Package 'SummarizedBenchmark'

May 20, 2019

Type Package

Title Classes and methods for performing benchmark comparisons

Version 2.2.0

Author
Alejandro Reyes <alejandro.reyes.ds@gmail.com>, Patrick Kimes <patrick.kimes@gmail.com>

Maintainer
Alejandro Reyes <alejandro.reyes.ds@gmail.com>, Patrick Kimes <patrick.kimes@gmail.com>

Description This package defines the BenchDesign and SummarizedBenchmark classes for building, executing, and evaluating benchmark experiments of computational methods. The Summarized-Benchmark
class extends the RangedSummarizedExperiment object, and is designed to provide infrastructure to store and compare the results of applying different methods to a shared data set.
This class provides an integrated interface to store metadata such as method parameters and software
versions as well as ground truths (when these are available) and evaluation metrics.

biocViews Software, Infrastructure

Depends R (>= 3.5), tidyr, SummarizedExperiment, S4Vectors, BiocGenerics, methods, UpSetR, rlang, stringr, utils, BiocParallel, ggplot2, mclust, dplyr, digest, sessioninfo, crayon, tibble

Suggests iCOBRA, BiocStyle, knitr, magrittr, IHW, qvalue, testthat, DESeq2, edgeR, limma, tximport, readr, scRNAseq, splatter, scater, rnaseqcomp, biomaRt

License GPL (>= 3)

Encoding UTF-8

LazyData true

VignetteBuilder knitr

RoxygenNote 6.1.0

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

git_branch RELEASE_3_9

git_last_commit 3dcb284

git_last_commit_date 2019-05-02

Date/Publication 2019-05-19

R topics documented:

addMethod . . . 2
addPerformanceMetric . . . 4
allSB . . . 5
availableMetrics . . . 5
BDData . . . 6
BDData-class . . . 6
BDMethod . . . 7
BDMethod-class . . . 8
BDMethodList . . . 8
BDMethodList-accessors . . . 9
BDMethodList-class . . . 10
BenchDesign . . . 10
BenchDesign-accessors . . . 11
BenchDesign-class . . . 12
buildBench . . . 12
compareBDData . . . 14
compareBDMethod . . . 15
compareBenchDesigns . . . 15
dropMethod . . . 16
estimateMetricsForAssay . . . 17
expandMethod . . . 19
hashBDData . . . 20
modifyMethod . . . 21
performanceMetrics . . . 22
plotMethodsOverlap . . . 23
plotROC . . . 24
printMethod . . . 24
sb . . . 25
SummarizedBenchmark . . . 26
SummarizedBenchmark-accessors . . . 27
SummarizedBenchmark-class . . . 28
tdat . . . 29
tidyBDMethod . . . 29
tidyUpMetrics . . . 30
updateBench . . . 31

Index **33**

`addMethod` *Add new method to BenchDesign object*

Description

This function takes a BenchDesign object and returns a modified object with the specified method included. At a minimum, a string name for the method, `label`, and the workhorse function for the method, `func`, must be specified in addition to the primary BenchDesign object.

Usage

```
addMethod(bd, label, func, params = rlang::quos(), post = NULL,
  meta = NULL)
```

Arguments

bd	BenchDesign object.
label	Character name for the method.
func	Primary function to be benchmarked.
params	Named quosure list created using rlang::quos of parameter = value pairs to be passed to func.
post	Optional post-processing function that takes results of func as input. Ignored if NULL. If multiple assays (metrics) should be generated for each method, this can be accomplished by specifying a named list of post-processing functions, one for each assay. (default = NULL)
meta	Optional metadata information for method to be included in colData of SummarizedBenchmark object generated using buildBench. See Details for more information. Ignored if NULL. (default = NULL)

Details

The inputs for the call to label should be specified as parameter = value pairs, where the value can be any fixed value, variable, or column in the bdata of the BenchDesign object.

An optional secondary function, post, can be specified if the output of the workhorse function, func, needs to be further processed. As an example, post may be a simple "getter" function for accessing the column of interest from the large object returned by func.

The optional meta parameter accepts a named list of metadata tags to be included for the method in the resulting SummarizedBenchmark object. This can be useful for two primary cases. First, it can help keep analyses better organized by allowing the specification of additional information that should be stored with methods, e.g. a tag for "method type" or descriptive information on why the method was included in the comparison. Second, and more improtantly, the meta parameter can be used to overwrite the package and version information that is automatically extracted from the function specified to func. This is particularly useful when the function passed to func is a wrapper for a script in (or outside of) R, and the appropriate package and version information can't be directly pulled from func. In this case, the user can either manually specify the "pkg_name" and "pkg_vers" values to meta as a list, or specify a separate function that should be used to determine the package name and version. If a separate function should be used, it should be passed to meta as a list entry with the name pkg_func and first quoted using rlang::quo, e.g. list(pkg_func = quo(p.adjust)).

Value

A copy of the originally supplied BenchDesign with the new method added.

