodom.github.io/pathwayPCA/articles/Supplement4-Methods_Walkthrough.html)
- **Analyze and Visualize Results**: [https://gabrielodom.github.io/pathwayPCA/articles/Supplement5-Analyze\\_Results.html](https://gabrielodom.github.io/pathwayPCA/articles/Supplement5-Analyze_Results.html)

For an overview of these four topics in context, please see our Quickstart Guide: [https://gabrielodom.github.io/pathwayPCA/articles/Supplement1-Quickstart\\_Guide.html](https://gabrielodom.github.io/pathwayPCA/articles/Supplement1-Quickstart_Guide.html)



---

read_gmt	<i>Read a .gmt file in as a pathwayCollection object</i>
----------	--

---

### Description

Read a set list file in Gene Matrix Transposed (.gmt) format, with special performance consideration for large files. Present this object as a pathwayCollection object.

### Usage

```
read_gmt(
  file,
  setType = c("pathways", "genes", "regions"),
  description = FALSE,
  nChars = 1e+07,
  delim = "\t"
)
```

### Arguments

file	A path to a file or a connection. This file must be a .gmt file, otherwise input will likely be nonsense. See the "Details" section for more information.
setType	What is the type of the set: pathway set of gene, gene sites in RNA or DNA, or regions of CpGs. Defaults to ' 'pathway' '.
description	Should the "description" field (the second field in the .gmt file on each line) be included in the output? Defaults to FALSE.
nChars	The number of characters to read from a connection. The largest .gmt file we have encountered is the full C5 pathway collection from MSigDB (5917 pathways), which has roughly 5 million characters in UTF-8 encoding. Therefore, we default this argument to be twice the size of the largest pathway collection we have seen so far, 10,000,000.
delim	The .gmt delimiter. As proper .gmt files are tab delimited, this defaults to "\t".

### Details

This function uses R's [readChar](#) function to improve character input performance over [readLines](#) (and far improve input performance over [scan](#)).

See the Broad Institute's "Data Formats" page for a description of the Gene Matrix Transposed file format: [https://software.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data\\_formats#GMT:\\_Gene\\_Matrix\\_Transposed\\_file\\_format\\_.28.2A.gmt.29](https://software.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data_formats#GMT:_Gene_Matrix_Transposed_file_format_.28.2A.gmt.29)

### Value

A pathwayCollection list of sets. This list has three elements:

- `'setType'` : A named list of character vectors. Each vector contains the names of the individual genes, sites, or CpGs within that set as a vector of character strings. The name of this list entry is equal to the value specified in `setType`.
- `TERMS` : A character vector the same length as the `'setType'` list with the proper names of the sets.
- `description` : (OPTIONAL) A character vector the same length as the `'setType'` list with a note on that set (for the `.gmt` file included with this package, this field contains hyperlinks to the MSigDB description card for that pathway). This field is included when `description = TRUE`.

### See Also

[print.pathwayCollection](#); [write\\_gmt](#)

### Examples

```
# If you have installed the package:
data_path <- system.file(
  "extdata", "c2.cp.v6.0.symbols.gmt",
  package = "pathwayPCA", mustWork = TRUE
)
geneset_ls <- read_gmt(data_path, description = TRUE)

## If you are using the development version from GitHub:
# geneset_ls <- read_gmt(
#   "inst/extdata/c2.cp.v6.0.symbols.gmt",
#   description = TRUE
# )
```

---

SE2Tidy

*Tidy a SummarizedExperiment Assay*

---

### Description

Extract the assay information from a [SummarizedExperiment-class](#)-object, transpose it, and return it as a tidy data frame that contains assay measurements, feature names, and sample IDs

### Usage

```
SE2Tidy(summExperiment, whichAssay = 1)
```

### Arguments

`summExperiment` A [SummarizedExperiment-class](#) object

`whichAssay` Because `SummarizedExperiment` objects can store multiple related assays, which assay will be paired with a given pathway collection to create an `Omics*`-class data container? Defaults to 1, for the first assay in the object.

