Package ‘rcellminer’

January 31, 2017

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
<td>rcellminer: Molecular Profiles and Drug Response for the NCI-60 Cell Lines</td>
</tr>
<tr>
<td>Version</td>
<td>1.6.0</td>
</tr>
<tr>
<td>Date</td>
<td>2016-01-27</td>
</tr>
<tr>
<td>Author</td>
<td>Augustin Luna, Vinodh Rajapakse, Fabricio Sousa</td>
</tr>
<tr>
<td>Maintainer</td>
<td>Augustin Luna <a href="mailto:lunaa@cbio.mskcc.org">lunaa@cbio.mskcc.org</a>, Vinodh Rajapakse <a href="mailto:vinodh.rajapakse@nih.gov">vinodh.rajapakse@nih.gov</a></td>
</tr>
</tbody>
</table>

| Description | The NCI-60 cancer cell line panel has been used over the course of several decades as an anti-cancer drug screen. This panel was developed as part of the Developmental Therapeutics Program (DTP, http://dtp.nci.nih.gov/) of the U.S. National Cancer Institute (NCI). Thousands of compounds have been tested on the NCI-60, which have been extensively characterized by many platforms for gene and protein expression, copy number, mutation, and others (Reinhold, et al., 2012). The purpose of the CellMiner project (http://discover.nci.nih.gov/cellminer) has been to integrate data from multiple platforms used to analyze the NCI-60 and to provide a powerful suite of tools for exploration of NCI-60 data. |
| License    | LGPL-3      |
| LazyData   | true        |
| Imports    | stringr, gplots, methods, shiny |
| Depends    | R (>= 3.2), Biobase, rcdk, fingerprint, rcellminerData |
| Suggests   | knitr, RColorBrewer, sqldf, BiocGenerics, testthat, BiocStyle, jsonlite |
| URL        | http://discover.nci.nih.gov/cellminer/ |
| VignetteBuilder | knitr |
| biocViews  | aCGH, CellBasedAssays, CopyNumberVariation, GeneExpression, Pharmacogenomics, Pharmacogenetics, miRNA, Cheminformatics, Visualization, Software, SystemsBiology |
| NeedsCompilation | no |

R topics documented:

```
  cmVersion .......................................................... 3
```
topics documented:

- compareFingerprints .................................................. 4
- crossCors ............................................................... 4
- crossCorsSpearman ...................................................... 5
- DrugData ................................................................. 6
- DrugData,eSet,eSet,MIAxE-method ................................... 6
- DrugData-class ........................................................... 7
- drugDB .................................................................... 7
- Drug_MOA_Key ............................................................ 8
- elNetMolDataNCI60 ...................................................... 8
- fingerprintList .......................................................... 9
- getAct ..................................................................... 9
- getAct,DrugData-method .............................................. 10
- getActivityRangeStats ................................................ 10
- getAllFeatureData ...................................................... 11
- getAllFeatureData,MolData-method ................................ 11
- getBinaryMutationData ................................................ 12
- getColumnQuantiles ..................................................... 13
- getDrugActivityData ................................................... 13
- getDrugActivityRange .................................................. 14
- getDrugActivityRepeatData .......................................... 14
- getDrugMoaList .......................................................... 15
- getDrugName ............................................................ 16
- getESetList ............................................................... 16
- getESetList,MolData-method ........................................ 17
- getFeatureAnnot ........................................................ 17
- getFeatureAnnot,DrugData-method ................................ 18
- getFeatureAnnot,MolData-method .................................. 18
- getFeatureDataFromMatList .......................................... 19
- getFingerprintList ..................................................... 19
- getMedSenLineActivity ............................................... 20
- getMinDrugActivityRepeatCor ...................................... 21
- getMoaStr ............................................................... 21
- getMoaToCompounds ................................................... 22
- getMolDataMatrices .................................................... 22
- getMolDataType ........................................................ 23
- getNumDrugActivityRepeats ......................................... 23
- getNumMissingLines ................................................... 24
- getRepeatAct ............................................................ 24
- getRepeatAct,DrugData-method .................................... 25
- getRsd ................................................................. 25
- getSampleData .......................................................... 26
- getSampleData,DrugData-method .................................. 26
- getSampleData,MolData-method .................................... 27
- getSmiles ............................................................... 27
- hasMoa ................................................................. 28
- initialize,DrugData-method ....................................... 28
- initialize,MolData-method .......................................... 29
- isPublic ............................................................... 29
- loadCellminerPlotInfo ................................................ 30
- loadNciColorSet ........................................................ 30
- MolData ................................................................. 31
- MolData,list,MIAxE-method ........................................ 31
<table>
<thead>
<tr>
<th>cmVersion</th>
<th>CellMiner Version</th>
</tr>
</thead>
</table>

**Description**

CellMiner Version

**Details**

The version of CellMiner used

**Author(s)**

Vinodh Rajapakse <vinodh.rajapakse AT nih.gov>

**References**

http://discover.nci.nih.gov/cellminer
### compareFingerprints

**Compare Structure Fingerprints to NCI DTP Compounds**

**Description**

This function compares the first SMILES structure to all other structures given.

**Usage**

```r
compareFingerprints(ids = NULL, smiles = NULL, fpType = "standard",
                    verbose = TRUE, fingerprint.list = NULL)
```

**Arguments**

- `ids` a vector of IDs corresponding to structures
- `smiles` a vector of strings SMILES structures
- `fpType` the type of fingerprint to be used; uses the RCDK get.fingerprint() (default: standard)
- `verbose` a boolean whether to display debugging information
- `fingerprint.list` a list of fingerprints generated with getFingerprintList

**Value**

a sorted named vector of Tanimoto distances

**See Also**

rcdk::get.fingerprint

**Examples**

```r
drugAnnot <- as(featureData(getAct(rcellminerData::drugData)), "data.frame")
ids <- head(drugAnnot$NSC)
smiles <- head(drugAnnot$SMILES)
fingerprintList <- getFingerprintList(ids, smiles)
compareFingerprints(fingerprint.list=fingerprintList)
```

---

### crossCors

**Calculate cross-correlations with between rows of input matrices**

**Description**

Calculate cross-correlations with between rows of input matrices

**Usage**

```r
crossCors(X, Y = NULL, method = "pearson")
```
crossCorsSpearman

Arguments

<table>
<thead>
<tr>
<th>X</th>
<th>a matrix or data.frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>a matrix or data.frame</td>
</tr>
<tr>
<td>method</td>
<td>a string specifying the type of correlation, chosen from pearson (default) or spearman.</td>
</tr>
</tbody>
</table>

Value

a list containing matrices of pairwise correlations and their p-values between rows of the input matrices or dataframes.