Author(s)

Patrick Kimes

Examples

```
## create example data set of p-values
df <- data.frame(pval = runif(100))

## example calculating qvalue from pvalues

## using standard call
```

```
qv <- qvalue::qvalue(p = df$pval)
qv <- qv$qvalue

## adding same method to BenchDesign
bench <- BenchDesign(data = df)
bench <- addMethod(bench,
                   label = "qv",
                    func = qvalue::qvalue,
                    post = function(x) { x$qvalue },
                    params = rlang::quos(p = pval))
```

`addPerformanceMetric` *Add a performance metric definition to a* `SummarizedBenchmark` *object.*

Description

This is a function to define performance metrics for benchmarking methods. The function is saved into the `performanceMetrics` slot.

Usage

```
addPerformanceMetric(object, evalMetric, assay, evalFunction = NULL)
```

Arguments

`object`	A `SummarizedBenchmark` object.
`evalMetric`	A string with the name of the evaluation metric.
`assay`	A string with an assay name. Indicates the assay that should be given as input to this performance metric.
`evalFunction`	A function that calculates a performance metric. It should contain at least two arguments, query and truth, where query is the output vector of a method and truth is the vector of true values. If additional parameters are specified, they must contain default values. If NULL, the 'evalMetric' string must be the name of a predefined metric available through 'availableMetrics()$function'.

Value

A `SummarizedBenchmark` object.

Author(s)

Alejandro Reyes

Examples

```
data( sb )
sb <- addPerformanceMetric(
   object=sb,
   assay="qvalue",
   evalMetric="TPR",
   evalFunction = function( query, truth, alpha=0.1 ){
       goodHits <- sum( (query < alpha) & truth == 1 )
       goodHits / sum(truth == 1)
   }
)
```

`allSB` *SummarizedBenchmark object of isoform quantification results*

Description

This object is a `SummarizedBenchmark` object containing isoform quantifications from salmon, sailfish and kallisto from 4 mouse samples (2 hearts and 2 brains) part of the Mouse BodyMap. Its generation is described in one of the vignettes of this package.

Source

Mouse BodyMap (Li et al, 2014). SRA accession numbers SRR5273705, SRR5273689, SRR5273699 and SRR5273683.

Examples

```
data(quantSB)
```

`availableMetrics` *availableMetrics*

Description

This function returns a data frame summarizing the default performance metrics provided in this package. The data.frame contains three columns, 'functions' is the name of the performance metric, 'description' is longer description of the performance metric and 'requiredTruth' is logical depending on whether the performance metrics require ground truths.

Usage

```
availableMetrics()
```

Value

A data.frame summarizing the default performance metrics provided in this package.

Examples

```
availableMetrics()
```

`BDData` *Create a new BDData*

Description

Initialized a new BDData object for benchmarking methods.

Usage

```
BDData(data)

## S4 method for signature 'ANY'
BDData(data)

## S4 method for signature 'BenchDesign'
BDData(data)

## S4 method for signature 'SummarizedBenchmark'
BDData(data)

## S4 method for signature 'BDData'
BDData(data)
```

Arguments

`data`	a list object of data or MD5 hash string

Value

BDData object

Author(s)

Patrick Kimes

`BDData-class` *BDMethod class*

Description

Container for individual methods to be compared as part of a BenchDesign object.

Usage

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

Arguments

object	BDData object to show

Slots

data a list or MD5 hash of the data.

type a character string indicating whether the data slot contains the 'data' or a 'md5hash' of the data.

Author(s)

Patrick Kimes

BDMethod *Create a new BDMethod*

Description

Initializes a new BenchDesign method (BDMethod) object for benchmarking methods.

Usage

```
BDMethod(x, params = rlang::quos(), post = NULL, meta = NULL, ...)

## S4 method for signature 'quosure'
BDMethod(x, params = rlang::quos(), post = NULL,
  meta = NULL, ...)

## S4 method for signature '`function`'
BDMethod(x, params = rlang::quos(), post = NULL,
  meta = NULL, ...)
```

Arguments

x	main method function or function quosure
params	list of quosures specifying function parameters. (default = coderlang::quos())
post	list of functions to be applied to the output of x. (default = NULL)
meta	list of meta data. (default = NULL)
...	other parameters.

Value

BDMethod object

BDMethod-class *BDMethod class*

Description

Container for individual methods to be compared as part of a BenchDesign object.

Usage

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

Arguments

object	BDMethod object to show

Slots

f a function to be benchmarked

fc a captured expression of the function f

params a list of quosures specifying function parameters

post a list of functions to be applied to the output of f

meta a list of meta data

Author(s)

Patrick Kimes

BDMethodList *Create a new BDMethodList*

Description

Initialized a new SimpleList of BenchDesign method (BDMethod) objects.

Usage

```
BDMethodList(..., x = NULL)

## S4 method for signature 'ANY'
BDMethodList(..., x = NULL)
```

Arguments

...	a named list of BDMethod objects
x	a BenchDesign or SummarizedBenchmark object to extract the BDMethodList from. (default = NULL)

Value

BDMethodList object

Author(s)

Patrick Kimes

Examples

```
BDMethodList()
```

BDMethodList-accessors *BDMethodList Accessors*

Description

These methods can be used to access, set, and remove elements of a BDMethodList object.

Usage

```
## S4 replacement method for signature 'BDMethodList,character,BDMethod'
BDMethod(x, i) <- value

## S4 replacement method for signature 'BDMethodList,character,`NULL`'
BDMethod(x, i) <- value

## S4 method for signature 'BDMethodList'
BDMethod(x, i = 1)
```

Arguments

x	BDMethodList object.
i	character name or integer index of a BDMethod object. (Note: must be a character name for replacement operations.)
value	replacement object, either BDMethod or NULL object.