## Details

This function is designed to extract and transpose a "tall" assay data frames (where genes or proteins are the rows and patient or tumour samples are the columns) from a `SummarizedExperiment` object. This function also transposes the row (feature) names to column names and the column (sample) names to row names via the [TransposeAssay](#) function.

## Value

The transposition of the assay in `summExperiment` to tidy form, with the column data (from the `colData` slot of the object) appended as the first columns of the data frame.

## Examples

```
# THIS REQUIRES THE SummarizedExperiment PACKAGE.
library(SummarizedExperiment)
data(airway, package = "airway")

airway_df <- SE2Tidy(airway)
```

---

SubsetOmicsPath	<i>Access and Edit Assay or pathwayCollection Values in Omics* Objects</i>
-----------------	--

---

## Description

"Get" or "Set" the values of the `assayData_df`, `sampleIDs_char`, or `pathwayCollection` slots of an object of class `OmicsPathway` or a class that extends this class (`OmicsSurv`, `OmicsReg`, or `OmicsCateg`).

## Usage

```
getAssay(object, ...)

getAssay(object) <- value

getSampleIDs(object, ...)

getSampleIDs(object) <- value

getPathwayCollection(object, ...)

getPathwayCollection(object) <- value

getTrimPathwayCollection(object, ...)

## S4 method for signature 'OmicsPathway'
```

```

getAssay(object, ...)

## S4 replacement method for signature 'OmicsPathway'
getAssay(object) <- value

## S4 method for signature 'OmicsPathway'
getSampleIDs(object, ...)

## S4 replacement method for signature 'OmicsPathway'
getSampleIDs(object) <- value

## S4 method for signature 'OmicsPathway'
getPathwayCollection(object, ...)

## S4 replacement method for signature 'OmicsPathway'
getPathwayCollection(object) <- value

## S4 method for signature 'OmicsPathway'
getTrimPathwayCollection(object, ...)

```

### Arguments

object	An object of or extending <a href="#">OmicsPathway-class</a> : that class, <a href="#">OmicsSurv-class</a> , <a href="#">OmicsReg-class</a> , or <a href="#">OmicsCateg-class</a> .
...	Dots for additional internal arguments (currently unused).
value	The replacement object to be assigned to the specified slot.

### Details

These functions can be useful to set or extract the assay data or pathways list from an `Omics*`-class object. However, we recommend that users simply create a new, valid `Omics*` object instead of modifying an existing one. The validity of edited objects is checked with the [ValidOmicsSurv](#), [ValidOmicsCateg](#), or [ValidOmicsReg](#) functions.

Further, because the `pathwayPCA` methods require a cleaned (trimmed) pathway collection, the `trimPathwayCollection` slot is read-only. Users may only edit this slot by updating the pathway collection provided to the `pathwayCollection` slot. Despite this functionality, we **strongly** recommend that users create a new object with the updated pathway collection, rather than attempting to overwrite the slots within an existing object. See [IntersectOmicsPwyCollct](#) for details on trimmed pathway collection.

### Value

The "get" functions return the objects in the slots specified: `getAssay` returns the `assayData_df` data frame object, `getSampleIDs` returns the `sampleIDs_char` character vector, `getPathwayCollection` returns the `pathwayCollection` list object, and `getTrimPathwayCollection` returns the `trimPathwayCollection`. These functions can extract these values from any valid `OmicsPathway`, `OmicsSurv`, `OmicsReg`, or `OmicsCateg` object.

The "set" functions enable the user to edit or replace objects in the assayData\_df, sampleIDs\_char, or pathwayCollection slots for any OmicsPathway, OmicsSurv, OmicsReg, or OmicsCateg objects, provided that the new values do not violate the validity checks of their respective objects. Because the slot for trimPathwayCollection is filled upon object creation, and to ensure that this pathway collection is "clean", there is no "set" function for the trimmed pathway collection slot. Instead, users can update the pathway collection, and the trimmed pathway collection will be updated automatically. See "Details" for more information on the "set" functions.