Author(s)

Sudhir Varma, NCI-LMP, with input checks, support for Spearman’s correlation added by VNR.

Examples

```
drugActData <- exprs(getAct(rcellminerData::drugData))
crossCors(drugActData[c("94600"), ], drugActData[c("727625", "670655"), ])
crossCors(drugActData[c("94600"), ], drugActData[c("727625", "670655"), ], method="spearman")
```

crossCorsSpearman

**Calculate Spearman’s correlations with between rows of input matrices**

Description

Calculate Spearman’s correlations with between rows of input matrices

Usage

```
crossCorsSpearman(X, Y = NULL)
```

Arguments

<table>
<thead>
<tr>
<th>X</th>
<th>a matrix or data.frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>a matrix or data.frame</td>
</tr>
</tbody>
</table>

Value

a list containing matrices of pairwise Spearman’s correlations and their p-values between rows of the input matrices or dataframes.

Examples

```
## Not run:
crossCorsSpearman(drugActData[c("94600"), ], drugActData[c("727625", "670655"), ])
## End(Not run)
```
DrugData

Returns a DrugData object.

Description

Returns a DrugData object.

Usage

DrugData(act, repeatAct, sampleData, ...)

Arguments

- **act**: An eSet object containing drug activity data across a set of biological samples.
- **repeatAct**: An eSet object containing repeat drug activity experiment data with respect to the same samples associated with act.
- **sampleData**: A MIAxE object capturing sample and other data set information.
- **...**: Other possible parameters.

Value

A DrugData object.

---

DrugData,eSet,eSet,MIAxE-method

Returns a DrugData object.

Description

Returns a DrugData object.

Usage

## S4 method for signature 'eSet,eSet,MIAxE'

DrugData(act, repeatAct, sampleData, ...)

Arguments

- **act**: An eSet object containing drug activity data across a set of biological samples.
- **repeatAct**: An eSet object containing repeat drug activity experiment data with respect to the same samples associated with act.
- **sampleData**: A MIAxE object capturing sample and other data set information.
- **...**: Other possible parameters.

Value

A DrugData object.
DrugsData-class

Description

An S4 class to represent drug activity and related data recorded for a set of biological samples.

Arguments

... Other possible parameters.

Slots

act An eSet object containing drug activity data across a set of biological samples.
repeatAct An eSet object containing repeat drug activity experiment data with respect to the same samples associated with act.
sampleData A MIAxE object capturing sample and other data set information.

drugDB

CellMiner Drug Response Values

Description

CellMiner Drug Response Values

Details

A list containing response values and annotations:

- act Z-scores of the averaged negative log GI (growth inhibition) 50 values across repeats for the NCI-60; assay described here: http://dtp.nci.nih.gov/branches/btb/ivclsp.html
- annot
  - id Dataset identifier; NOTE: DO NOT use this column; the NSC is the primary drug identifier
  - nsc National Service Center identifier; the primary drug identifier
  - name Compound name
  - brand_name Brand name for the compound, if sold commercially
  - formula Compound chemical formula
  - testing_status Information on whether it is known if the compound is FDA approved or undergoing testing in clinical trials
  - source TODO
  - smiles Compound chemical structure as a SMILES string
  - weight Compound chemical weight in g/mol
  - mechanism Pharmacological mechanism of action
  - confidential_flag A flag to indicate if compound information is public
  - total_probes TODO
Author(s)

Vinodh Rajapakse <vinodh.rajapakse AT nih.gov>

References

http://discover.nci.nih.gov/cellminer/loadDownload.do

<table>
<thead>
<tr>
<th>Drug_MOA_Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drug_MOA_Key</td>
<td>A data frame with descriptive information for all compound mechanism of action (MOA) abbreviations used in CellMiner.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>elNetMolDataNCI60</th>
<th>NCI60 Molecular Data</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Z-scores of values for a variety of assays conducted on the NCI-60 to facilitate comparison. Z-scores calculated over the 60 cell lines for the given feature.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>A list containing various assay values:</td>
</tr>
</tbody>
</table>

- cop Copy number values; Described in Pubmed ID: 24670534
- exp Expression values; Obtained from "RNA: 5 Platform Gene Transcript" http://discover.nci.nih.gov/cellminer/loadDownload.do; Missing values imputed using the R package "impute"
- mut Mutation values; Deleterious mutations obtained from TODO
- mir MicroRNA values; Obtained from "RNA: Agilent Human microRNA (V2)" http://discover.nci.nih.gov/cellminer/loadDownload.do
- pro Reverse protein lysate array values; Obtain from "Protein: Lysate Array" http://discover.nci.nih.gov/cellminer/loadDownload.do
- mda NCI-60 metadata.
  - CNV_GAIN Proportion of genome copy number gains; Described in Pubmed ID: 24670534
  - CNV_LOSS Proportion of genome copy number losses; Described in Pubmed ID: 24670534
fingerprintList 

- CNV_TOTAL Sum of CNV_GAIN and CNV_LOSS
- P53_BIN Binary TP53 profile curated by William Reinhold
- MSI_OGAN_BIN Binary microsatellite instability (MSI) profile curated by Ogan Abaan using COSMIC data; Obtained from Supplementary Table 1 - Ogan Whole Exome Sequencing (WES) paper in Cancer Res.
- EPITHELIAL Epithelial by tissue of origin - pattern extracted from the CellMiner cell line metadata http://discover.nci.nih.gov/cellminer/celllineMetadata.do
- EPITHELIAL_KURT Kurt Kohn curation for epithelial-like cell lines based on molecular parameters described in Pubmed ID: 24940735
- DELETERIOUS Total deleterious variants from WES dataset; Fabricio Sousa curation
- MISSENSE Total missense variants from WES dataset; Fabricio Sousa curation
- SILENT Total silent variants from WES dataset; Fabricio Sousa curation
- TOTAL_AA Total amino acid changing variants from WES dataset; Fabricio Sousa curation
- CELL-CELL Cell-to-cell adhesion curated by William Reinhold
- DOUBLINGTIME The doubling time pattern was extracted from the CellMiner cell line metadata http://discover.nci.nih.gov/cellminer/celllineMetadata.do

Author(s)
Vinodh Rajapakse <vinodh.rajapakse AT nih.gov>

References
http://discover.nci.nih.gov/cellminer

```
fingerprintList          A list of pre-computed fingerprints using getFingerprintList()

Description
A list of pre-computed fingerprints using getFingerprintList()

getAct                   Returns an eSet object with drug activity data.

Description
Returns an eSet object with drug activity data.

Usage
getAct(object, ...)

Arguments
object                Object for which drug activity data is to be returned.
...                    Other possible parameters.
getActivityRangeStats

**Description**

Returns a table of activity range statistics for a set of compounds.

**Usage**