Value

modified BDMethodList object

Author(s)

Patrick Kimes

BDMethodList-class *BDMethodList class*

Description

Extension of the SimpleList class to contain a list of BDMethod objects.

Usage

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

Arguments

object	BDMethodList object to show

Author(s)

Patrick Kimes

BenchDesign *Create a new BenchDesign*

Description

Initializes a new BenchDesign object for benchmarking methods.

Usage

```
BenchDesign(..., methods = NULL, data = NULL)

## S4 method for signature 'ANY'
BenchDesign(..., methods = NULL, data = NULL)
```

Arguments

...	named set of BDMethod objects and/or unnamed BenchDesign objects. Only the methods of any BenchDesign object will be used, and the data slot of the objects will be ignored.
methods	named set of BDMethod objects and/or unnamed BenchDesign objects as a list. (default = NULL)
data	optional data.frame or other list object to be used in the benchmark. (default = NULL)

Value

BenchDesign object.

Author(s)

Patrick Kimes

Examples

```
## with no input
bd <- BenchDesign()

## with toy data.frame
df <- data.frame(x1 = rnorm(20), y1 = rnorm(20))
bd <- BenchDesign(data = df)
```

BenchDesign-accessors *BenchDesign Accessors*

Description

These methods can be used to access, set, and remove elements of a BenchDesign object.

Usage

```
BDData(x) <- value

BDMethod(x, i) <- value

BDMethodList(x) <- value

## S4 replacement method for signature 'BenchDesign,BDDataOrNULL'
BDData(x) <- value

## S4 replacement method for signature 'BenchDesign,character,BDMethod'
BDMethod(x, i) <- value

## S4 replacement method for signature 'BenchDesign,character,`NULL`'
BDMethod(x, i) <- value

## S4 replacement method for signature 'BenchDesign,BDMethodList'
BDMethodList(x) <- value

## S4 method for signature 'BenchDesign'
BDMethod(x, i = 1)

## S4 method for signature 'BenchDesign'
BDMethodList(..., x = NULL)
```

Arguments

x	BenchDesign object.
value	replacement object, either BDData, BDMethod, BDMethodList or NULL object.

i	character name or integer index of a BDMethod object. (Note: must be a character name for replacement operations.)
...	Futher arguments, perhaps used by methods

Value

modified BenchDesign object

Author(s)

Patrick Kimes

BenchDesign-class *BenchDesign class*

Description

Container for benchmark methods and data.

Usage

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

Arguments

object	BenchDesign object to show

Slots

`data` a list containing the data to be used in the benchmark.

`methods` a list of `BDMethods` to be compared in the benchmark.

Author(s)

Patrick Kimes

---

buildBench *Make SummarizedBenchmark from BenchDesign*

Description

Function to evaluate `BenchDesign` methods on supplied data set to generate a `SummarizedBenchmark`. In addition to the results of applying each method on the data, the returned `SummarizedBenchmark` also includes metadata for the methods in the `colData` of the returned object, metadata for the data in the `rowData`, and session information in the `metadata`.

Usage

```
buildBench(bd, data = NULL, truthCols = NULL, ftCols = NULL,
  sortIDs = FALSE, keepData = FALSE, catchErrors = TRUE,
  parallel = FALSE, BPPARAM = bpparam())
```

Arguments

bd	BenchDesign object.
data	Data set to be used for benchmarking, will take priority over data set specified to BenchDesign object. Ignored if NULL. (default = NULL)
truthCols	Character vector of column names in data set corresponding to ground truth values for each assay. If specified, column will be added to the groundTruth DataFrame of the returned SummarizedBenchmark object. If the BenchDesign includes only a single assay, the same name will be used for the assay. If the BenchDesign includes multiple assays, to map data set columns with assays, the vector must have names corresponding to the assay names specified to the post parameter at each addMethod call. (default = NULL)
ftCols	Vector of character names of columns in data set that should be included as feature data (row data) in the returned SummarizedBenchmark object. (default = NULL)
sortIDs	Whether the output of each method should be merged and sorted using IDs. See Details for more information. (default = FALSE)
keepData	Whether to store the data as part of the BenchDesign slot of the returned SummarizedBenchmark object. If FALSE, a MD5 hash of the data will be stored with the BenchDesign slot. (default = FALSE)
catchErrors	logical whether errors produced by methods during evaluation should be caught and printed as a message without stopping the entire build process. (default = TRUE)
parallel	Whether to use parallelization for evaluating each method. Parallel execution is performed using BiocParallel. Parameters for parallelization should be specified with BiocParallel::register or through the BPPARAM parameter. (default = FALSE)
BPPARAM	Optional BiocParallelParam instance to be used when parallel is TRUE. If not specified, the default instance from the parameter registry is used.

Details

Parallelization is performed across methods. Therefore, there is currently no benefit to specifying more cores than the total number of methods in the BenchDesign object.