## See Also

[CreateOmics](#)

## Examples

```
data("colonSurv_df")
data("colon_pathwayCollection")

colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection
)

getAssay(colon_Omics)
getPathwayCollection(colon_Omics)
```

---

SubsetOmicsResponse    *Access and Edit Response of an OmicsReg or OmicsReg Object*

---

## Description

"Get" or "Set" the values of the response\_num or response\_fact slots of an object of class OmicsReg or OmicsReg, respectively.

## Usage

```
getResponse(object, ...)

getResponse(object) <- value

## S4 method for signature 'OmicsPathway'
getResponse(object, ...)

## S4 replacement method for signature 'OmicsPathway'
getResponse(object) <- value
```

## Arguments

object	An object of class <a href="#">OmicsReg-class</a> or <a href="#">OmicsCateg-class</a> .
...	Dots for additional internal arguments (currently unused).
value	The replacement object to be assigned to the response slot.

## Details

These functions can be useful to set or extract the response vector from an object of class [OmicsReg](#) or [OmicsReg](#). However, we recommend that users simply create a new, valid object instead of modifying an existing one. The validity of edited objects is checked with their respective [ValidOmicsCateg](#) or [ValidOmicsReg](#) function. Because both classes have a response slot, we set this method for the parent class, [OmicsPathway-class](#).

## Value

The "get" functions return the objects in the slots specified: `getResponse` returns the `response_num` vector from objects of class [OmicsReg](#) and the `response_fact` vector from objects of class [OmicsCateg](#). These functions can extract these values from any valid object of those classes.

The "set" functions enable the user to edit or replace the object in the `response_num` slot for any [OmicsReg](#) object or `response_fact` slot for any [OmicsCateg](#) object, provided that the new values do not violate the validity check of such an object. See "Details" for more information.

## See Also

[CreateOmics](#)

## Examples

```
data("colonSurv_df")
data("colon_pathwayCollection")

colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, c(1, 2)],
  respType = "reg"
)

getResponse(colon_Omics)
```

---

SubsetOmicsSurv	<i>Access and Edit Event Time or Indicator in an OmicsSurv Object</i>
-----------------	---

---

### Description

"Get" or "Set" the values of the eventTime\_num or eventObserved\_lgl slots of an object of class OmicsSurv.

### Usage

```
getEventTime(object, ...)  
  
getEventTime(object) <- value  
  
getEvent(object, ...)  
  
getEvent(object) <- value  
  
## S4 method for signature 'OmicsSurv'  
getEventTime(object, ...)  
  
## S4 replacement method for signature 'OmicsSurv'  
getEventTime(object) <- value  
  
## S4 method for signature 'OmicsSurv'  
getEvent(object, ...)  
  
## S4 replacement method for signature 'OmicsSurv'  
getEvent(object) <- value
```

### Arguments

object	An object of class <a href="#">OmicsSurv-class</a> .
...	Dots for additional internal arguments (currently unused).
value	The replacement object to be assigned to the specified slot.

### Details

These functions can be useful to set or extract the event time or death indicator from an OmicsSurv object. However, we recommend that users simply create a new, valid OmicsSurv object instead of modifying an existing one. The validity of edited objects is checked with the [ValidOmicsSurv](#) function.

**Value**

The "get" functions return the objects in the slots specified: `getEventTime` returns the `eventTime_num` vector object and `getEvent` returns the `eventObserved_lgl` vector object. These functions can extract these values from any valid `OmicsSurv` object.

The "set" functions enable the user to edit or replace objects in the `eventTime_num` or `eventObserved_lgl` slots for any `OmicsSurv` object, provided that the new values do not violate the validity check of an `OmicsSurv` object. See "Details" for more information.

**See Also**

[CreateOmics](#)

**Examples**

```
data("colonSurv_df")
data("colon_pathwayCollection")

colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:3],
  respType = "survival"
)

getEventTime(colon_Omics)
getEvent(colon_Omics)
```

---

SubsetPathwayCollection

*Subset a pathwayCollection-class Object by Pathway.*

---

**Description**

The subset method for pathways lists as returned by the [read\\_gmt](#) function.