```r
getActivityRangeStats(nscSet, concFormat = "NegLogGI50M", onlyCellMinerExps = TRUE)
```

**Arguments**

- `nscSet` a character vector specifying NSC identifier(s) for compound(s) of interest.
- `concFormat` a string selected from "NegLogGI50M" or "IC50MicroM". "NegLogGI50M" specifies activities as the negative log of the 50 inhibitory concentration (molar). "IC50MicroM" specifies activities as the 50 inhibitory concentration (micromolar).
- `onlyCellMinerExps` a logical value indicating whether to only return experimental data included in CellMiner (default=TRUE).

**Value**

a table of activity range statistics for a set of compounds.
`getAllFeatureData`  

**Examples**

```r
nscSet <- c("609699", "740")
getActivityRangeStats(nscSet)
getActivityRangeStats(nscSet, concFormat="IC50MicroM")
```

---

**Description**

Returns a list of feature data matrices.

**Usage**

```r
getAllFeatureData(object, 
...)```

**Arguments**

- `object`  
  Object for which a list of feature data matrices is to be returned.
- `...`  
  Other possible parameters.

**Value**

A list of feature data matrices.

---

`getAllFeatureData, MolData-method`

---

**Description**

Returns a list of feature data matrices.

**Usage**

```r
## S4 method for signature 'MolData'
getAllFeatureData(object)
```

**Arguments**

- `object`  
  MolData object for which a list of feature data matrices is to be returned.

**Value**

A list of feature data matrices.
getBinaryMutationData  

Compute a binary gene mutation data matrix from SNP and other mutation event-level data.

Description

Compute a binary gene mutation data matrix from SNP and other mutation event-level data.

Usage

getBinaryMutationData(mutInfo, mutData, maxVariantFreq = 0.2, maxNormalPopulationFreq = 0.005, maxSiftScore = 0.05, minPolyPhenScore = 0.85)

Arguments

mutInfo  
A data frame with the following named columns: Gene, the name of the gene associated with the mutation event; probe.ids, a unique identifier specifying the mutation event; SNP_1000_genome, the frequency of the mutation event in SNP 1000; ESP5400, the frequency of the mutation event in ESP5400; SNP_type, the type of mutation event, chosen from "MISSENSE", "FRAMESHIFT", "NON-FRAMESHIFT", "NONSENSE", "SPlicing", SIFT_score, the SIFT score; Polyphen_score, the POLYPHEN score. Rownames of mutInfo should be set to probe.ids, i.e., the unique mutation event specifier.

mutData  
A matrix with event level mutation information, with SNPs, etc. along rows and samples along columns. Rownames of mutData should exactly match those of mutInfo. The i-th row of mutInfo should thus give detailed information for the mutation event with data specified in the i-th row of mutData.

maxVariantFreq  
The maximum proportion of mutant samples (used to exclude frequently occurring events); default value = 0.2.

maxNormalPopulationFreq  
The maximum frequency of a mutation in the normal population (used to exclude likely germline variants); default value = 0.005.

maxSiftScore  
The maximum accepted SIFT score (used to exclude presumed non-deleterious mutations); default value = 0.05.

minPolyPhenScore  
The minimum accepted POLYPHEN score (used to exclude presumed non-deleterious mutations); default value = 0.85.

Value

A binary gene mutation matrix, with genes along rows, samples along columns, and 1s indicating deleterious mutations.
getColumnQuantiles

Description
Calculate quantile for the columns in a matrix

Usage
getColumnQuantiles(X, prob, naRm = FALSE, onlyNonzeroVals = FALSE)

Arguments
X
the matrix
prob
a numeric probability
naRm
a boolean, whether to remove NAs
onlyNonzeroVals
a boolean, whether to only include non-zero values

Value
a vector of quantiles

Examples
getColumnQuantiles(matrix(1:25, nrow=5), prob = 0.5)

getDrugActivityData

Returns a matrix containing activity (-logGI50) data for a set of compounds.

Description
Returns a matrix containing activity (-logGI50) data for a set of compounds.

Usage
getDrugActivityData(nscSet, onlyCellMinerExps = TRUE)

Arguments
nscSet
A string specifying the NSC identifiers for the compounds.
onlyCellMinerExps
A logical value indicating whether to compute results using only experimental
data included in CellMiner (default=TRUE).

Value
a matrix with NCI-60 average (over experiments) -logGI50 activity data; compound activity profiles
are along rows.
**Examples**

```r
nscSet <- c("141540", "123127") # Etoposide, Doxorubicin.
actData <- getDrugActivityData(nscSet)
```

---

**getDrugActivityRange**  
*Returns a vector of log activity range values for set of compounds.*

**Description**

Returns a vector of log activity range values for set of compounds.

**Usage**

```r
getDrugActivityRange(nscSet, computeIQR = FALSE)
```

**Arguments**

- `nscSet`: a character vector specifying NSC identifier(s) for compound(s) of interest.
- `computeIQR`: logical value indicated whether inter-quartile range is to be computed; otherwise absolute range is computed (default=FALSE).

**Value**

a numeric vector of NCI-60 log activity (-logGI50) range values indexed by the identifiers in nscSet.

**Examples**

```r
nscSet <- c("609699", "740")
getDrugActivityRange(nscSet)
```

---

**getDrugActivityRepeatData**  
*Returns a matrix containing repeat activity experiment data for a compound.*

**Description**

Returns a matrix containing repeat activity experiment data for a compound.

**Usage**

```r
getDrugActivityRepeatData(nscStr, concFormat = "NegLogGI50M", onlyCellMinerExps = TRUE)
```
**getDrugMoaList**

Get a list of applicable MOA strings for a drug.

**Description**

Get a list of applicable MOA strings for a drug.

**Usage**

getDrugMoaList(nsc, moaToCompoundListMap = NULL)

**Arguments**

nsc

An NSC string.

moaToCompoundListMap

A named list of character vectors, with each name indicating an MOA class, and its corresponding character vector specifying MOA-associated drugs. If unspecified, this is constructed based on MOA information provided by CellMiner.

**Value**

A character vector giving all MOA classes for the drug.

**Examples**

getDrugMoaList("754365")
getDrugName

Get the drug names for a set of NSC identifiers.

Description
Get the drug names for a set of NSC identifiers.

Usage
getDrugName(nscSet)

Arguments
nscSet: A character vector of NSC strings

Value
A named character vector indicating the compound names for each NSC in nscSet (with an empty string returned if no such information is available, and an NA returned if the NSC is not included in the CellMiner database).

Examples
nscSet <- c("609699", "94600")
getDrugName(nscSet)

getESetList

Returns a list of eSet objects.