By default, errors thrown by individual methods in the BenchDesign are caught during evaluation and handled in a way that allows buildBench to continue running with the other methods. The error is printed as a message, and the corresponding column in the returned SummarizedBenchmark object is set to NA. Since many benchmarking experiments can be time and computationally intensive, having to rerun the entire analysis due to a single failed method can be frustrating. Default error catching was included to alleviate these frustrations. However, if this behavior is not desired, setting catchErrors = FALSE will turn off error handling.

If sortIDs is TRUE, each method must return a named vector or list. The names will be used to align the output of each method in the returned SummarizedBenchmark. Missing values from each method will be set to NA. This can be useful if the different methods return overlapping, but

not identical, results. If `truthCols` is also specified, and sorting by IDs is necessary, rather than specifying 'TRUE', specify the string name of a column in the `data` to use to sort the method output to match the order of `truthCols`.

When a method specified in the BenchDesign does not have a postprocessing function specified to 'post', the trivial `base::identity` function is used as the default postprocessing function.

Value

`SummarizedBenchmark` object.

Author(s)

Patrick Kimes

Examples

```
## with toy data.frame
df <- data.frame(pval = rnorm(100))
bench <- BenchDesign(data = df)

## add methods
bench <- addMethod(bench, label = "bonf", func = p.adjust,
                   params = rlang::quos(p = pval, method = "bonferroni"))
bench <- addMethod(bench, label = "BH", func = p.adjust,
                   params = rlang::quos(p = pval, method = "BH"))

## evaluate benchmark experiment
sb <- buildBench(bench)

## evaluate benchmark experiment w/ data sepecified
sb <- buildBench(bench, data = df)
```

---

compareBDData *Compare BDData objects*

Description

Simple comparison of two BDData objects based on comparing both type and data hash.

Usage

```
compareBDData(x, y)
```

Arguments

`x`	a `BDData` or `BenchDesign` object
`y`	a `BDData` or `BenchDesign` object

Value

list of two values giving agreement of "data" and "type".

Author(s)

Patrick Kimes

`compareBDMethod` *Compare BDMethod objects*

Description

Simple comparison of two BDMethod objects based on meta data.

Usage

```
compareBDMethod(x, y)
```

Arguments

`x`	a `BDMethod` object
`y`	a `BDMethod` object

Value

logical value indicating whether the two objects produced the same meta data.

Author(s)

Patrick Kimes

`compareBenchDesigns` *Compare BenchDesign objects*

Description

Comparison of BenchDesign objects and BenchDesign method information stored in SummarizedBenchmark objects. Inputs can be either BenchDesign or SummarizedBenchmark objects. If SummarizedBenchmark objects are specified, the method metadata stored in the `colData` will be used for the comparison. If only a single SummarizedBenchmark object is specified, the `colData` information will be compared with the BenchDesign object in the `BenchDesign` slot of the object. To compare the `BenchDesign` slots of SummarizedBenchmark objects, the BenchDesigns should be extracted with `BenchDesign(sb)` and passed as inputs (see Examples).

Usage

```
compareBenchDesigns(x, y = NULL, ...)

## S4 method for signature 'SummarizedBenchmark,missing'
compareBenchDesigns(x, y = NULL,
  ...)

## S4 method for signature 'SummarizedBenchmark,SummarizedBenchmark'
compareBenchDesigns(x,
  y = NULL, ...)

## S4 method for signature 'SummarizedBenchmark,BenchDesign'
compareBenchDesigns(x,
  y = NULL, ...)

## S4 method for signature 'BenchDesign,SummarizedBenchmark'
compareBenchDesigns(x,
  y = NULL, ...)

## S4 method for signature 'BenchDesign,BenchDesign'
compareBenchDesigns(x, y = NULL, ...)
```

Arguments

x	a SummarizedBenchmark or BenchDesign object
y	an optional second SummarizedBenchmark or BenchDesign object (default = NULL)
...	other parameters

Value

list of comparison results

Author(s)

Patrick Kimes

`dropMethod` *Remove method from BenchDesign object*

Description

This function takes a BenchDesign object and the name of a method already defined in the object, and returns a reduced BenchDesign object with the specified method removed.

Usage

```
dropMethod(bd, label)
```

Arguments

bd	BenchDesign object.
label	Character name of method to be modified.

Value

Modified BenchDesign object.

Author(s)

Patrick Kimes

Examples

```
## empty BenchDesign
bench <- BenchDesign()

## add methods
bench <- addMethod(bench, label = "bonf", func = p.adjust,
                   params = rlang::quos(p = pval, method = "bonferroni"))
bench <- addMethod(bench, label = "BH", func = p.adjust,
                   params = rlang::quos(p = pval, method = "BH"))

## remove methods
bench <- dropMethod(bench, label = "bonf")
```

---

estimateMetricsForAssay

Estimate performance metrics.

Description

These functions estimate the performance metrics, either passed as arguments or added previously with the addPerformanceMetric function. The function will estimate the performance metric for each method.

