Package 'Xeva'

August 27, 2018

Type Package

Title Analysis of patient-derived xenograft (PDX) data

Version 1.0.0

Author Arvind Mer, Benjamin Haibe-Kains

Maintainer Arvind Mer <amer@uhnresearch.ca>

Description Contains set of functions to perform analysis of patient-derived xenograft (PDX) data.

License Artistic-2.0

RoxygenNote 6.0.1

VignetteBuilder knitr

Suggests knitr

Imports methods,
BBmisc,
Biobase,
plyr,
stringr,
ggplot2,
ComplexHeatmap,
reshape2,
grDevices,
PharmacoGx,
foreach,
parallel,
doParallel,
doSNOW,
acepack,
mgcv

R topics documented:

ABC *compute area between two curves compute area between two time-volume curves*

Description

compute area between two curves compute area between two time-volume curves

Usage

```
ABC(contr.time = NULL, contr.volume = NULL, treat.time = NULL,
  treat.volume = NULL)
```

Arguments

contr.time	time vector for control
contr.volume	volume vector for control
treat.time	time vector for treatment
treat.volume	volume vector for treatment
degree	default TRUE will give angle in Degree and FALSE will return Radians

Value

returns batch response object

Examples

```
contr.time <- treat.time  <- c(0, 3, 7, 11, 18, 22, 26, 30, 32, 35)
contr.volume<- contr.time * tan(60*pi/180)
treat.volume<- treat.time * tan(15*pi/180)
abc <- ABC(contr.time, contr.volume, treat.time, treat.volume)
par(pty="s")
xylimit <- range(c(contr.time, contr.volume, treat.time, treat.volume))
plot(contr.time, contr.volume, type = "b", xlim = xylimit, ylim = xylimit)
lines(treat.time, treat.volume, type = "b")
polygon(c(treat.time, rev(treat.time)), c(contr.volume, rev(treat.volume)), col = "#fa9fb5", border = NA)
```

addExperimentalDesign *Add a new experimental design*

Description

Add a new experimental design in expDesign slot.

Usage

```
addExperimentalDesign(object, treatment, control = NULL, batch.id = NULL,
  replace = FALSE)
```

Arguments

object	The Xeva dataset
treatment	The model.id of treatment
control	The model.id of control
batch.id	The batch.id for new batch
replace	If TRUE will replace the old batch with new values

Value

returns Xeva dataset with new experimental design added

Examples

```
data(pdxe)
pdxe = addExperimentalDesign(object=pdxe, treatment= c("X.010.BG98"), control=c("X.010.uned"),
                             batch.id="new.batch", replace=FALSE)
```

angle *compute angle computes angle between two time-volume curves*

Description

compute angle computes angle between two time-volume curves

Usage

```
angle(contr.time = NULL, contr.volume = NULL, treat.time = NULL,
  treat.volume = NULL, degree = TRUE)
```

Arguments

contr.time	time vector for control
contr.volume	volume vector for control
treat.time	time vector for treatment
treat.volume	volume vector for treatment
degree	default TRUE will give angle in Degree and FALSE will return Radians

Value

returns batch response object

Examples

```
contr.time <- treat.time  <- c(0, 3, 7, 11, 18, 22, 26, 30, 32, 35)
contr.volume<- contr.time * tan(60*pi/180)
treat.volume<- treat.time * tan(15*pi/180)
ang <- angle(contr.time, contr.volume, treat.time, treat.volume)
par(pty="s")
xylimit <- range(c(contr.time, contr.volume, treat.time, treat.volume))
plot(contr.time, contr.volume, type = "b", xlim = xylimit, ylim = xylimit)
lines(treat.time, treat.volume, type = "b")
abline(lm(contr.volume~contr.time))
abline(lm(treat.volume~treat.time))
```