Description
Returns a list of eSet objects.

Usage
getESetList(object, ...)

Arguments
object: Object for which a list of eSets is to be returned.
...: Other possible parameters.

Value
A list of eSet objects.
getESetList,MolData-method

Returns a list of eSet objects.

Description

Returns a list of eSet objects.

Usage

## S4 method for signature 'MolData'
getESetList(object)

Arguments

object MolData object for which a list of eSet objects is to be returned.

Value

A list of eSet objects.

getFeatureAnnot

Returns a list of data frames with feature information.

Description

Returns a list of data frames with feature information.

Usage

getFeatureAnnot(object, ...)

Arguments

object Object for which feature data is to be returned.
... Other possible parameters.

Value

A list of data frames with feature information.
getFeatureAnnot, DrugData-method

Returns a list of data frames with feature information.

Description

Returns a list of data frames with feature information.

Usage

```r
## S4 method for signature 'DrugData'
getFeatureAnnot(object)
```

Arguments

- `object`: DrugData object for which feature data is to be returned.

Value

A named list of data frames with feature information for drugs and drug repeat experiments.

getFeatureAnnot, MolData-method

Returns a list of data frames with feature information.

Description

Returns a list of data frames with feature information.

Usage

```r
## S4 method for signature 'MolData'
getFeatureAnnot(object)
```

Arguments

- `object`: MolData object for which feature data is to be returned.

Value

A named list of data frames with feature information for available molecular data types.
**getFeatureDataFromMatList**

Extract from a list of matrices the data associated with a set of features.

**Usage**

```r
getFeatureDataFromMatList(featureSet, dataMatList, 
excludeMissingFeatures = TRUE)
```

**Arguments**

- `featureSet` a character vector of feature names.
- `dataMatList` a list of matrices with feature data organized along the rows, and feature names accessible via rownames(dataMatList).
- `excludeMissingFeatures` a logical value indicating whether features whose data cannot be found in any matrices in dataMatList should be excluded in the output. (default=TRUE).

**Value**

a single matrix containing data for all features in featureSet.

**Examples**

```r
featureSet <- c("expSLFN11", "mutSLX4")
molDataMats <- getMolDataMatrices()
featureData <- getFeatureDataFromMatList(featureSet, molDataMats)
```

**getFingerprintList**

Get a list of fingerprints for a set of compounds

**Usage**

```r
getFingerprintList(ids, smiles, fpType = "standard", verbose = TRUE)
```

**Arguments**

- `ids` a vector of IDs corresponding to structures
- `smiles` a vector of strings SMILES structures
- `fpType` the type of fingerprint to be used; uses the RCDK get.fingerprint() (default: standard)
- `verbose` show debugging output

**Description**

Get a list of fingerprints for a set of compounds
getMedSenLineActivity

Returns a vector of median sensitive cell line activity (-logGI50) values for a set of compounds.

Description

Returns a vector of median sensitive cell line activity (-logGI50) values for a set of compounds.

Usage

getMedSenLineActivity(idSet, senLineActZThreshold = 0.5,
                       onlyCellMinerExps = TRUE, dataSource = "NCI60")

Arguments

- idSet a character vector specifying identifier(s) for compound(s) of interest.
- senLineActZThreshold the minimum activity z-score for a sensitive cell line (default=0.5).
- onlyCellMinerExps a logical value indicating whether to base results strictly on experimental data included in CellMiner (default=TRUE).
- dataSource character string indicating data source (default="NCI60"). Currently only "NCI60" is supported.

Value

a numeric vector of median sensitive cell line activity (-logGI50) values indexed by the identifiers in idSet.
### getMinDrugActivityRepeatCor

**Description**

Returns a table indicating, for each compound in a specified set, the least significant correlation and associated p-value between its replicate experiments.

**Usage**

`getMinDrugActivityRepeatCor(nscSet)`

**Arguments**

- `nscSet` a character vector specifying NSC identifier(s) for compound(s) of interest.

**Value**

a dataframe containing the following columns: NSC, cor, pval

**Examples**

```r
nscSet <- c("123528", "339316")
repExpCorTab <- getMinDrugActivityRepeatCor(nscSet)
```

### getMoaStr

**Description**

Get MOA string

**Usage**

`getMoaStr(nscStr)`

**Arguments**

- `nscStr` an NSC string

**Details**

LINK TO MOAs?
getMolDataMatrices

Value

A comma-delimited string with MOA

Examples

getMoaStr("94600")
getMoaStr(c("94600", "609699"))

getMoaToCompounds

Get a named list mapping MOA classes to associated compound sets.

Description

Get a named list mapping MOA classes to associated compound sets.

Usage

getMoaToCompounds()

Value

A named list mapping MOA classes to associated compound sets (each represented as a character vector).

Examples

moaToCompounds <- getMoaToCompounds()

getMolDataMatrices

Returns a list of molecular data type matrices, with rownames in each matrix prefixed with a data type abbreviation.

Description

Returns a list of molecular data type matrices, with rownames in each matrix prefixed with a data type abbreviation.

Usage

getMolDataMatrices(molDataMats = NULL)

Arguments

molDataMats A named list of molecular data type matrices with feature data specified along the rows, and feature names indicated in the row names.

Value

A list containing molecular data type matrices, with rownames in each matrix prefixed with a data type abbreviation, e.g., 'exp' for mRNA expression, etc. The matrix-specific data type abbreviations are derived from the names of molDataMats.
getMolDataType

Examples

molDataMats <- getMolDataMatrices()

getMolDataType

Get the molecular data type prefixes for a set of features.

Description

Get the molecular data type prefixes for a set of features.

Usage

getMolDataType(features, prefixLen = 3)

Arguments

features A vector of features.
prefixLen The length of the molecular data type prefix.

Value

A character vector of molecular data type prefixes.

Examples

getMolDataType(c("expTP53", "copMDM2", "mutCHEK2", "mutBRAF"))

getNumDrugActivityRepeats

Returns a vector indicating the number of drug activity repeat experiments with available data for each member of a set of compounds.

Description

Returns a vector indicating the number of drug activity repeat experiments with available data for each member of a set of compounds.

Usage

ggetNumDrugActivityRepeats(nscSet, onlyCellMinerExps = TRUE)

Arguments

nscSet a character vector specifying NSC identifier(s) for compound(s) of interest.
onlyCellMinerExps a logical value indicating whether to return only the number of experiments with data included in CellMiner (default=TRUE).
getRepeatAct

Value

a numeric vector, indexed by nscSet, indicating the number of drug activity repeat experiments for each one of its compounds.

Examples

nscSet <- c("1", "17", "89", "609699")
getNumDrugActivityRepeats(nscSet)

getRepeatAct

Returns an eSet object with drug repeat activity experiment data.