Usage

```
estimateMetricsForAssay(object, assay, evalMetric = NULL,
  addColData = FALSE, evalFunction = NULL, tidy = FALSE, ...)

estimatePerformanceMetrics(object, addColData = FALSE, tidy = FALSE,
  rerun = TRUE, ...)
```

Arguments

object	A SummarizedBenchmark object.
assay	A string with an assay name. Indicates the assay that should be given as input to this performance metric.
evalMetric	A string with the name of the evaluation metric.

addColData	Logical (default: FALSE). If TRUE, the results are added to the colData slot of the SummarizedExperiment object and the object is returned. If FALSE, only a DataFrame with the results is returned.
evalFunction	A function that calculates a performance metric. It should contain at least two arguments, query and truth, where query is the output vector of a method and truth is the vector of ground true values. If additional parameters are specified, they must contain default values. If this parameter is passed, the metrics in the object are ignored and only this evaluation metric is estimated.
tidy	Logical (default: FALSE). If TRUE, a long formated data.frame is returned.
...	Additional parameters passed to the performance functions.
rerun	Logical (default: TRUE). By default, all performance metrics are recalculated everytime that estimatePerformanceMetrics is called. If FALSE, performance metrics will only be calculated for newly added methods or modified methods.

Value

Either a SummarizedBenchmark object, a DataFrame or a data.frame.

Functions

- estimateMetricsForAssay: Estimate performance metrics for a given assay
- estimatePerformanceMetrics: Estimate performance metrics for all assays

Author(s)

Alejandro Reyes

Examples

```
data( sb )
sb <- addPerformanceMetric(
   object=sb,
   assay="qvalue",
   evalMetric="TPR",
   evalFunction = function( query, truth, alpha=0.1 ){
       goodHits <- sum( (query < alpha) & truth == 1 )
       goodHits / sum(truth == 1)
   }
)

qvalueMetrics <- estimateMetricsForAssay( sb, assay="qvalue" )
allMetrics <- estimatePerformanceMetrics( sb )
allMetricsTidy <- estimatePerformanceMetrics( sb, tidy=TRUE )
```

`expandMethod` *Expand method in BenchDesign object across parameter settings*

Description

This function takes a BenchDesign object and the name of a method already defined in the object, and returns a modified BenchDesign object with multiple variants of the method differing only by the specified parameter sets. In other words, this function "expands" the set of methods using a set of parameters.

Usage

```
expandMethod(bd, label, params, onlyone = NULL, .replace = FALSE,
  .overwrite = FALSE)
```

Arguments

`bd`	BenchDesign object.
`label`	Character name of method to be expanded.
`params`	Named list of quosure lists specifying the label of the new methods to be added to the BenchDesign, and the set of parameters to overwrite in the original method definition for each new method. Alternatively, if `onlyone` is non-NULL, a single quosure list with `name = value` pairs specifying the label of the new methods and the values to use for overwriting the parameter specified in `onlyone`.
`onlyone`	Character name of a parameter to be modified. Only specify if just a single parameter should be replaced in the original method definition. Ignored if NULL. (default = NULL)
`.replace`	Logical whether original `label` object should be removed if method expansion is successful. (default = FALSE)
`.overwrite`	Logical whether to overwrite the existing list of parameters (TRUE) or to simply add the new parameters to the existing list (FALSE). (default = FALSE)

Value

Modified BenchDesign object.

Author(s)

Patrick Kimes

Examples

```
## empty BenchDesign
bench <- BenchDesign()

## add basic 'padjust' method
bench <- addMethod(bench, label = "padjust",
                   func = p.adjust,
                   params = rlang::quos(p = pval,
                                        method = "none"))
```

```
## modify multiple parameters, "p" and "method"
bench_exp <- expandMethod(bench, label = "padjust",
                          params = list(
        bonf = rlang::quos(p = round(pval, 5),
                           method = "bonferonni"),
        bh = rlang::quos(p = round(pval, 3),
                         method = "BH")))
printMethods(bench_exp)

## only modify a single parameter using the 'onlyone=' parameter
bench_exp <- expandMethod(bench, label = "padjust",
                          onlyone = "method",
                          params = rlang::quos(bonf = "bonferonni",
                                               BH = "BH"))
printMethods(bench_exp)
```

`hashBDData` *Convert BDData to BDData with MD5 Hash*

Description

Simple converting function to replace data object in BDData object with MD5 hash value.

Usage

```
hashBDData(object)

## S4 method for signature 'BDData'
hashBDData(object)

## S4 method for signature 'BenchDesign'
hashBDData(object)
```

Arguments

`object`	a `BDData` or `BenchDesign` object

Value

an object of the same class as `object` with data converted to a MD5 hash.

Author(s)

Patrick Kimes

`modifyMethod` *Modify definition of method in BenchDesign object*

Description

This function takes a BenchDesign object and the name of a method already defined in the object, and returns a modified BenchDesign object with the specified changes made only to the named method. At a minimum, a string name for the method, `label`, must be specified in addition to the primary BenchDesign object.

Usage

```
modifyMethod(bd, label, params, .overwrite = FALSE)
```

Arguments

`bd`	BenchDesign object.
`label`	Character name of method to be modified.
`params`	Named quosure list created using `rlang::quos` of `parameter = value` paiars to replace in the method definition. The `post`, and `meta` parameters of the method can be modified using the special keywords, `bd.post`, and `bd.meta` (the prefix denoting that these values should modify BenchDesign parameters). All other named parameters will be added to the list of parameters to be passed to `func`.
`.overwrite`	Logical whether to overwrite the complete existing list of parameters to be passed to `func` (TRUE), or to simply add the new parameters to the existing list and only replace overlapping parameters (FALSE). (default = FALSE)

Value

Modified BenchDesign object.