**Usage**

```
## S3 method for class 'pathwayCollection'
x[[name_char]]
```

**Arguments**

`x` An object of class `pathwayCollection`.

`name_char` The name of a pathway in the collection or its unique ID.



**Details**

This function finds the index matching the `name_char` argument to the `TERMS` field of the `pathwayCollection`-class Object, then subsets the `pathways` list, `TERMS` vector, `description` vector, and `setsize` vector by this index. If you subset a trimmed `pathwayCollection` object, and the function errors with "Pathway not found.", then the pathway specified has been trimmed from the pathway collection.

Also, this function does not allow for users to overwrite any portion of a pathway collection. These objects should rarely, if ever, be changed. If you absolutely must change the components of a `pathwayCollection` object, then create a new one with the code `CreatePathwayCollection` function.

**Value**

A list of the pathway name (Term), unique ID (pathID), contents (IDs), description (description), and number of features (Size).

**Examples**

```
data("colon_pathwayCollection")
colon_pathwayCollection[["KEGG_RETINOL_METABOLISM"]]
```

---

SubsetPathwayData      *Subset Pathway-Specific Data*

---

**Description**

Given an Omics object and the name of a pathway, return the -omes in the assay and the response as a (tibble) data frame.

**Usage**

```
SubsetPathwayData(object, pathName, ...)

## S4 method for signature 'OmicsPathway'
SubsetPathwayData(object, pathName, ...)
```

**Arguments**

<code>object</code>	An object of class <code>OmicsPathway</code> , or an object extending this class.
<code>pathName</code>	The name of a pathway contained in the pathway collection in the object.
<code>...</code>	Dots for additional internal arguments (currently unused).

**Details**

This function subsets the assay by the matching gene symbols or IDs in the specified pathway.

**Value**

A data frame of the columns of the assay in the Omics object which are listed in the specified pathway, with a leading column for sample IDs. If the Omics object has response information, these are also included as the first column(s) of the data frame, after the sample IDs. If you have the suggested tidyverse package suite loaded, then this data frame will print as a `tibble`. Otherwise, it will print as a data frame.

**Examples**

```
data("colonSurv_df")
data("colon_pathwayCollection")

colon_Omics <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:3],
  respType = "survival"
)

SubsetPathwayData(
  colon_Omics,
  "KEGG_RETINOL_METABOLISM"
)
```

---

 SuperPCA\_pVals

*Test pathways with Supervised PCA*


---

**Description**

Given a supervised OmicsPath object (one of OmicsSurv, OmicsReg, or OmicsCateg), extract the first  $k$  principal components (PCs) from each pathway-subset of the -Omics assay design matrix, test their association with the response matrix, and return a data frame of the adjusted  $p$ -values for each pathway.

**Usage**

```
SuperPCA_pVals(
  object,
  n.threshold = 20,
  numPCs = 1,
  parallel = FALSE,
  numCores = NULL,
  adjustpValues = TRUE,
  adjustment = c("Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BH", "BY",
    "ABH", "TSBH"),
  ...
)
```

```

)

## S4 method for signature 'OmicsPathway'
SuperPCA_pVals(
  object,
  n.threshold = 20,
  numPCs = 1,
  parallel = FALSE,
  numCores = NULL,
  adjustpValues = TRUE,
  adjustment = c("Bonferroni", "Holm", "Hochberg", "SidakSS", "SidakSD", "BH", "BY",
    "ABH", "TSBH"),
  ...
)