Description

Returns an eSet object with drug repeat activity experiment data.

Usage

getRepeatAct(object, ...)

Arguments

object Object for which drug repeat activity experiment data is to be returned.
...
Other possible parameters.

getNumMissingLines

Returns a vector indicating the number of NCI-60 cell lines with missing activity data for set of compounds.

Description

Returns a vector indicating the number of NCI-60 cell lines with missing activity data for set of compounds.

Usage

getNumMissingLines(nscSet)

Arguments

nscSet a character vector specifying NSC identifier(s) for compound(s) of interest.

Value

a numeric vector indicating the number of NCI-60 cell lines with missing activity data, indexed by the identifiers in nscSet.

Examples

nscSet <- c("1", "17", "89", "609699")
getNumMissingLines(nscSet)
Value

An eSet object with drug repeat activity experiment data.

Description

Returns an eSet object with drug repeat activity experiment data.

Usage

```r
## S4 method for signature 'DrugData'
getRepeatAct(object)
```

Arguments

- `object`: DrugData object for which drug repeat activity experiment data is to be returned.

Value

An eSet object with drug repeat activity experiment data.

getRsd

Computes the relative standard deviation values with respect to the columns of a matrix or data.frame.

Description

Computes the relative standard deviation values with respect to the columns of a matrix or data.frame.

Usage

```r
getRsd(dat, onlyReturnMedian = TRUE)
```

Arguments

- `dat`: a matrix or data.frame with numeric values.
- `onlyReturnMedian`: a logical value indicating whether only the median column RSD value should be returned (vs. all RSD values).

Value

median RSD value over the data set columns or all RSD values, depending on value of onlyReturnMedian (default=TRUE).

Examples

```r
A <- matrix(rnorm(10*60), nrow=10)
getRsd(A)
getRsd(A, onlyReturnMedian=FALSE)
```
getSampleData

Returns a data frame with sample information.

Description

Returns a data frame with sample information.

Usage

getSampleData(object, ...)

Arguments

object

Object for which sample data is to be returned.

...

Other possible parameters.

Value

A data frame with sample information.

getSampleData, DrugData-method

Returns a data frame with sample information.

Description

Returns a data frame with sample information.

Usage

## S4 method for signature 'DrugData'
getSampleData(object)

Arguments

object

DrugData object for which sample data is to be returned.

Value

A data frame with sample information.
getSampleData, MolData-method

Returns a data frame with sample information.

Description

Returns a data frame with sample information.

Usage

## S4 method for signature 'MolData'
getSampleData(object)

Arguments

object MolData object for which sample data is to be returned.

Value

A data frame with sample information.

getSmiles

Get the SMILES strings for a set of NSC identifiers.

Description

Get the SMILES strings for a set of NSC identifiers.

Usage

getSmiles(nscSet)

Arguments

nscSet A character vector of NSC strings

Value

A named character vector indicating the SMILES string for each NSC in nscSet (or NA if no structural information is available).

Examples

nscSet <- c("609699", "94600")
getSmiles(nscSet)
hasMoa

Check if NSC has Mechanism of Action (MOA) Annotation

Description
Check if NSC has Mechanism of Action (MOA) Annotation

Usage
hasMoa(nsc)

Arguments
nsc a string, an NSC identifier

Value
a boolean whether the NSC has an MOA

Examples
hasMoa("754365")

initialize,DrugData-method

Returns a DrugData object.

Description
Returns a DrugData object.

Usage
## S4 method for signature 'DrugData'
initialize(.Object, act, repeatAct, sampleData)

Arguments
.Object An object: see "new()" documentation in "methods" package.
act An eSet object containing drug activity data across a set of biological samples.
repeatAct An eSet object containing repeat drug activity experiment data with respect to the same samples associated with act.
sampleData A MIAxE object capturing sample and other data set information.

Value
A DrugData object.

Note
Seems to be required for definition of a constructor.
initialize,MolData-method

Returns a MolData object.

Description

Returns a MolData object.

Usage

## S4 method for signature 'MolData'
initialize(.Object, eSetList, sampleData)

Arguments

.Object An object: see "new()" documentation in "methods" package.
eSetList A list of eSet objects for a common set of samples.
sampleData A MIAxE object capturing sample and other data set information.

Value

A MolData object.

isPublic

Check if an NSC ID is public

Description

Check if an NSC ID is public.

Usage

isPublic(nscs)

Arguments

nscts a vector of NSC string IDs

Value

a vector of boolean values of whether each NSC is public

Examples

isPublic("-1")
isPublic(c("-1", "609699"))
loadCellminerPlotInfo  

Returns data to plot CellMiner plots

Description

Returns data to plot CellMiner plots

Usage

loadCellminerPlotInfo(returnDf = FALSE)

Arguments

returnDf a boolean if a data.frame with all information (default: FALSE)

Value

a vector of colors as strings or a data.frame with dataType, label, xMin, xMax

Examples

loadCellminerPlotInfo()

loadNciColorSet  

Returns a 60-element color set that matches the color set used on http://discover.nci.nih.gov/

Description

Returns a 60-element color set that matches the color set used on http://discover.nci.nih.gov/

Usage

loadNciColorSet(returnDf = FALSE)

Arguments

returnDf a boolean if a data.frame with tissue names and abbreviations should be returned (default: FALSE)

Value

a vector of colors as strings or a data.frame with tissues, tissue abbreviations, cell line abbreviations and colors

Examples

loadNciColorSet()
MolData

Returns a MolData object.

Description

Returns a MolData object.

Usage

MolData(eSetList, sampleData, ...)

Arguments

eSetList A list of eSet objects for a common set of samples.
sampleData A MIAxE object capturing sample and other data set information.
... Other possible parameters.

Value

A MolData object.

MolData,list,MIAXE-method

Returns a MolData object.

Description

Returns a MolData object.

Usage

## S4 method for signature 'list,MIAXE'
MolData(eSetList, sampleData, ...)

Arguments

eSetList A list of eSet objects for a common set of samples.
sampleData A MIAxE object capturing sample and other data set information.
... Other possible parameters.

Value

A MolData object.
MolData-class

An S4 class to represent molecular data recorded for a set of biological samples.

Description

An S4 class to represent molecular data recorded for a set of biological samples.

Arguments

... Other possible parameters.