Author(s)

Patrick Kimes

Examples

```
## empty BenchDesign
bench <- BenchDesign()

## add method
bench <- addMethod(bench, label = "qv",
                   func = qvalue::qvalue,
                   post = function(x) { x$qvalue },
                   meta = list(note = "storey's q-value"),
                   params = rlang::quos(p = pval))

## modify method 'meta' property of 'qv' method
bench <- modifyMethod(bench, label = "qv",
                      params = rlang::quos(bd.meta =
                           list(note = "Storey's q-value")))
```

```
## verify that method has been updated
printMethod(bench, "qv")
```

`performanceMetrics` *Accessor and replacement functions for the slot 'performanceMetrics' of a SummarizedBenchmark object.*

Description

Accessor and replacement functions for the slot 'performanceMetrics' of a SummarizedBenchmark object.

Usage

```
performanceMetrics(object, ...)

performanceMetrics(object, ...) <- value

## S4 method for signature 'SummarizedBenchmark'
performanceMetrics(object, assay = NULL)


  ## S4 replacement method for signature 'SummarizedBenchmark,SimpleList'
performanceMetrics(object) <- value
```

Arguments

`object`	a `SummarizedBenchmark` object.
`...`	Futher arguments, perhaps used by methods
`value`	A SimpleList of the same length as the number of assays
`assay`	A character string indicating an assay name

Value

A SimpleList with one element for each assay. Each element of the list contains a list of performance functions.

Author(s)

Alejandro Reyes

See Also

`addPerformanceMetric`, `estimatePerformanceMetrics`

Examples

```
data( sb )
performanceMetrics( sb )
performanceMetrics( sb, assay="qvalue" )
performanceMetrics( sb ) <- SimpleList( qvalue=list(), logFC=list() )
```

---

plotMethodsOverlap *Calls UpSetR for qvalues of a* SummarizedBenchmark *object.*

Description

This function looks for an assay, called by default 'qvalue', and given an alpha threshold, it binarizes the assay matrix depending on whether its values are below the alpha threshold. Then it uses the function upset to plot the overlaps. The plot is only generated if at least 2 methods have observations that pass the alpha threshold.

Usage

```
plotMethodsOverlap(object, assay = "qvalue", alpha = 0.1, ...)
```

Arguments

object	A SummarizedBenchmark object.
assay	The name of an assay.
alpha	An alpha value.
...	Further arguments passed to upset

Value

An upseR plot.

Author(s)

Alejandro Reyes

Examples

```
data( sb )
## Not run:
plotMethodsOverlap(sb)

## End(Not run)
```

plotROC *Plotting ROC curves*

Description

This function inputs a SummarizedBenchmark object, looks for an assay called 'qvalue' and plots receiver operating characteristic curves for each of the methods to benchmark.

Usage

```
plotROC(object, assay = "qvalue")
```

Arguments

object	A SummarizedBenchmark object.
assay	An assay name.

Value

A ggplot object.

Author(s)

Alejandro Reyes

Examples

```
data( sb )
## Not run:
plotROC( sb )

## End(Not run)
```

printMethod *Pretty print methods in a BenchDesign*

Description

Print out details about a method included in the BenchDesign. The printMethods function is just a wrapper to call printMethod on all methods in the BenchDesign.

Usage

```
printMethod(bd, n = NULL)

printMethods(bd)
```

Arguments

bd	BenchDesign object.
n	name of a method in the BenchDesign to show.

Value

Brief description is returned to console.

Author(s)

Patrick Kimes

Patrick Kimes

Examples

```
## create empty BenchDesign
bench <- BenchDesign()

## currently no methods
printMethods(bench)

## add method
bench <- addMethod(bench, label = "method_a", p.adjust)
bench <- addMethod(bench, label = "method_b", qvalue::qvalue)

## show a single method
printMethod(bench, "method_a")

## show all methods
printMethods(bench)
```

---

sb *SummarizedBenchmark example*

Description

This object contains the example data from the `iCOBRA` package reformatted as a `SummarizedBenchmark` object. It consists of differential expression results from `DESeq2` `edgeR` and `limma`-voom.

Source

Example data from the `iCOBRA` package.

Examples

```
data(sb)
```

SummarizedBenchmark *Constructor function for SummarizedBenchmark objects.*

Description

Function to construct SummarizedBenchmark objects.

Usage

```
SummarizedBenchmark(assays, colData, ftData = NULL, groundTruth = NULL,
  performanceMetrics = NULL, BenchDesign = NULL, ...)
```

Arguments

assays	A list containing outputs of the methods to be benchmark. Each element of the list must contain a matrix or data.frame of n x m, n being the number of features tested (e.g. genes) and m being the number of methods in the benchmark. Each element of the list must contain a single assay (the outputs of the methods). For example, for a benchmark of differential expression methods, one assay could contain the q-values from the different methods and another assay could be the estimated log fold changes.
colData	A DataFrame describing the annotation of the methods. These could include version of the software or the parameters used to run them.
ftData	A DataFrame object describing the rows. This parameter is equivalent to the parameter rowData of a SummarizedExperiment.
groundTruth	If present, a DataFrame containing the ground truths. If provided, the number of columns must be the same as the number of assays (NA's are accepted). The names of the columns should have the same names as the assays.
performanceMetrics	A SimpleList of the same length as the number of assays. Each element of the list must be a list of functions. Each function must contain the parameters 'query' and 'truth'.
BenchDesign	A BenchDesign containing the code used to construct the object. (default = NULL)
...	Additional parameters passed to SummarizedExperiment.

