

Package ‘DGEobj’

December 7, 2020

Type Package

Title Differential Gene Expression (DGE) Analysis Results Data Object

Version 1.0.1

Description Provides a flexible container to manage and annotate Differential Gene Expression (DGE) analysis results (Smythe et. al (2015) <doi:10.1093/nar/gkv007>). The DGEobj has data slots for row (gene), col (samples), assays (matrix n-rows by m-samples dimensions) and metadata (not keyed to row, col, or assays). A set of accessory functions to deposit, query and retrieve subsets of a data workflow has been provided. Attributes are used to capture metadata such as species and gene model, including reproducibility information such that a 3rd party can access a DGEobj history to see how each data object was created or modified. Since the DGEobj is customizable and extensible it is not limited to RNA-seq analysis types of workflows -- it can accommodate nearly any data analysis workflow that starts from a matrix of assays (rows) by samples (columns).

Depends R (>= 3.5.0)

License GPL-3

Language en-US

Encoding UTF-8

Imports assertthat, magrittr, stringr, utils, GenomicRanges

RoxygenNote 7.1.1

Suggests knitr, rmarkdown, testthat, edgeR

VignetteBuilder knitr

NeedsCompilation no

Author John Thompson [aut],
Connie Brett [aut, cre],
Isaac Neuhaus [aut],
Ryan Thompson [aut]

Maintainer Connie Brett <connie@aggregate-genius.com>

Repository CRAN

Date/Publication 2020-12-07 09:30:02 UTC

R topics documented:

DGEobj-package	2
addItem	3
addItem	4
annotateDGEobj	5
as.list.DGEobj	6
baseType	6
baseTypes	7
dim.DGEobj	8
dimnames.DGEobj	8
getAttribute	9
getAttributes	9
getBaseType	10
getItem	11
getItem	11
getType	12
initDGEobj	13
inventory	14
newType	15
print.DGEobj	15
resetDGEobj	16
rmItem	17
rmItems	17
setAttribute	18
setAttributes	19
showAttributes	20
showMeta	20
showTypes	21
subset.DGEobj	22
[.DGEobj	22
Index	24

Description

DGEobj is an S3 data class that provides a flexible container for Differential Gene Expression (DGE) analysis results. The DGEobj class is designed to be extensible allowing definition of new data types as needed. A set of accessory functions to deposit, query and retrieve subsets of a data workflow has been provided. Attributes are used to capture metadata such as species and gene model, including reproducibility information such that a 3rd party can access a DGEobj history to see how each data object was created or modified.

Details

Operationally, the DGEobj is styled after the RangedSummarizedExperiment (RSE). The DGEobj has data slots for row (gene), col (samples), assays (anything with n-rows by m-samples dimensions) and metadata (anything that can't be keyed to row, col or assay). The key motivations for creating the DGEobj data structure is that the RSE only allows one data item each in the row and col slots and thus is unsuitable for capturing the plethora of data objects created during a typical DGE workflow. The DGEobj data structure can hold any number of row and col data objects and thus is engineered for capturing the multiple steps of a downstream analysis.

Certain object types, primarily the count matrix and associated row and column info, are defined as unique which means only one instance of that type may be added to the DGEobj.

When multiple objects of one type are included in a DGEobj (e.g. two different fits), the concept of parent attributes is used to associate downstream data objects (e.g. contrasts) with the appropriate data object they are derived from.