AUC *AUC returns area under the curve*

Description

AUC returns area under the curve

Usage

```
AUC(time, volume)
```

Arguments

time	vector of time
volume	first vector of volume

Value

returns angle and slope object

Examples

```
time  <- c(0, 3, 7, 11, 18, 22, 26, 30, 32, 35)
volume1<- time * tan(30*pi/180)
volume2<- time * tan(45*pi/180)
auc1 <- AUC(time, volume1)
auc2 <- AUC(time, volume2)
par(pty="s")
xylimit <- range(c(time, volume1, volume2))
plot(time, volume1, type = "b", xlim = xylimit, ylim = xylimit)
lines(time, volume2, type = "b")
abline(lm(volume1~time))
abline(lm(volume2~time))
```

---

batchNames *Get batch names/ids*

Description

Get all batch names/ids from a Xeva dataset

Usage

```
batchNames(object)
```

Arguments

object	The XevaSet to replace drug info in

Value

A Vector with all batch.name

Examples

```
data(pdxe)
batchNames(pdxe)
```

brca.pdxe *PDXE breast cancer dataset*

Description

PDXE breast cancer dataset

Usage

```
data(brca.pdxe)
```

Format

A Xeva object of PDXE breast cancer dataset

creatXevaSet *Creat Xeva class object* creatXevaSet *returns Xeva class object*

Description

Creat Xeva class object creatXevaSet returns Xeva class object

Usage

```
creatXevaSet(name, model = data.frame(), drug = data.frame(),
  experiment = data.frame(), expDesign = list(),
  modelSensitivity = data.frame(), batchSensitivity = data.frame(),
  molecularProfiles = list(), modToBiobaseMap = data.frame())
```

Arguments

name	a character string detailing the name of the dataset
model	a data.frame containg the annotations for all models used in the experiment
drug	a data.frame containg the annotations for all the drugs profiled in the data set, across all data types
experiment	a data.frame containg all experiment information
molecularProfiles	a list of ExpressionSet objects containing different molecular profiles

Value

Returns Xeva object

Examples

```
geoExp <- readRDS("DATA-raw/Geo_Exp.Rda")
pdxe <- creatPharmacoPxSet(name = "PDXE",
                           molecularProfiles = list(RNASeq = geoExp$RNASeq),
                           experiment = geoExp$experiment,
                           model = geoExp$model,
                           drug  = geoExp$drug )
save(pdxe, file = "data/pdxe.rda")
data("pdxe")
```

`drugInfo` *get drug information get drug information slot*

Description

get drug information get drug information slot

Usage

```
drugInfo(object)
```

Arguments

`object`	The `XevaSet` to retrieve drug info from

Value

a `data.frame` with the drug annotations

Examples

```
data(pdxe)
drugInfo(pdxe)
```

`drugInfo<-` *set drug information set drug information slot*

Description

set drug information set drug information slot

Usage

```
drugInfo(object) <- value
```

Arguments

object	The XevaSet to replace drug info in
value	A data.frame with the new drug annotations

Value

Updated XevaSet

Examples

```
data(pdxe)
drugInfo(pdxe) <- drugInfo(pdxe)
```

---

drugSensitivitySig *drugSensitivitySig*

Description

Given a Xeva object, and drug name it will return sensitivity value for all the genes/fetures

Usage

```
drugSensitivitySig(object, drug, mDataType = NULL, molData = NULL,
  features = NULL, model.ids = NULL, model2bidMap = NULL,
  sensitivity.measure = "slope", fit = c("lm", "maxCor", "gam"),
  standardize = c("SD", "rescale", "none"), nthread = 1, tissue = NULL,
  verbose = TRUE)
```

Arguments

object	The Xeva dataset
drug	Name of the drug
mDataType	molecular data type
molData	External data matrix. Rows as features and columns as samples
features	which molecular data fetures to use. Default NULL will use all fetures
model.ids	which model.id to use from the dataset. Default NULL will use all model.id
model2bidMap	a datafram with model.id and biobase.id. Default NULL will use internal mapping
sensitivity.measure	Name of the sensitivity measure
fit	Default lm. Name of the model to be fitted. Options are "lm", "maxCor", "gam"
type	Tissue type. Default is NULL which will use 'tissue' from object

Details

A matrix of values can be directly passed to molData. fit can be "lm", "maxCor" or "gam". In case where a model.id map to multipal biobase.id the first biobase.id in the datafram will be used.