Value

A SummarizedBenchmark object.

Author(s)

Alejandro Reyes

Examples

```
## loading the example data from iCOBRA
library(iCOBRA)
data(cobradata_example)
```

```
## a bit of data wrangling and reformatting
assays <- list(
    qvalue=cobradata_example@padj,
    logFC=cobradata_example@score )
assays[["qvalue"]]$DESeq2 <- p.adjust(cobradata_example@pval$DESeq2, method="BH")
groundTruth <- DataFrame( cobradata_example@truth[,c("status", "logFC")] )
colnames(groundTruth) <- names( assays )
colData <- DataFrame( method=colnames(assays[[1]]) )
groundTruth <- groundTruth[rownames(assays[[1]]),]

## constructing a SummarizedBenchmark object
sb <- SummarizedBenchmark(
    assays=assays, colData=colData,
    groundTruth=groundTruth )
colData(sb)$label <- rownames(colData(sb))
```

SummarizedBenchmark-accessors *Accessor and replacement functions for the slots of a SummarizedBenchmark object.*

Description

Accessor and replacement functions for the slots of a SummarizedBenchmark object.

Usage

```
groundTruths(object, ...)

groundTruths(object, ...) <- value

## S4 replacement method for signature 'SummarizedBenchmark,character'
assayNames(x, ...) <- value

## S4 replacement method for signature 'SummarizedBenchmark'
mcols(x, ...) <- value

## S4 method for signature 'SummarizedBenchmark'
groundTruths(object, ...)

## S4 replacement method for signature 'SummarizedBenchmark'
groundTruths(object, ...) <- value

## S4 method for signature 'SummarizedBenchmark'
BDMethodList(..., x = NULL)

## S4 method for signature 'SummarizedBenchmark'
BenchDesign(methods, data)
```

Arguments

object	a SummarizedBenchmark object.
...	Futher arguments, perhaps used by methods
value	A character vector.
x	A SummarizedBenchmark object.
methods	A SummarizedBenchmark object.
data	A data set to be used.

Value

Either a SummarizedBenchmark object or a slot of the SummarizedBenchmark object.

Author(s)

Alejandro Reyes

See Also

performanceMetrics

Examples

```
data( sb )
assayNames( sb )[2] <- "log2FC"
```

SummarizedBenchmark-class

SummarizedBenchmark class documentation

Description

Extension of the RangedSummarizedExperiment to store the output of different methods intended for the same purpose in a given dataset. For example, a differential expression analysis could be done using **limma**-voom, **edgeR** and **DESeq2**. The SummarizedBenchmark class provides a framework that is useful to store, benckmark and compare results.

Slots

performanceMetrics A SimpleList of the same length as the number of assays containing performance functions to be compared with the ground truths.

BenchDesign A BenchDesign originally used to generate the results in the object.

Author(s)

Alejandro Reyes

Examples

```
## loading the example data from iCOBRA
library(iCOBRA)
data(cobradata_example)

## a bit of data wrangling and reformatting
assays <- list(
    qvalue=cobradata_example@padj,
    logFC=cobradata_example@score )
assays[["qvalue"]]$DESeq2 <- p.adjust(cobradata_example@pval$DESeq2, method="BH")
groundTruth <- DataFrame( cobradata_example@truth[,c("status", "logFC")] )
colnames(groundTruth) <- names( assays )
colData <- DataFrame( method=colnames(assays[[1]]) )
groundTruth <- groundTruth[rownames(assays[[1]]),]

## constructing a SummarizedBenchmark object
sb <- SummarizedBenchmark(
    assays=assays, colData=colData,
    groundTruth=groundTruth )
```

---

tdat *Example data.frame containing results for 50 two-sample t-tests.*

Description

Example data.frame containing results for 50 two-sample t-tests.

Format

a data.frame that contains the results of 50 simulated two-sample t-tests, with each row corresponding to an independent test. The data.frame includes the following 5 columns: 1. H = binary 0/1 whether data for the test was simulated under the null (0) or alternative (1) 2. test_statistic = test-statistics of the t-test 3. effect_size = mean difference between the two sample groups 4. pval = p-value of the t-test 5. SE = standard error of the t-test

Examples

```
data(tdat)
```

---

tidyBDMethod *Tidy BDMethod Data*

Description

A helper function to extract information for a single or multiple BDMethod object or a list of BDMethod objects.

Usage

```
tidyBDMethod(obj, dat = NULL, eval = FALSE, label = FALSE)

## S4 method for signature 'BDMethod'
tidyBDMethod(obj, dat, eval)

## S4 method for signature 'list'
tidyBDMethod(obj, dat = NULL, eval = FALSE,
  label = FALSE)

## S4 method for signature 'SimpleList'
tidyBDMethod(obj, dat = NULL, eval = FALSE,
  label = FALSE)

## S4 method for signature 'BenchDesign'
tidyBDMethod(obj, dat = NULL, eval = FALSE,
  label = FALSE)
```

Arguments

obj	BDMethod object, list/List of BDMethod objects (e.g. a BDMethodList), or a BenchDesign object
dat	optional data object to use when evaluating any unevaluated expressionsin bdm meta data. (default = NULL)
eval	logical whether to evaluate any quosures in the meta slot of the BDMethod objects. (default = FALSE)
label	logical whether to add a "label" column to the resulting table containing the names of the methods if obj was specified as a named list. (default = FALSE)

Details

If any quosures are specified to the "meta" slot of a BDMethod object, the quosure is converted to a text string using `rlang::quo_text`.