More Information

```
browseVignettes(package = 'DGEobj')
```

addItem	<i>Add a data item</i>
---------	------------------------

Description

Add a data item

Usage

```
addItem(
  dgeObj,
  item,
  itemName,
  itemType,
  funArgs = match.call(),
  itemAttr,
  parent = "",
  overwrite = FALSE,
  init = FALSE
)
```

Arguments

dgeObj	A class DGEobj created by function <code>initDGEobj()</code>
item	The data item to be deposited in the DGEobj
itemName	The assigned name for this data item
itemType	The type attribute. See <code>showTypes()</code> to see the predefined types – types are extensible with the <code>newType()</code> function.

funArgs	(optional) A text field to annotate how the data object was created. If the result of <code>match.call()</code> is passed as this argument, the name and arguments used in the current function are captured
itemAttr	(optional) A named list of attributes to add directly to the item
parent	(optional) itemName of the parent of this item
overwrite	Whether to overwrite a matching data object stored in the itemName slot (default = FALSE)
init	Internal Use (default = FALSE)

Value

A DGEobj

Examples

```
## Not run:
myFunArgs <- match.call() # Capture calling function and arguments

myDGEobj <- addItem(myDGEobj, item = MyCounts,
                   itemName = "counts",
                   itemType = "counts",
                   funArgs = myFunArgs)

## End(Not run)
```

addItem	<i>Add multiple data items</i>
---------	--------------------------------

Description

Add multiple data items

Usage

```
addItem(dgeObj, itemList, itemTypes, parents, itemAttr, overwrite = FALSE)
```

Arguments

dgeObj	A DGEobj
itemList	A list of data items to add to DGEobj
itemTypes	A list of type values for each item on itemList
parents	(optional) A list of parent values for each item on itemList (optional, but highly recommended)
itemAttr	(optional) An named list of attributes to add to each item
overwrite	Whether to overwrite a matching data object stored in the itemName slot (default = FALSE)

Value

A DGEobj

Examples

```
## Not run:
# Replace a set of contrasts after adding something to each
myDGEobj <- addItems(myDGEobj, myContrastList, overwrite= TRUE)

## End(Not run)
```

annotateDGEobj	<i>Add annotations</i>
----------------	------------------------

Description

Reads an annotation file of key/value pairs or a named list and attaches them attributes to a DGEobj. If a file is used, it should be a text file containing key/value pairs separated by an equals sign. The keys parameter specifies which key we want to capture as attributes on the DGEobj. The value will then be the value of that attribute.

Usage

```
annotateDGEobj(dgeObj, annotations, keys = NULL)
```

Arguments

- dgeObj A class DGEobj created by function initDGEobj()
- annotations Either A character string path to a file with annotations given as key/value pairs separated by an equal sign, or a named list of key/value pairs
- keys A subset of keys to look for in the annotations and transfer to the DGEobj (default = NULL)

Value

A DGEobj

Examples

```
## Not run:
MyDgeObj <- annotateDGEobj(DGEobj, annotations)

## End(Not run)
```

as.list.DGEobj	<i>Cast as a simple list</i>
----------------	------------------------------

Description

Cast as a simple list

Usage

```
## S3 method for class 'DGEobj'
as.list(x, ...)
```

Arguments

x	A DGEobj
...	Additional parameters

Value

A simple list representation

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

as.list(exObj)
```

baseType	<i>Get the baseType of an internal data item</i>
----------	--

Description

Get the baseType of an internal data item

Usage

```
baseType(dgeObj, type)
```

Arguments

dgeObj	A DGEobj
type	An item type for which you want the baseType

Value

character string

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

baseType(exObj, type = "DGEList")
```

baseTypes *Get a list of the available baseTypes*

Description

Get a list of the available baseTypes

Usage

```
baseTypes(dgeObj)
```

Arguments

dgeObj (optional) A DGEobj

Value

A character vector of baseTypes

Examples

```
# Global definition of baseTypes
baseTypes()

# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

# Basetypes from a specific DGEobj
baseTypes(exObj)
```

dim.DGEobj *Get the "assay" dimensions (row/genes by col/samples)*

Description

Returns the dimensions of the assay data (baseType)

Usage

```
## S3 method for class 'DGEobj'  
dim(x)
```

Arguments

x A DGEobj

Value

An integer vector [r,c] with a length of 2.

dimnames.DGEobj *Get the "assay" names (row/genes by col/samples)*

Description

Returns a list of length 2 containing the the assay data names (baseType)