Slots

eSetList A list of eSet objects for a common set of samples.
sampleData A MIAXE object capturing sample and other data set information.

passRuleOf5

Checks if SMILES passes Lipinski’s Rule of 5

Description

Checks if SMILES passes Lipinski’s Rule of 5

Usage

passRuleOf5(smiles, acceptableViolations = 0, verbose = FALSE)

Arguments

smiles a string, the SMILES structure to be checked
acceptableViolations, a number, the number of acceptable rule violations (default: 0)
verbose a boolean, whether to write out the failing criteria (default: FALSE)

Details

Uses RCDK: org.openscience.cdk.qsar.descriptors.molecular.RuleOfFiveDescriptor

Value

a boolean, whether the SMILES passes the criteria

Examples

# Docetaxel
passRuleOf5("CC1=C2C(C(=O)C3(C(CC4C(C(C2(C(C10C(=O)C(C(C5=CC=CC=C5)NC(=O)C(C)C)O)0)0)C6=CC=CC=C6)(CO4)OC(=O)C)O)O)C)O", verbose=TRUE)

# Gemcitabine
passRuleOf5("Cl=CN(C(=O)N=C1N)C2C(C(C02)CO)O(F)F", verbose=TRUE)
passRuleOf5FromNsc  Checks if NSC passes Lipinski’s Rule of 5

Description
Checks if NSC passes Lipinski’s Rule of 5

Usage
passRuleOf5FromNsc(nsc, acceptableViolations = 0, verbose = FALSE)

Arguments
- **nsc**: a string, the NSC identifier to be checked
- **acceptableViolations**: a number, the number of acceptable rule violations (default: 0)
- **verbose**: a boolean, whether to write out the failing criteria (default: FALSE)

Details
Uses RCDK: org.openscience.cdk.qsar.descriptors.molecular.RuleOfFiveDescriptor

Value
a boolean or NA, whether the NSC passes the criteria or NA if the NSC had no structure

Examples
passRuleOf5FromNsc("94600", verbose=TRUE)

patternComparison  Compare an input pattern against a set of patterns.

Description
Compare an input pattern against a set of patterns.

Usage
patternComparison(pattern, profileMatrixList, method = "pearson")

Arguments
- **pattern**: An N element input pattern specified as either a named vector or an 1 x N matrix or data frame. Names (or column names) must match the column names of each element of profileMatrixList.
- **profileMatrixList**: A single matrix (or data frame) or a list of matrices (or data frames). Each matrix (data frame) must be k x N - that is the k patterns for comparison with the input pattern must be specified along the rows, with rownames set appropriately.
- **method**: a string specifying the type of correlation, chosen from pearson (default) or spearman.
Value

A data frame with pattern comparison results. Specifically, if M is the total number patterns in profileMatrixList elements, an M x 2 matrix is returned with sorted Pearson’s correlations in the first column and corresponding p-values in the second column. Comparison pattern names are indicated in the row names.

Examples

drugAct <- exprs(getAct(rcellminerData::drugData))
molDataMats <- getMolDataMatrices()[c("exp", "mut")]
molDataMats <- lapply(molDataMats, function(X) X[1:10, ])
pcResults <- patternComparison(drugAct["609699", ], molDataMats)
pcResults <- patternComparison(drugAct["609699", ], molDataMats, method="spearman")
pcResults <- patternComparison(drugAct["609699", ], molDataMats$exp, method="spearman")

plotCellMiner

Description: Produces CellMiner-like plots in R

Usage

plotCellMiner(drugAct, molData, plots, nsc = NULL, gene = NULL,
features = NULL, sub = NULL, xLimits = NULL, xLabel = NULL,
extraPlot = NULL, verbose = FALSE)

Arguments

drugAct a matrix of drug activity values (cell lines as columns, drug entries as rows)
molData a list of matricies a molecular
plots a vector of characters denoting the plots to include and the order (e.g. c("mut", "drug", "cop"). Currently, supported entries mutations (mut), drug activities (drug), copy number variations (cop)
nsc a string NSC ID that will be plotted when a "drug" entry appears in the plots vector
gene a string HUGO gene symbol for which the "mut", "cop", or "exp" plots will be produced if in plots vector
features a vector of strings that provide the full IDs for elements to be plotted (e.g. mutCDK4 for CDK4 mutations). This overwrites the nsc and gene parameters, but is needed in advanced plots that involve data that involves one-to-many relationships (e.g. many entries for a given gene in the exome data) and a gene symbol is ambiguous.
sub a vector of strings with sub-titles for each plot
xLimits a 2 number vector with the the minimum and maximum X-axis values (default: -3.3 for Z-scores, 0.1 for binary entries)
xLabel a string for the default X-axis label
extraPlot a list containing title, label, and values (numeric vector of length 60); only one extra plot can be included
verbose a boolean to show debugging information
plotDrugActivityRepeats

Value

None

Author(s)

Augustin Luna <augustin AT mail.nih.gov>

Examples

drugAct <- exprs(getAct(rcellminerData::drugData))
molData <- getMolDataMatrices()
plots <- c("mut", "drug", "cop", "xai", "pro")
plotCellMiner(drugAct, molData, plots=plots, nsc="94600", gene="CDK4", verbose=TRUE)

plots <- c("mut", "xai", "cop", "cop", "cop", "cop")
plotCellMiner(drugAct, molData, plots=plots, nsc="94600", gene=c("CDK4", "TP53", "BRAF", "GAPDH"), verbose=TRUE)

plotCellMiner(drugAct, molData, plots=plots, nsc=NULL, features=c("mutCDK4", "xaiCDK4", "exochr1:101704532_G_T", "mdais_p53_mut", "mirhsa-miR-22", "proTP53_26_GBL00064"), verbose=TRUE)

plotDrugActivityRepeats

Plot NCI-60 drug activity profiles for repeat experiments.

Description

Plot NCI-60 drug activity profiles for repeat experiments.

Usage

plotDrugActivityRepeats(nscStr, useZScore = FALSE, maxRepNum = 5,
pdfFilename = NULL, pdfWidth = 12, pdfHeight = 6)

Arguments

nscStr a string specifying the NSC identifier for a compound.
useZScore a boolean specifying whether to plot z-transformed data (as opposed to -logGI50 values).
maxRepNum an integer specifying the maximum number of repeat experiments to plot.
pdfFilename name of a PDF output
pdfWidth with of the PDF (default: 12)
pdfHeight with of the PDF (default: 6)

Value

NONE

Examples

plotDrugActivityRepeats("609699")
plotDrugActivityRepeats("609699", useZScore=TRUE, maxRepNum=3)
plotDrugSets

**Description**

Produces a barplot of the average values for a set of NSCs with a error bar (one standard deviation)

**Usage**

```r
plotDrugSets(drugAct, drugs, mainLabel = "", pdfFilename = NULL, statistic = "mean")
```

**Arguments**

- `drugAct`: a matrix of drug activity values (cell lines as columns, drug entries as rows)
- `drugs`: a vector of NSC IDs whose values will be averaged by cell line
- `mainLabel`: a main label for the plot
- `pdfFilename`: a string file name for a PDF plot, no file output will be produced if this is not provided
- `statistic`: a string, either 'mean' or 'median' (Default: mean)