Value

A datafram with fetures and values

Examples

```
data(pdxe)
## select BRCA samples
mid <- modelInfo(pdxe)[modelInfo(pdxe)$tissue=="BRCA", ]
senSig <- drugSensitivitySig(object=pdxe, drug="tamoxifen",
                             mDataType="RNASeq", features=1:5,
                             model.ids = mid$model.id,
                             sensitivity.measure="slope", fit = "lm")
```

---

expDesign *Given a batch.name get batch*

Description

Given a batch.name get batch from a Xeva dataset

Usage

```
expDesign(object, batch.name)
```

Arguments

object	The XevaSet
object	The batch.name

Value

A Vector with all batch.name

Examples

```
data(pdxe)
expDesign(pdxe, batch.name = "X-6047.paclitaxel")
```

expDesignInfo *expDesignInfo Generic Generic for expDesignInfo method*

Description

expDesignInfo Generic Generic for expDesignInfo method

Usage

```
expDesignInfo(object)
```

Arguments

object	The XevaSet to retrieve drug info from

Value

a list with the all experiment designs

Examples

```
data(pdxe)
expDesignInfo(pdxe)
```

expDesignInfo<- *expDesignInfo<- Generic Generic for expDesignInfo replace method*

Description

expDesignInfo<- Generic Generic for expDesignInfo replace method

Usage

```
expDesignInfo(object) <- value
```

Arguments

object	The XevaSet to replace drug info in
value	A list with the experiment designs

Value

Updated XevaSet

Examples

```
data(pdxe)
expDesignInfo(pdxe) <- expDesignInfo(pdxe)
```

geneSensitivityPlot *geneSensitivityPlot*

Description

Given a Xeva object, feture name and drug name it will plot feture values against sensitivity value

Usage

```
geneSensitivityPlot(object, mDataType, feature, drug,
  sensitivity.measure = "slope", standardize = c("SD", "rescale", "log",
  "none"))
```

Details

Plot a gene expression against sensitivity signatures for a drug

Examples

```
data(pdxe)
geneSensitivityPlot(object=pdxe, mDataType="RNASeq", feature="A1BG", drug="binimetinib",
sensitivity.measure="slope", standardize="log")
```

getBatchName *Get batch.name for a given model.id*

Description

Get batch.name for a given model.id. If no batch.name found it will return NULL

Usage

```
getBatchName(object, model.id)
```

Arguments

object	The Xeva dataset
model.id	The model.id for which batch name required

Value

a vector with all batch names

Examples

```
data(pdxe)
# extract batch.name for a given model.id
getBatchName(object=pdxe, model.id="X.1655.uned")
getBatchName(object=pdxe, model.id="X.010.fiab")
```

`getControls` *Get controls for a given model.id*

Description

Get controls for a given model.id. If no control found it will return NULL

Usage

```
getControls(object, model.id)
```

Arguments

`object`	The `Xeva` dataset
`model.id`	The `model.id`

Value

a `vector` with control model.id

Examples

```
data(pdxe)
# extract controls for a given model.id
getControls(object=pdxe, model.id="X.1655.LE11.biib")
# if no control found it will return NULL
getControls(object=pdxe, model.id="X.1655.uned")
```

`getExpDesignDF` *Given a model.id it will return a data.fram of experiemt design with columns as "treatment", "control", "batch.name"*

Description

Given a model.id it will return a data.fram of experiemt design with columns as "treatment", "control", "batch.name"