Usage

```
## S3 method for class 'DGEobj'  
dimnames(x)
```

Arguments

x A class DGEobj created by function initDGEobj()

Value

A list of length 2 containing rownames and colnames of the DGEobj

getAttribute	<i>Get a specified attribute</i>
--------------	----------------------------------

Description

Get a specified attribute

Usage

```
getAttribute(dgeObj, attrName)
```

Arguments

dgeObj	A DGEobj
attrName	Name of the attribute to retrieve

Value

The specified attribute value or NULL if the attribute doesn't exist

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

# Get an attribute from a DGEobj
getAttribute(exObj, "type")

# Get an attribute from a DGEobj item
getAttribute(exObj$designMatrix, "formula")
```

getAttributes	<i>Get all attributes</i>
---------------	---------------------------

Description

Get all user-defined attributes from a DGEobj except for any listed in the excludeList argument.

Usage

```
getAttributes(
  dgeObj,
  excludeList = list("dim", "dimnames", "names", "row.names", "class")
)
```

Arguments

dgeObj A DGEobj
 excludeList A list of attribute names to exclude from the output (default = list("dim", "dim-names", "names", "row.names"))

Value

A named list

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

getAttributes(exObj)

# Get the formula attribute from the design (if set)
attr(exObj$design, "formula")
```

getBaseType *Retrieve data items by baseType*

Description

Retrieve data items by baseType

Usage

```
getBaseType(dgeObj, baseType)
```

Arguments

dgeObj A DGEobj
 baseType One or more of: ["row", "col", "assay", "meta"]

Value

A list of data items

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

Assays <- getBaseType(exObj, baseType = "assay")
AssaysAndMeta <- getBaseType(exObj, c("assay", "meta"))
```

getItem *Retrieve a data item by name*

Description

Retrieve a data item by name

Usage

```
getItem(dgeObj, itemName)
```

Arguments

dgeObj	A DGEobj
itemName	Name of item to retrieve

Value

The requested data item

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

MyCounts <- getItem(exObj, "counts")
```

getItem *Retrieve data items by name*

Description

Retrieve data items by name

Usage

```
getItem(dgeObj, itemNames)
```

Arguments

dgeObj	A DGEobj
itemNames	A character string, character vector, or list names to retrieve

Value

A list

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

MyCounts <- getItem(exObj, "counts")
```

getType*Retrieve data items by type*

Description

Retrieve data items by type

Usage

```
getType(dgeObj, type, parent)
```

Arguments

dgeObj	A DGEobj
type	A type or list of types to retrieve
parent	(optional) Filter return list for common parent (e.g. useful to select one set of contrast results when multiple fits have been performed)

Value

A list of data items

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

MyContrastList <- getType(exObj, type = "topTable")
MyRawData <- getType(exObj, type = list("counts", "design", "geneData"))
```

initDGEobj	<i>Initialize with base data (counts, gene annotations, sample annotations)</i>
------------	---

Description

Initialize with base data (counts, gene annotations, sample annotations)

Usage

```
initDGEobj(
  counts,
  rowData,
  colData,
  level,
  customAttr,
  allowShortSampleIDs = FALSE,
  DGEobjDef = .DGEobjDef
)
```

Arguments

counts	A count matrix or dataframe with row and colnames
rowData	Gene, isoform or exon level annotation. Rownames must match the rownames in count matrix
colData	A dataframe describing the experiment design. Rownames much match the colnames(counts)
level	One of "gene", "isoform", or "exon"
customAttr	(optional) Named list of attributes
allowShortSampleIDs	Using sequential integer rownames (even if typed as character) is discouraged and by default will abort the DGEobj creation. If you have a legitimate need to have short sample names composed of numeric characters, you can set this argument to TRUE (default = FALSE)
DGEobjDef	An object definition. Defaults to the global DGEobj definition (.DGEobjDef) and you usually shouldn't change this unless you're customizing the object for new data types.