```

### Arguments

<code>object</code>	An object of superclass <code>OmicsPathway</code> with a response matrix or vector.
<code>n.threshold</code>	The number of bins into which to split the feature scores in the fit object returned internally by the <code>superpc.train</code> function to the <code>pathway_tScores</code> and <code>pathway_tControl</code> functions. Defaults to 20. Smaller values may result in less accurate pathway $p$ -values while larger values increase computation time.
<code>numPCs</code>	The number of PCs to extract from each pathway. Defaults to 1.
<code>parallel</code>	Should the computation be completed in parallel? Defaults to FALSE.
<code>numCores</code>	If <code>parallel = TRUE</code> , how many cores should be used for computation? Internally defaults to the number of available cores minus 1.
<code>adjustpValues</code>	Should you adjust the $p$ -values for multiple comparisons? Defaults to TRUE.
<code>adjustment</code>	Character vector of procedures. The returned data frame will be sorted in ascending order by the first procedure in this vector, with ties broken by the unadjusted $p$ -value. If only one procedure is selected, then it is necessarily the first procedure. See the documentation for the <code>ControlFDR</code> function for the adjustment procedure definitions and citations.
<code>...</code>	Dots for additional internal arguments.

### Details

This is a wrapper function for the `pathway_tScores`, `pathway_tControl`, `OptimGumbelMixParams`, `GumbelMixpValues`, and `TabulatepValues` functions.

Please see our Quickstart Guide for this package: [https://gabrielodom.github.io/pathwayPCA/articles/Supplement1-Quickstart\\_Guide.html](https://gabrielodom.github.io/pathwayPCA/articles/Supplement1-Quickstart_Guide.html)

### Value

A data frame with columns:

- `pathways` : The names of the pathways in the `Omics*` object (given in `object@trimPathwayCollection$pathways`.)

- `setSize` : The number of genes in each of the original pathways (given in the `object@trimPathwayCollection$setSize` object).
- `terms` : The pathway description, as given in the `object@trimPathwayCollection$TERMS` object.
- `rawp` : The unadjusted  $p$ -values of each pathway.
- `...` : Additional columns as specified through the adjustment argument.

The data frame will be sorted in ascending order by the method specified first in the adjustment argument. If `adjustpValues = FALSE`, then the data frame will be sorted by the raw  $p$ -values. If you have the suggested tidyverse package suite loaded, then this data frame will print as a [tibble](#). Otherwise, it will print as a data frame.

### See Also

[CreateOmics](#); [TabulatepValues](#); [pathway\\_tScores](#); [pathway\\_tControl](#); [OptimGumbelMixParams](#); [GumbelMixpValues](#); [clusterApply](#)

### Examples

```
### Load the Example Data ###
data("colonSurv_df")
data("colon_pathwayCollection")

### Create an OmicsSurv Object ###
colon_OmicsSurv <- CreateOmics(
  assayData_df = colonSurv_df[, -(2:3)],
  pathwayCollection_ls = colon_pathwayCollection,
  response = colonSurv_df[, 1:3],
  respType = "surv"
)

### Calculate Pathway p-Values ###
colonSurv_superpc <- SuperPCA_pVals(
  object = colon_OmicsSurv,
  parallel = TRUE,
  numCores = 2,
  adjustpValues = TRUE,
  adjustment = c("Hoch", "SidakSD")
)
```

---

TransposeAssay

*Transpose an Assay (Data Frame)*

---

### Description

Transpose an object of class `data.frame` that contains assay measurements while preserving row (feature) and column (sample) names.

## Usage

```
TransposeAssay(  
  assay_df,  
  omeNames = c("firstCol", "rowNames"),  
  stringsAsFactors = FALSE  
)
```

## Arguments

<code>assay_df</code>	A data frame with numeric values to transpose
<code>omeNames</code>	Are the data feature names in the first column or in the row names of <code>df</code> ? Defaults to the first column. If the feature names are in the row names, this function assumes that these names are accessible by the <code>rownames</code> function called on <code>df</code> .
<code>stringsAsFactors</code>	Should columns containing string information be coerced to factors? Defaults to <code>FALSE</code> .

## Details

This function is designed to transpose "tall" assay data frames (where genes or proteins are the rows and patient or tumour samples are the columns). This function also transposes the row (feature) names to column names and the column (sample) names to row names. Notice that all rows and columns (other than the feature name column, as applicable) are numeric.