Usage

```
getExpDesignDF(object, model.id)
```

Arguments

object	The Xeva dataset
model.id	The model.id

Value

a data.fram with treatment, control and batch.name

Examples

```
data(pdxe)
# This will give a data.fram with columns as "treatment", "control", "batch.name"
getExpDesignDF(object=pdxe, model.id="X.1655.LE11.biib")
```

---

getExperiment *For a given model.id, it will return a data.fram containing all data stored in experiment slot*

Description

For a given model.id, it will return a data.fram containing all data stored in experiment slot

Usage

```
getExperiment(object, model.id = NULL, batchName = NULL, expDig = NULL,
  treatment.only = FALSE, max.time = NULL, vol.normal = FALSE,
  return.list = FALSE, impute.value = FALSE, concurrent.time = FALSE)
```

Arguments

object	The XevaSet
model.id	The model.id for which data is required, multipal allowed
batchName	batch name from the Xeva set
expDig	Experiment design
treatment.only	Default FALSE. If TRUE give data only for non-zero dose periode (if dose data avalible)
max.time	maximum time for data
vol.normal	default TRUE will use
return.list	default FALSE will return a datafram
impute.value	default FALSE. If TRUE will impute the values
concurrent.time	default FALSE. If TRUE will cut the batch data such that control and treatment will end at same time point

Value

a data.fram will all the the values stored in experiment slot

Examples

```
data(pdxe)
getExperiment(pdxe, model.id="X.6047.uned", treatment.only=TRUE)
getExperiment(pdxe, model.id=c("X.6047.uned", "X.6047.pael"), treatment.only=TRUE)
getExperiment(pdxe, batchName="X-6047.paclitaxel", treatment.only=TRUE)
expDesign <- list(batch.name="myBatch", treatment=c("X.050.LE11","X.050.LE11.evus"), control=c("X.050.uned"))
getExperiment(pdxe, expDig=expDesign)
```

---

getMolecularProfiles *Get Molecular Profiles*

Description

Get Molecular Profiles

Usage

```
getMolecularProfiles(object, data.type)
```

Arguments

object	The XevaSet
data.type	character, which one of the molecular data types is needed

Value

a ExpressionSet where sample names are biobase.id of model

Examples

```
data(pdxe)
pdxe_RNA <- getMolecularProfiles(pdxe, data.type="RNASeq")
```

getTreatment *Get treatment for a given model.id*

Description

Get treatment for a given model.id. If no treatment found it will return NULL

Usage

```
getTreatment(object, model.id)
```

Arguments

object	The Xeva dataset
model.id	The model.id

Value

a vector with treatment model.id

Examples

```
data(pdxe)
# extract treatment model.id for a given model.id
getTreatment(object=pdxe, model.id="X.1655.uned")
```

mapModelSlotIds *Map ids of model slot*

Description

Map one id type to another in model slot. For example map a model.id to patient.id

Usage

```
mapModelSlotIds(object, id, id.name, map.to = "all", unique = TRUE)
```

Arguments

object	The Xeva dataset
id	The id
id.name	The id name
map.to	The name of the mapped id. Default all
unique	Default TRUE. If unique=FALSE output will be mapped to input

Value

a data.fram with id and mapped id

Examples

```
data(pdxe)
mapModelSlotIds(object=pdxe, id="X-007", id.name="patient.id", map.to="model.id")
##map batch ids
mapModelSlotIds(pdxe, id= "X-011.INC280", id.name = "batch.name", map.to = "tissue")
```

---

model2BiobaseIdMap *model2BiobaseIdMap Gives model.id to biobase.id mapping datafram*

Description

model2BiobaseIdMap Gives model.id to biobase.id mapping datafram

Usage

```
model2BiobaseIdMap(object, mDataType = NULL)
```

Arguments

object	The XevaSet
mDataType	Data type for which ids to be retrive. Default NULL will return full datafram