**Value**

no values are returned

**Examples**

```r
drugAct <- exprs(getAct(rcellminerData::drugData))
drugs <- rownames(drugAct)[1:8]
plotDrugSets(drugAct, drugs, "Test")
```

plotStructures

**Description**

Plot Structures

**Usage**

```r
plotStructures(ids, smiles, titleCex = 1, structSize = 300, structAnnotPos = 50, mainLabel = "", rows = 1, cols = length(ids))
```
Arguments

ids a vector of strings of IDs used as structure titles
smiles a vector of strings where the strings are SMILES structures
titleCex a number, the scaling factor for the title (default: 1)
structSize a number, the size of the structure image (default: 200)
structAnnotPos a number, how far above the structure to display the title (default: 50)
mainLabel a string, the main plot label
rows number of rows in figure (default: 1)
cols number of columns in figure (default: input structures number)

Details

The parameter ids is used as a title, this function does not search for IDs, but works based off the smiles given. This is a wrapper around rcdkplot for plotting multiple structures.

Value

the function does not return anything

Author(s)

Augustin Luna <augustin AT mail.nih.gov>

Examples

drugAnnot <- as(featureData(getAct(rcellminerData::drugData)), "data.frame")
plotStructures("94600", drugAnnot["94600","SMILES"])
plotStructures(c("609699", "94600"), drugAnnot[c("609699", "94600"),"SMILES"],
mainLabel=c("609699", "94600"))

plotStructuresFromNscs

Plot the structures for NSCs

Description

Plot the structures for NSCs

Usage

plotStructuresFromNscs(nscs)

Arguments

nscs a vector of strings of NSCs used as structure titles

Details

This is a wrapper for the plotStructures() function that takes only NSCs
Value

the function does not return anything

Author(s)

Augustin Luna <augustin AT mail.nih.gov>

Examples

plotStructuresFromNscs("94600")
plotStructuresFromNscs(c("609699", "94600"))

description

Plot molecules in R plot window instead of separate Java window

Usage

rcdkplot(molecule, width = 300, height = 300, marg = 0, main = "")

Arguments

molecule an RCDK molecule
width an integer width of the molecule
height an integer height of the molecule
marg margin for all side of the plot (default: 0)
main a string the main title of the figure

details


Value

None

Examples

tmp <- parse.smiles("C1=CN(N=C1N)C2C(C(C(O2)CO)O)(F)F")
rcdkplot(tmp[[1]], width=300, height=300, main="Gemcitabine")
removeMolDataType

Remove molecular data type prefixes from features.

Description
Remove molecular data type prefixes from features.

Usage
removeMolDataType(features, prefixLen = 3)

Arguments
- features: A vector of features.
- prefixLen: The length of the molecular data type prefix.

Details
This function is primarily used to remove prefixes from elastic net features.

Value
A named vector of features without molecular data type prefixes.

Examples
removeMolDataType(c("expTP53", "copMDM2", "mutCHEK2", "mutBRAF"))

restrictFeatureMat

Restricts a feature matrix to only include features associated with a specified gene set.

Description
Restricts a feature matrix to only include features associated with a specified gene set.

Usage
restrictFeatureMat(geneSet, featureMat, prefixSet = c("cop", "exp", "mut"))

Arguments
- featureMat: a matrix or data frame with feature vectors along rows and feature names specified in rownames(featureMat).
- prefixSet: a set of feature name prefixes to be prepended to each element of geneSet to obtain a collection of geneSet-associated features.
Value

A matrix containing the features in the intersection of rownames(featureMat) and the set of geneSet-derived features (obtained by prepending each element of prefixSet to each gene in geneSet).

Examples

```r
X <- matrix(1:25, nrow=5)
rownames(X) <- c("expA", "expB", "copC", "mutC", "expD")
restrictFeatureMat(geneSet = c("B", "C"), X)
```

Description

Correlation between ith row of x and ith row of y for all i

Usage

```r
rowCors(X, Y)
```

Arguments

- **X**: a matrix
- **Y**: a matrix

Value

A list of two vectors: cor (correlation values) and pval (correlation p-values)

Author(s)

Sudhir Varma, NCI-LMP

Examples

```r
a <- matrix(runif(100), nrow=10, ncol=10)
b <- matrix(runif(100), nrow=10, ncol=10)
c <- rowCors(a, b)
```
runShinyApp

---

runShinyApp | Run Shiny App

**Description**

Run Shiny App

**Usage**

```
runShinyApp(app = "shinyComparePlots", launch.browser = TRUE, port = 3838)
```

**Arguments**

- `app`: string Shiny app to run (See Details, OPTIONS: shinyComparePlots, shinyCompareStructures, shinyReprodPlots, DEFAULT: shinyComparePlots)
- `launch.browser`: launch in browser? (default: TRUE)
- `port`: port number to use

**Details**

- shinyComparePlots: The "Comparison" application allows users to plot any two variables from the CellMiner data against each other. It additionally allows users to search for compound NSC IDs using names and mechanisms of action.
- shinyCompareStructures: The "Compound Browser" application allows users to see information about each compound, including structures and any repeat assay information.
- shinyReprodPlots: The "Structure Comparison" application allows users to identify similar compounds within the dataset either by NSC ID or SMILES string.

**Value**

None

**Examples**

```
# Uncomment first
#runShinyApp()
```

---

runShinyComparePlots | Run the Compare Plots Shiny App

**Description**

Run the Compare Plots Shiny App

**Usage**

```
runShinyComparePlots(launch.browser = TRUE, port = 3838)
```
runShinyCompareStructures

Run the Compare Structures Shiny App

Description

Run the Compare Structures Shiny App

Usage

runShinyCompareStructures(launch.browser = TRUE, port = 3838)

Arguments

launch.browser  launch in browser? (default: TRUE)
port           port number to use

Value

None

Examples

port <- 3838  
# Uncomment first  
#runShinyCompareStructures(port=port)
## runShinyCompoundBrowser

**Run the Compound Browser**

### Description

Run the Compound Browser

### Usage

```r
runShinyCompoundBrowser(launch.browser = TRUE, port = 3838)
```

### Arguments

- `launch.browser` launch in browser? (default: TRUE)
- `port` port number to use

### Value

None

### Examples

```r
port <- 3838
# Uncomment first
#runShinyCompoundBrowser(port=port)
```

## searchForNscs

**Search for NSCs**

### Description

Search for NSCs

### Usage

```r
searchForNscs(pattern)
```

### Arguments

- `pattern` a search pattern. This string will be treated as a regular expression with the case ignored.