Recall that data frames require that all elements of a single column to have the same `class`. Therefore, sample IDs of a "tall" data frame **must** be stored as the column names rather than in the first row.

## Value

The transposition of `df`, with row and column names preserved and reversed.

## Examples

```
x_mat <- matrix(rnorm(5000), ncol = 20, nrow = 250)  
rownames(x_mat) <- paste0("gene_", 1:250)  
colnames(x_mat) <- paste0("sample_", 1:20)  
x_df <- as.data.frame(x_mat, row.names = rownames(x_mat))
```

```
TransposeAssay(x_df, omeNames = "rowNames")
```

---

**WhichPathways***Filter and Subset a pathwayCollection-class Object by Symbol.*

---

**Description**

The filter-subset method for pathways lists as returned by the [read\\_gmt](#) function. This function returns the subset of pathways which contain the set of symbols requested

**Usage**

```
WhichPathways(x, symbols_char, ...)
```

**Arguments**

x	An object of class pathwayCollection.
symbols_char	A character vector or scalar of gene symbols or regions
...	Additional arguments passed to the <a href="#">Contains</a> function

**Details**

This function finds the index of each set that contains the symbols supplied, then returns those sets as a new pathwayCollection object. Find pathways that contain geneA OR geneB by passing the argument matches = "any" through ... to [Contains](#) (this is the default value). Find pathways that contain geneA AND geneB by changing this argument to matches = "all". Find all genes in a specified family by passing in one value to short and setting partial = TRUE.

**Value**

An object of class pathwayCollection, but containing only the sets which contain the symbols supplied to symbols\_char. If no sets are found to contain the symbols supplied, this function returns NULL and prints a warning.

**Examples**

```
data("colon_pathwayCollection")

WhichPathways(colon_pathwayCollection, "MAP", partial = TRUE)

WhichPathways(
  colon_pathwayCollection,
  c("MAP4K5", "RELA"),
  matches = "all"
)
```

---

wikipwsHS\_Entrez\_pathwayCollection  
*Wikipathways Homosapiens EntrezIDs*

---

**Description**

A pathwayCollection object containing the homosapiens pathways list from Wikipathways (<https://www.wikipathways.org/>).

**Usage**

```
data(wikipwsHS_Entrez_pathwayCollection)
```

**Format**

A pathwayCollection list of three elements:

- pathways : A named list of 443 character vectors. Each vector contains the Entrez Gene IDs of the individual genes within that pathway as a vector of character strings. The names are the shorthand pathway names.
- TERMS : A character vector of length 443 containing the shorthand names of the gene pathways.
- description : A character vector of length 443 containing the full names of the gene pathways.

**Details**

This pathwayCollection was sent to us from Dr. Alexander Pico at the Gladstone Institute (<https://gladstone.org/our-science/people/alexander-pico>).

**Source**

Dr. Alexander Pico, Wikipathways

---

wikipwsHS\_Symbol\_pathwayCollection  
*Wikipathways Homosapiens Gene Symbols*

---

**Description**

A pathwayCollection object containing the homosapiens pathways list from Wikipathways (<https://www.wikipathways.org/>).

**Usage**

```
data(wikipwsHS_Symbol_pathwayCollection)
```

**Format**

A pathwayCollection list of three elements:

- pathways : A named list of 457 character vectors. Each vector contains the Gene Symbols of the individual genes within that pathway as a vector of character strings. The names are the shorthand pathway names.
- TERMS : A character vector of length 457 containing the shorthand names of the gene pathways.
- description : A character vector of length 457 containing the full names of the gene pathways.

**Details**

This pathwayCollection was sent to us from Dr. Alexander Pico at the Gladstone Institute (<https://gladstone.org/our-science/people/alexander-pico>).

This pathway collection was translated from EntrezIDs to HGNC Symbols with the script `convert_EntrezID_to_HGNC_Ensembl` in scripts.

**Source**

Dr. Alexander Pico, Wikipathways

---

write\_gmt

*Write a pathwayCollection Object to a .gmt File*

---

**Description**

Write a pathwayCollection object as a pathways list file in Gene Matrix Transposed (.gmt) format.