Value

a data.frame with the model.id and biobase.id

Examples

```
data(pdxe)
idMap <- model2BiobaseIdMap(pdxe, mDataType="RNASeq")
head(idMap)
```

modelInfo *modelInfo Generic Generic for modelInfo method*

Description

modelInfo Generic Generic for modelInfo method

Usage

```
modelInfo(object, mDataType = NULL)
```

Arguments

object	The XevaSet to retrieve drug info from

Value

a data.frame with the model annotations

Examples

```
data(pdxe)
mid <- modelInfo(pdxe)
head(mid)
```

modelInfo<- *modelInfo<- Generic Generic for modelInfo replace method*

Description

modelInfo<- Generic Generic for modelInfo replace method

Usage

```
modelInfo(object) <- value
```

Arguments

object	The XevaSet to replace drug info in
value	A data.frame with the new model annotations

Value

Updated XevaSet

Examples

```
data(pdxe)
modelInfo(pdxe) <- modelInfo(pdxe)
```

mRECIST *Computes the mRECIST*

Description

mRECIST returns the mRECIST for given volume response

Usage

```
mRECIST(time, volume, min.time = 10, return.detail = FALSE)
```

Arguments

time	Value of best response
volume	Value of best average response
min.time	minimum time after which tumore volume will be considered
return.detail	default FALSE. If TRUE will return all intermediate values

Value

Returns the mRECIST

Examples

```
time  <- c(0, 3, 7, 11, 18, 22, 26, 30, 32, 35)
volume<- c(250.8, 320.4, 402.3, 382.6, 384, 445.9, 460.2, 546.8, 554.3, 617.9)
mRECIST(time, volume, min.time=10, return.detail=FALSE)
```

pdxe *Example dataset with 1x1x1 experiment design*

Description

This is PDXE dataset without microarray data.

Usage

```
data(pdxe)
```

Format

A Xeva object with 1x1x1 experiment design and moleculer data

PDXMI *PDX-MI data*

Description

A dataset containing PDX models minimal information (PDX-MI) standard and corresponding Xeva variable.

Usage

```
data(PDXMI)
```

Format

An object of class `data.frame` with 45 rows and 4 columns.

Details

For details about PDX-MI see:

Meehan, Terrence F., et al. "PDX-MI: minimal information for patient-derived tumor xenograft models." Cancer research 77.21 (2017): e62-e66.

Source

http://cancerres.aacrjournals.org/lookup/doi/10.1158/0008-5472.CAN-17-0582

PDX_MI *PDX-MI: Minimal Information for PDX*

Description

PDX-MI: Minimal Information for PDX

Usage

```
PDX_MI(object)
```

plotBatch *Plot batch data*

Description

Plot data for a batch id or experiment design

Usage

```
plotBatch(object, batchName = NULL, expDig = NULL, max.time = NULL,
  treatment.only = FALSE, vol.normal = FALSE, impute.value = TRUE,
  concurrent.time = FALSE, control.col = "#6baed6",
  treatment.col = "#fc8d59", title = "", xlab = "Time", ylab = "Volume",
  log.y = FALSE, drug.name = NULL, SE.plot = c("all", "none", "errorbar",
  "ribbon"), aspect.ratio = c(1, NULL), minor.line.size = 0.5,
  major.line.size = 0.7)
```

Arguments

object	Xeva object
batchName	batch name
expDig	Experiment design list
max.time	maximum time point of the plot, default NULL will plot complete data
treatment.only	default FALSE. Given full data treatment.only=TRUE will plot data only during treatment
vol.normal	default FALSE . If TRUE volume will ne normalised
impute.value	default TRUE, will impute values where missing
control.col	color for control plots
treatment.col	color for treatment plots
title	title of the plot
xlab	title of x axis
ylab	title of y axis
log.y	default FALSE, if TRUE y axis will be in log
drug.name	default NULL will extract drug name from data
SE.plot	plot type. Default "all" will plot all plots and average curves. Possible values are "all", "none", "errorbar", "ribbon"
aspect.ratio	default 1 will create equeal width and height plot
minor.line.size	line size for minor lines default 0.5
major.line.size	line size for major lines default 0.7