### Details

Use this function with caution. Not all compounds have names and compounds can have many synonyms not included in CellMiner.

### Value

A vector of matching NSCs
**selectCorrelatedRows**

Select features that are correlated with a given feature (or one or more features from a set of features).

**Description**

Select features that are correlated with a given feature (or one or more features from a set of features).

**Usage**

`selectCorrelatedRows(Y, X, corThreshold = 0.1, useAbsCor = TRUE)`

**Arguments**

- **Y**: a vector or matrix; rows from X will be correlated with Y if Y is a vector or with rows of Y, if Y is a matrix.
- **X**: a matrix of values that will be compared with Y (vector) or rows of Y (matrix)
- **corThreshold**: the minimum correlation threshold for the row to be returned
- **useAbsCor**: a logical value indicating whether absolute correlations should be used (default=TRUE).

**Value**

a matrix of rows of X correlated with Y (if Y is a vector) or correlated with at least one row of Y if Y is a matrix or data frame.

**Examples**

```r
vec <- runif(10)
mat <- matrix(runif(100), 10, 10)
selectCorrelatedRows(vec, mat)
```

**selectCorrelatedRowsFromMatrices**

Select features that are correlated with a given feature (or one or more features from a set of features), merging results from multiple candidate feature matrices.

**Description**

Select features that are correlated with a given feature (or one or more features from a set of features), merging results from multiple candidate feature matrices.
Usage

selectCorrelatedRowsFromMatrices(Y, XList, corThreshold = 0.1,
   useAbsCor = TRUE)

Arguments

Y      a vector or matrix; rows from each matrix element of X will be correlated with
       Y if Y is a vector or with rows of Y, if Y is a matrix.
XList  a list of matrices whose rows will be correlated with Y (vector) or rows of Y
       (matrix). The rownames in each matrix element of XList must be specified to
       values that are unique with respect to the total set of rownames (as derived from
       all matrices in XList).
corThreshold the minimum correlation threshold for the row to be returned
useAbsCor   a logical value indicating whether absolute correlations should be used (de-
            fault=TRUE).

Value

a matrix formed from rows of matrices in XList that are correlated with Y (if Y is a vector) or
 correlated with at least one row of Y if Y is a matrix or data frame.

Examples

vec <- runif(10)
names(vec) <- 1:10
matList <- list(X1 = matrix(runif(100), 10, 10), X2 = matrix(runif(100), 10, 10))
rownames(matList$X1) <- paste0("X1_row", 1:10)
colnames(matList$X1) <- paste0("X1_col", 1:10)
rownames(matList$X2) <- paste0("X2_row", 1:10)
colnames(matList$X2) <- paste0("X2_col", 1:10)
selectCorrelatedRowsFromMatrices(vec, matList)

[[,MolData-method

Returns an indexed eSet object from a MolData object eSet list.

Description

Returns an indexed eSet object from a MolData object eSet list.

Usage

## S4 method for signature 'MolData'
x[[i]]

Arguments

x      A MolData object.
i      Index or named item in MolData object eSet list.

Value

An indexed eSet object from a MolData object eSet list.
[[<-,MolData-method

Assigns an eSet object to a specified position in a MolData object eSet list.

Description

Assigns an eSet object to a specified position in a MolData object eSet list.

Usage

## S4 replacement method for signature 'MolData'
x[[i]] <- value

Arguments

x A MolData object.
i Index or named item in MolData object eSet list.
value An eSet object to be assigned.

Value

An eSet object to a specified position in a MolData object eSet list.
Index

*Topic data*

- `cmVersion`, 3
- `Drug_MOA_Key`, 8
- `drugDB`, 7
- `elNetMolDataNCI60`, 8
- `fingerprintList`, 9
- `.DrugData(DrugData-class).`, 7
- `.MolData(MolData-class).`, 32
- `[[,MolData-method`, 45
- `[[-,MolData-method`, 46
- `cmVersion`, 3
- `compareFingerprints`, 4
- `crossCors`, 4
- `crossCorsSpearman`, 5
- `Drug_MOA_Key`, 8
- `DrugData`, 6
- `DrugData,eSet,eSet,MIAxE-method`, 6
- `DrugData-class`, 7
- `drugDB`, 7
- `elNetMolDataNCI60`, 8
- `fingerprintList`, 9
- `getAct`, 9
- `getAct,DrugData-method`, 10
- `getActivityRangeStats`, 10
- `getAllFeatureData`, 11
- `getAllFeatureData,MolData-method`, 11
- `getBinaryMutationData`, 12
- `getColumnQuantiles`, 13
- `getDrugActivityData`, 13
- `getDrugActivityRange`, 14
- `getDrugActivityRepeatData`, 14
- `getDrugMoaList`, 15
- `getDrugName`, 16
- `getESetList`, 16
- `getESetList,MolData-method`, 17
- `getFeatureAnnot`, 17
- `getFeatureAnnot,DrugData-method`, 18
- `getFeatureAnnot,MolData-method`, 18
- `getFeatureDataFromMatList`, 19
- `getFingerprintList`, 19
- `getMedSenLineActivity`, 20
- `getMinDrugActivityRepeatCor`, 21
- `getMoaStr`, 21
- `getMoaToCompounds`, 22
- `getMolDataMatrices`, 22
- `getMolDataType`, 23
- `getNumDrugActivityRepeats`, 23
- `getNumMissingLines`, 24
- `getRepeatAct`, 24
- `getRepeatAct,DrugData-method`, 25
- `getRepeatAct,DrugData-method`, 25
- `getRsd`, 25
- `getSampleData`, 26
- `getSampleData,DrugData-method`, 26
- `getSampleData,MolData-method`, 27
- `getSmiles`, 27
- `hasMoa`, 28
- `initialize,DrugData-method`, 28
- `initialize,MolData-method`, 29
- `isPublic`, 29
- `loadCellminerPlotInfo`, 30
- `loadNciColorSet`, 30
- `MolData`, 31
- `MolData,list,MIAxE-method`, 31
- `MolData-class`, 32
- `passRuleOf5`, 32
- `passRuleOf5FromNsc`, 33
- `patternComparison`, 33
- `plotCellMiner`, 34
- `plotDrugActivityRepeats`, 35
- `plotDrugSets`, 36
- `plotStructures`, 36
- `plotStructuresFromNscs`, 37
- `rcdkplot`, 38
- `removeMolDataType`, 39
- `restrictFeatureMat`, 39
- `rowCors`, 40
- `runShinyApp`, 41
- `runShinyComparePlots`, 41
- `runShinyCompareStructures`, 42
runShinyCompoundBrowser, 43
searchForNscs, 43
selectCorrelatedRows, 44
selectCorrelatedRowsFromMatrices, 44