**Usage**

```
write_gmt(pathwayCollection, file, setType = c("pathways", "genes", "regions"))
```

**Arguments**

pathwayCollection

A pathwayCollection list of sets. This list contains the following two or three elements:

- 'setType' : A named list of character vectors. Each vector contains the names of the individual genes, sites, or CpGs within that set as a vector of character strings. If you are using genes, these genes can be represented by HGNC gene symbols, Entrez IDs, Ensembl IDs, GO terms, etc.
- TERMS : A character vector the same length as the 'setType' list with the proper names of the sets.



- `description` : An optional character vector the same length as the 'set-Type' list with a note on that set (such as a url to the description if the set is a pathway). If this element of the `pathwayCollection` is `NULL`, then the file will be written with "" (the empty character string) as its second field in each line.
- `file` Either a character string naming a file or a connection open for writing. File names should end in `.gmt` for clarity.
- `setType` What is the type of the set: pathway set of gene, gene sites in RNA or DNA, or regions of CpGs. Defaults to `'pathway'`.

### Details

See the Broad Institute's "Data Formats" page for a description of the Gene Matrix Transposed file format: [https://software.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data\\_formats#GMT:\\_Gene\\_Matrix\\_Transposed\\_file\\_format\\_.28.2A.gmt.29](https://software.broadinstitute.org/cancer/software/gsea/wiki/index.php/Data_formats#GMT:_Gene_Matrix_Transposed_file_format_.28.2A.gmt.29)

### Value

`NULL`. Output written to the file path specified.

### See Also

[print.pathwayCollection](#); [read\\_gmt](#)

### Examples

```
# Toy pathway set
toy_pathwayCollection <- list(
  pathways = list(
    c("C1orf27", "NR5A1", "BLOC1S4", "C4orf50"),
    c("TARS2", "DUSP5", "GPR88"),
    c("TRX-CAT3-1", "LINC01333", "LINC01499", "LINC01046", "LINC01149")
  ),
  TERMS = c("C-or-f_paths", "randomPath2", "randomLINC"),
  description = c("these are", "totally made up", "pathways")
)
class(toy_pathwayCollection) <- c("pathwayCollection", "list")
toy_pathwayCollection

# write_gmt(toy_pathwayCollection, file = "example_pathway.gmt")
```

# Index

## \* datasets

- colon\_pathwayCollection, 8
- colonSurv\_df, 7
- wikipwsHS\_Entrez\_pathwayCollection, 39
- wikipwsHS\_Symbol\_pathwayCollection, 39
- [[.pathwayCollection (SubsetPathwayCollection), 32
- aespca, 4
- AESPCA\_pVals, 4, 5, 16–20
- AESPCA\_pVals, OmicsPathway-method (AESPCA\_pVals), 5
- CheckAssay, 11
- CheckPwyColl, 11
- class, 37
- clusterApply, 7, 36
- colon\_pathwayCollection, 8
- colonSurv\_df, 7, 8
- Contains, 9, 38
- ControlFDR, 6, 35
- CreateOmics, 7, 10, 22–24, 29, 30, 32, 36
- CreateOmicsCateg, 10, 11
- CreateOmicsCateg (CreateOmicsPath), 12
- CreateOmicsPath, 10, 11, 12
- CreateOmicsReg, 10, 11
- CreateOmicsReg (CreateOmicsPath), 12
- CreateOmicsSurv, 10, 11
- CreateOmicsSurv (CreateOmicsPath), 12
- CreatePathwayCollection, 15, 33
- ExtractAESPCs, 4, 6, 7, 20
- getAssay (SubsetOmicsPath), 27
- getAssay, OmicsPathway-method (SubsetOmicsPath), 27
- getAssay<- (SubsetOmicsPath), 27
- getAssay<- , OmicsPathway-method (SubsetOmicsPath), 27
- getEvent (SubsetOmicsSurv), 31
- getEvent, OmicsSurv-method (SubsetOmicsSurv), 31
- getEvent<- (SubsetOmicsSurv), 31
- getEvent<- , OmicsSurv-method (SubsetOmicsSurv), 31
- getEventTime (SubsetOmicsSurv), 31
- getEventTime, OmicsSurv-method (SubsetOmicsSurv), 31
- getEventTime<- (SubsetOmicsSurv), 31
- getEventTime<- , OmicsSurv-method (SubsetOmicsSurv), 31
- getPathPCLs, 16
- getPathpVals, 18
- getPathwayCollection (SubsetOmicsPath), 27
- getPathwayCollection, OmicsPathway-method (SubsetOmicsPath), 27
- getPathwayCollection<- (SubsetOmicsPath), 27
- getPathwayCollection<- , OmicsPathway-method (SubsetOmicsPath), 27
- getResponse (SubsetOmicsResponse), 29
- getResponse, OmicsPathway-method (SubsetOmicsResponse), 29
- getResponse<- (SubsetOmicsResponse), 29
- getResponse<- , OmicsPathway-method (SubsetOmicsResponse), 29
- getSampleIDs (SubsetOmicsPath), 27
- getSampleIDs, OmicsPathway-method (SubsetOmicsPath), 27
- getSampleIDs<- (SubsetOmicsPath), 27
- getSampleIDs<- , OmicsPathway-method (SubsetOmicsPath), 27
- getTrimPathwayCollection (SubsetOmicsPath), 27
- getTrimPathwayCollection, OmicsPathway-method (SubsetOmicsPath), 27
- GumbelMixpValues, 35, 36