Value

A ggplot2 plot with control and treatment

Examples

```
data(pdxe)
plt <- plotBatch(pdxe, batchName="X-1228.CKX620", vol.normal=TRUE)
expDesign <- list(batch.name="myBatch", treatment=c("X.1228.LC61.pael","X.1228.pael"), control=c("X.1228.uned")
plotBatch(pdxe, expDig=expDesign, vol.normal=T)
plotBatch(pdxe, expDig=expDesign, vol.normal=F, SE.plot = "errorbar")
```

---

plotmRECIST *To plot mRECIST values*

Description

plotmRECIST plots the mRECIST matrix obtained from summarizeResponse

Usage

```
plotmRECIST(mat, control.name = NA, control.col = "#238b45",
  drug.col = "black", colPalette = NULL, name = "Drug & Models",
  sort = TRUE, row_fontsize = 12, col_fontsize = 12, draw_plot = TRUE)
```

Arguments

object	The Xeva dataset
model.id	The model.id

Value

plot

Examples

```
data(pdxe)
## select lung cancer pdxe data
pdxe.lung <- summarizeResponse(pdxe, response.measure = "mRECIST",
                               group.by="patient.id", tissue="NSCLC")
plotmRECIST(pdxe.lung, control.name = "untreated")
```

print.XevaSet *print Xeva object*

Description

print displays Xeva object information or model or batch information

Usage

```
## S3 method for class 'XevaSet'
print(object, id = NULL)
```

Arguments

object	Xeva object
id	default NULL, id can be model.id or batch.name

Value

Prints object, model or batch information.

Examples

```
data(pdxe)
# to print object information
print(pdxe)

# to print a model
model.id = modelInfo(pdxe)$model.id[1]
print(pdxe, id = model.id)

# to print a batch
batch.id = batchNames(pdxe)[1]
print(pdxe, id = batch.id)
```

response response *computes response of a PDX model or batch*

Description

response computes response of a PDX model or batch

Usage

```
response(object, model.id = NULL, batchName = NULL, expDig = NULL,
  res.measure = c("angle", "mRECIST", "AUC", "angle", "abc"),
  treatment.only = TRUE, max.time = NULL, impute.value = TRUE,
  min.time = 10, concurrent.time = TRUE, vol.normal = F, verbose = TRUE)
```

Arguments

object	Xeva object
model.id	model id for which response to be computed
batchName	batch id for which response to be computed
expDig	experiment design for which response to be computed
res.measure	response measure
treatment.only	Default FALSE. If TRUE give data only for non-zero dose periode (if dose data avalible)
max.time	maximum time for data
impute.value	default FALSE. If TRUE will impute the values
min.time	default **10** days. Used for *mRECIST* computation
concurrent.time	default FALSE. If TRUE will cut the batch data such that control and treatment will end at same time point
vol.normal	default TRUE will use
verbose	default TRUE will print infromation

Value

returns model or batch response object

Examples

```
data(pdxe)
response(pdxe, model.id="X.1270.LK36", res.measure="mRECIST")
response(pdxe, model.id="X.1270.LK36", res.measure="AUC")

response(pdxe, batchName="X-1270.HDM201", res.measure="angle")
ed <- list(batch.name="b1", treatment=c("X.1228.LC61.pael","X.1228.pael"), control=c("X.1228.uned"))
response(pdxe, expDig=ed, res.measure="angle")
```

---

selectModelIds *To select model ids based on drug name and/or tissue*

Description

To select model ids based on drug name and/or tissue

Usage

```
selectModelIds(object, drug = NULL, drug.match.exact = TRUE,
  tissue = NULL)
```