Value

A DGEobj

Examples

```
## Not run:
# Initialize a DGEobj
myDgeObj <- initDGEobj(counts = MyCounts,
                      rowData = MyGeneAnnotation,
                      colData = MyDesign,
                      level = "gene",
                      customAttr = list (Genome = "Mouse.B38",
                                       GeneModel = "Ensembl.R84"))

## End(Not run)
```

inventory

Retrieve the object inventory

Description

Retrieve the object inventory

Usage

```
inventory(dgeObj, verbose = FALSE)
```

Arguments

dgeObj A DGEobj
verbose Add funArgs to the output (default = FALSE)

Value

A data.frame summarizing the data contained in the DGEobj

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

inventory(exObj)
```

newType	<i>Add a new type definition to a DGEobj</i>
---------	--

Description

Add a new type definition to a DGEobj

Usage

```
newType(dgeObj, itemType, baseType, uniqueItem = FALSE)
```

Arguments

dgeObj	A DGEobj
itemType	The name of the new type to create
baseType	The baseType of the new item. One of [row, col, assay, meta]
uniqueItem	If set to TRUE, only one instance of the new type is allowed in a DGEobj

Value

A DGEobj

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

exObj <- newType(exObj,
                 itemType = "AffyRMA",
                 baseType = "assay",
                 uniqueItem = TRUE)
```

print.DGEobj	<i>Print the Inventory</i>
--------------	----------------------------

Description

Print the Inventory

Usage

```
## S3 method for class 'DGEobj'
print(x, ..., verbose = FALSE)
```

Arguments

x	A DGEobj
...	Additional parameters
verbose	Add funArgs to the output (default = FALSE)

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

print(exObj)
```

resetDGEobj	<i>Reset to original data</i>
-------------	-------------------------------

Description

During a workflow, a DGEobj typically gets filtered down to remove samples that fail QC or non-expressed genes. The resetDGEobj() function produces a new DGEobj with the original unfiltered data.

Usage

```
resetDGEobj(dgeObj)
```

Arguments

dgeObj	A DGEobj
--------	----------

Value

A DGEobj

Examples

```
#example object
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

# subset to first 10 rows to show reset functionality
exObj <- exObj[c(1:10), ]

exObj <- resetDGEobj(exObj)
```

rmItem	<i>Removes a named data item</i>
--------	----------------------------------

Description

Removes a named data item

Usage

```
rmItem(dgeObj, itemName)
```

Arguments

dgeObj	A class DGEobj
itemName	Name of the item to remove

Value

A DGEobj

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

exObj <- rmItem(exObj, "design")
```

rmItems	<i>Removes list of named data items</i>
---------	---

Description

Removes list of named data items

Usage

```
rmItems(dgeObj, items)
```

Arguments

dgeObj	A DGEobj
items	Either a character vector of names or numeric indexes of items to remove. Use <code>inventory(DGEobj)</code> to view the indexes of items.

Value

A DGEobj

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

exObj <- rmItems(exObj, c("design", "design_orig"))
exObj <- rmItems(exObj, c(1:2))
```

setAttribute	<i>Set an attribute</i>
--------------	-------------------------

Description

Set an attribute on a DGEobj or on a specific item within a DGEobj.

Usage

```
setAttribute(dgeObj, attrib, attribName)
```

Arguments

dgeObj	A DGEobj
attrib	An attribute value to add
attribName	A name for the attribute

Details

The function adds or updates the attribute passed to it. To remove an attribute, pass NULL as the attribute value.

Value

A DGEobj

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

# Assign attribute to a DGEobj
exObj <- setAttribute(exObj, "RNA-SEQ", "Platform")

# Set attributes on an item inside a DGEobj
exObj[["counts"]] <- setAttribute(exObj[["counts"]], FALSE, "normalized")
```

setAttributes	<i>Set attributes</i>
---------------	-----------------------

Description

Set one or more attributes on a DGEobj or on a specific item within a DGEobj.