- IntersectOmicsPwyCollct, [11](#), [28](#)
- lars.lsa, [4](#)
- list, [16](#)
- LoadOntoPCs, [20](#)
- normalize, [4](#)
- OmicsCateg, [11](#), [14](#)
- OmicsCateg-class, [21](#)
- OmicsPathway, [11](#), [14](#), [22–24](#)
- OmicsPathway-class, [22](#)
- OmicsReg, [11](#), [14](#)
- OmicsReg-class, [23](#)
- OmicsSurv, [11](#), [14](#)
- OmicsSurv-class, [23](#)
- OptimGumbelMixParams, [35](#), [36](#)
  
- pathway\_tControl, [35](#), [36](#)
- pathway\_tScores, [35](#), [36](#)
- pathwayPCA, [24](#)
- PermTestCateg, [6](#), [7](#)
- PermTestReg, [6](#), [7](#)
- PermTestSurv, [6](#), [7](#)
- print.pathwayCollection, [26](#), [41](#)
  
- read\_gmt, [10](#), [15](#), [16](#), [25](#), [32](#), [38](#), [41](#)
- readChar, [25](#)
- readLines, [25](#)
- rownames, [37](#)
  
- scale, [11](#)
- scan, [25](#)
- SE2Tidy, [26](#)
- SubsetOmicsPath, [27](#)
- SubsetOmicsResponse, [29](#)
- SubsetOmicsSurv, [31](#)
- SubsetPathwayCollection, [32](#)
- SubsetPathwayData, [33](#)
- SubsetPathwayData, OmicsPathway-method  
    (SubsetPathwayData), [33](#)
- superpc.train, [35](#)
- SuperPCA\_pVals, [16–20](#), [34](#)
- SuperPCA\_pVals, OmicsPathway-method  
    (SuperPCA\_pVals), [34](#)
- Surv, [11](#)
  
- TabulatepValues, [7](#), [35](#), [36](#)
- tibble, [7](#), [20](#), [34](#), [36](#)
- TransposeAssay, [27](#), [36](#)
  
- ValidOmicsCateg, [28](#), [30](#)
- ValidOmicsReg, [28](#), [30](#)
- ValidOmicsSurv, [28](#), [31](#)
  
- WhichPathways, [38](#)
- wikipwsHS\_Entrez\_pathwayCollection, [39](#)
- wikipwsHS\_Symbol\_pathwayCollection, [39](#)
- write\_gmt, [26](#), [40](#)