Arguments

object	The XevaSet
drug	Name of the drug
drug.match.exact	Default TRUE
tissue	Tumor type. Default NULL

Value

a vector with the matched model.ids

Examples

```
data(pdxe)
selectModelIds(pdxe, drug="paclitaxel", drug.match.exact=TRUE, tissue="BRCA")
```

---

sensitivity *Get sensitivity for an Xeva object*

Description

Given a Xeva object, it will return sensitivity datafram

Usage

```
sensitivity(object, type = c("model", "batch"), sensitivity.measure = NULL)
```

Arguments

object	The Xeva dataset
type	sensitivity type (either model or batch)
sensitivity.measure	Name of the sensitivity.measure. Default NULL, will return all

Value

a data.fram with model or batch id and sensitivity values

Examples

```
data(cm.pdxe)
head(sensitivity(cm.pdxe, type="batch"))
head(sensitivity(cm.pdxe, type="model"))
```

setResponse setResponse *sets response of an Xeva object*

Description

setResponse sets response of an Xeva object

Usage

```
setResponse(object, res.measure = c("mRECIST", "slope", "AUC", "angle",
  "abc"), min.time = 10, treatment.only = TRUE, max.time = NULL,
  vol.normal = TRUE, impute.value = TRUE, concurrent.time = TRUE,
  verbose = TRUE)
```

Arguments

object	Xeva object
res.measure	response measure, multipal measure allowed
min.time	default **10** days. Used for *mRECIST* computation
treatment.only	Default FALSE. If TRUE give data only for non-zero dose periode (if dose data avalible)
max.time	maximum time for data
vol.normal	default TRUE will use
impute.value	default FALSE. If TRUE will impute the values
concurrent.time	default FALSE. If TRUE will cut the batch data such that control and treatment will end at same time point
verbose	default TRUE will print infromation

Value

returns updated Xeva object

setSensitivity *add new sensitivity in a Xeva object*

Description

This will add a add new sensitivity in a Xeva object

Usage

```
setSensitivity(object, type, name, value)
```

Arguments

object	The Xeva dataset
type	sensitivity type (either model or batch)
name	name of new sensitivity column
value	a vector of values. If vector is named, values will be filled by name. Missing values will be NA

Value

a Xeva object with updated with updated sensitivity

Examples

```
data(pdxe)
s <- sensitivity(pdxe, "model")
pdxe <- setSensitivity(pdxe, "model", "mR", s$mRECIST)
```

---

show,XevaSet-method *A method to display object for "show" setGeneric is already defined*

Description

A method to display object for "show" setGeneric is already defined

Usage

```
## S4 method for signature 'XevaSet'
show(object)
```

---

slope *Computes slope*

Description

slope returns the slope for given time and volume data

Usage

```
slope(time, volume, degree = TRUE)
```

Arguments

time	vector of time
volume	vector of volume
degree	default TRUE will give angle in Degree and FALSE will return Radians

Value

returns the slope and a fit object

Examples

```
time  <- c(0, 3, 7, 11, 18, 22, 26, 30, 32, 35)
volume<- c(250.8, 320.4, 402.3, 382.6, 384, 445.9, 460.2, 546.8, 554.3, 617.9)
sl <- slope(time, volume)
par(pty="s")
xylimit <- range(c(time, volume))
plot(time, volume, type = "b", xlim = xylimit, ylim = xylimit)
abline(lm(volume~time))
```

---

subsetXeva *Subset Xeva object*

Description

Subset Xeva object

Usage

```
subsetXeva(object, ids, id.name, keep.batch = TRUE)
```

Arguments

object	the XevaSet
ids	ids to be selected for
id.name	names of the id
keep.batch	Default is TRUE. If FALSE will remove all the other model.ids from the experiemt design that do not belong to selection