Usage

```
setAttributes(dgeObj, attribs)
```

Arguments

dgeObj	A DGEobj
attribs	A named list of attribute/value pairs

Details

This function adds attributes without deleting the attributes that are already present. Any named attribute that already exists in the object will be updated. To remove an attribute from an object pass NULL as the attribute value.

Value

A DGEobj

Examples

```
# example DGEobj
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))

# Assign attributes to a DGEobj
MyAttributes <- list(Platform      = "RNA-Seq",
                    Instrument    = "HiSeq",
                    Vendor        = "Unknown",
                    readType      = "PE",
                    readLength    = 75,
                    strandSpecific = TRUE)
exObj <- setAttributes(exObj, MyAttributes)

# Set attributes on an item inside a DGEobj
MyAttributes <- list(normalized    = FALSE,
                    LowIntFilter = "FPK >5 in >= 1 group")
exObj[["counts"]] <- setAttributes(exObj[["counts"]], MyAttributes)
```

showAttributes *Print attributes*

Description

This function prints all attributes regardless of the class of the attribute value.

Usage

```
showAttributes(  
  dgeObj,  
  skipList = c("dim", "dimnames", "rownames", "colnames", "listData", "objDef")  
)
```

Arguments

dgeObj A DGEobj
skipList A character vector of attributes to skip. Use this to avoid printing certain lengthy attributes like rownames. Defaults to c("dim", "dimnames", "rownames", "colnames", "listData", "objDef")

Details

Note Use showMeta() to only retrieve attributes that are key/value pairs.

Examples

```
# example DGEobj  
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))  
  
showAttributes(exObj)
```

showMeta *Retrieve the Key/Value metadata attributes*

Description

Retrieve the Key/Value metadata attributes

Usage

```
showMeta(dgeObj, printed = TRUE)
```

Arguments

dgeObj A DGEobj with attributes
printed Whether to print the list (default = TRUE)

Value

A data.frame with "Attribute" and "Value" columns

Examples

```
# example DGEobj  
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))  
  
showMeta(exObj)
```

showTypes *Returns and prints the list of all defined types*

Description

Returns and prints the list of all defined types

Usage

```
showTypes(dgeObj, printed = TRUE)
```

Arguments

dgeObj A DGEobj
printed Whether to print the list (default = TRUE)

Value

data.frame

Examples

```
# example DGEobj  
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))  
  
showTypes(exObj)
```

subset.DGEobj *Subset internal row or column data*

Description

Subset internal row or column data

Usage

```
## S3 method for class 'DGEobj'  
subset(x, ..., row, col, drop = FALSE, debug = FALSE)
```

Arguments

x	A DGEobj
...	Additional parameters
row	Row index for the subset
col	Col index for the subset
drop	Included for compatibility only
debug	(default = FALSE) Set to TRUE to get additional information on the console if subsetting a DGEobj fails with a dimension error.

Value

A DGEobj

Examples

```
# example DGEobj  
exObj <- readRDS(system.file("exampleObj.RDS", package = "DGEobj"))  
  
exObj <- subset(exObj, 1:10, 5:50)
```

[.DGEobj *Subset with square brackets*

Description

Subset with square brackets

Usage

```
## S3 method for class 'DGEobj'  
x[...]
```

Arguments

x A DGEobj
... Additional parameters

Value

A DGEobj

Index

[.DGEobj, 22

addItem, 3
addItem, 4
annotateDGEobj, 5
as.list.DGEobj, 6

baseType, 6
baseTypes, 7

DGEobj-package, 2
dim.DGEobj, 8
dimnames.DGEobj, 8

getAttribute, 9
getAttributes, 9
getBaseType, 10
getItem, 11
getItems, 11
getType, 12

initDGEobj, 13
inventory, 14

newType, 15

print.DGEobj, 15

resetDGEobj, 16
rmItem, 17
rmItems, 17

setAttribute, 18
setAttributes, 19
showAttributes, 20
showMeta, 20
showTypes, 21
subset.DGEobj, 22