Value

New Xeva object

Examples

```
data(pdxe)
df <- subsetXeva(pdxe, ids = c("X-1008", "X-1027"), id.name="patient.id", keep.batch=TRUE)
```

summarizeMolecularProfiles
summarizeMolecularProfiles

Description

summarizeMolecularProfiles

Usage

```
summarizeMolecularProfiles(object, drug, mDataType, tissue = NULL,
  sensitivity.measure = NULL, unique.model = TRUE, batchName = NULL,
  expDig = NULL)
```

Arguments

object	The XevaSet
drug	Name of the drug
mDataType	character, which one of the molecular data types is needed
tissue	default NULL will return all across all tissue
sensitivity.measure	default NULL will return all sensitivity measure
unique.model	default TRUE will return only one sequncing id, in case where one model id mapes to several sequencing ids

Details

- If a sequencing sample belong to multipal models, summarizeMolecularProfiles will creat saperate column for each model.
- All the models without the moleculer data will be removed from the output expression set.

Value

A ExpressionSet where sample names are model.id and sensitivity measure will be present in pData

Examples

```
data(pdxe)
pacRNA <- summarizeMolecularProfiles(pdxe, drug="paclitaxel", mDataType="RNASeq",
                                     tissue= "BRCA", sensitivity.measure="mRECIST")
print(pacRNA)
```

summarizeResponse *Summarize Response of PDXs*

Description

Summarize Response of PDXs.

Usage

```
summarizeResponse(object, response.measure = "mRECIST", model.id = NULL,
  batch.id = NULL, group.by = "patient.id", summary.stat = c(";", "mean",
  "median"), tissue = NULL)
```

Arguments

object	The XevaSet
response.measure	character . Which response measure to use? Use the responseMeasures function to find out what measures are available for each Xeva set.
group.by	default patient.id. How the models should be grouped togather. See details
summary.stat	which summary method to use if multipal ids were found
batch.name	a vector of batch names. Default NULL will return all batchs

Details

There can be two types of response measure

- per model response : One response value for each Model, e.g. mRECIST_recomputed for each model
- per batch response : One response value for each Batch, e.g. angle between treatment and control groups

In case of per model response output columns will be model.id (or group.by). For per batch response group.by value can be "batch.name" .

Value

a matrix with rows as drug names, coulmn as group.by and each cell contains response.measure for the pair.

Examples

```
data(pdxe)
pdxe_mR <- summarizeResponse(pdxe, response.measure = "mRECIST", group.by="patient.id")
#to get only lung PDXE
pdxe_mR <- summarizeResponse(pdxe, response.measure = "mRECIST",
                             group.by="patient.id", tissue="NSCLC")
```

waterfall *waterfall plot creates waterfall plot for a given drug*

Description

waterfall plot creates waterfall plot for a given drug

Usage

```
waterfall(object, drug, res.measure, group.by = NULL, tissue = NULL,
  model.id = NULL, model.type = NULL, type.color = "#cc4c02",
  legend.name = NULL, yname = NULL, title = NULL, sort = TRUE)
```

Arguments

object	the XevaSet
drug	name of the drug
res.measure	PDX model response measure
group.by	group response data
tissue	tissue
model.id	which model.id to plot. Default is NULL will plot all models
model.type	type of model such as mutated or wild type
type.color	a list with colors used for each type
legend.name	name of the legend
yname	name for y axis
title	title of the plot
sort	default TRUE will sort the data

Examples

```
data(pdxe)
waterfall(pdxe, drug="binimetinib", res.measure="best.avg.response_published", tissue = "CRC")
##-- example with model.type
mut <- summarizeMolecularProfiles(pdxe,drug = "erlotinib", mDataType="mutation", tissue = "NSCLC")
model.type <- exprs(mut)["TP53", ]
waterfall(pdxe, drug="erlotinib", res.measure="best.avg.response_published",
          tissue="NSCLC", model.id=names(model.type), model.type= model.type)
```

Index