Value

A named vector of meta data if only a single BDMethod object specified, else a tibble of meta data for the specified list of methods.

Author(s)

Patrick Kimes

---

tidyUpMetrics *Reformat performance metrics to a long format.*

Description

This function takes as input a `SummarizedBenchmark` object, extracts the estimated performance metrics and reformats them into a long-formated data frame.

Usage

```
tidyUpMetrics(object)
```

Arguments

object	A SummarizedBenchmark object.

Value

A tidy data.frame

Author(s)

Alejandro Reyes

Examples

```
data( "sb", package="SummarizedBenchmark" )
sb <- estimateMetricsForAssay( sb, assay="qvalue", evalMetric="rejections",
    evalFunction=function( query, truth, alpha=0.1 ){
        sum( query < alpha )
    },
    addColData=TRUE )
tidyUpMetrics( sb )
```

---

updateBench *Update SummarizedBenchmark object*

Description

Update SummarizedBenchmark results with new methods.

Usage

```
updateBench(sb, bd = NULL, dryrun = TRUE, version = FALSE,
  keepAll = TRUE, reuseParams = TRUE, ...)
```

Arguments

sb	a SummarizedBenchmark object
bd	a BenchDesign object
dryrun	logical whether to just print description of what would be updated rather than actually running any methods. (default = TRUE)
version	logical whether to re-run methods with only package version differences. (default = FALSE)
keepAll	logical whether to keep methods run in original SummarizedBenchmark but not in new BenchDesign. Only used if bd is not NULL. (default = TRUE)
reuseParams	logical whether to reuse parameters from buildBench call used to create SummarizedBenchmark object (if available). Directly specified buildBench parameters still take precedence. (default = TRUE)
...	optional parameters to pass to buildBench.

Value

SumamrizedBenchmark object.

Author(s)

Patrick Kimes

Index

∗Topic **SummarizedBenchmark**
sb, 25
∗Topic **data**
allSB, 5
sb, 25
tdat, 29

addMethod, 2
addPerformanceMetric, 4, *17*, *22*
allSB, 5
allSB,quantSB *(*allSB*)*, 5
assayNames *(*SummarizedBenchmark-accessors*)*, 27
assayNames,SummarizedBenchmark-method *(*SummarizedBenchmark-accessors*)*, 27
assayNames<-,SummarizedBenchmark,character-method *(*SummarizedBenchmark-accessors*)*, 27
assays, *28*
availableMetrics, 5

BDData, 6
BDData,ANY-method *(*BDData*)*, 6
BDData,BDData-method *(*BDData*)*, 6
BDData,BenchDesign-method *(*BDData*)*, 6
BDData,SummarizedBenchmark-method *(*BDData*)*, 6
BDData-class, 6
BDData<- *(*BenchDesign-accessors*)*, 11
BDData<-,BenchDesign,BDDataOrNULL-method *(*BenchDesign-accessors*)*, 11
BDMethod, 7
BDMethod,BDMethodList-method *(*BDMethodList-accessors*)*, 9
BDMethod,BenchDesign-method *(*BenchDesign-accessors*)*, 11
BDMethod,function-method *(*BDMethod*)*, 7
BDMethod,quosure-method *(*BDMethod*)*, 7
BDMethod-class, 8
BDMethod<- *(*BenchDesign-accessors*)*, 11
BDMethod<-,BDMethodList,character,BDMethod-method *(*BDMethodList-accessors*)*, 9
BDMethod<-,BDMethodList,character,NULL-method *(*BDMethodList-accessors*)*, 9
BDMethod<-,BenchDesign,character,BDMethod-method *(*BenchDesign-accessors*)*, 11
BDMethod<-,BenchDesign,character,NULL-method *(*BenchDesign-accessors*)*, 11
BDMethodList, 8
BDMethodList,ANY-method *(*BDMethodList*)*, 8
BDMethodList,BenchDesign-method *(*BenchDesign-accessors*)*, 11
BDMethodList,SummarizedBenchmark-method *(*SummarizedBenchmark-accessors*)*, 27
BDMethodList-accessors, 9
BDMethodList-class, 10
BDMethodList<- *(*BenchDesign-accessors*)*, 11
BDMethodList<-,BenchDesign,BDMethodList-method *(*BenchDesign-accessors*)*, 11
BenchDesign, 10, *26*, *28*
BenchDesign,ANY-method *(*BenchDesign*)*, 10
BenchDesign,SummarizedBenchmark-method *(*SummarizedBenchmark-accessors*)*, 27
BenchDesign-accessors, 11
BenchDesign-class, 12
buildBench, 12, *31*

coerce *(*BDMethodList*)*, 8
colData, *18*
compareBDData, 14
compareBDMethod, 15
compareBenchDesigns, 15
compareBenchDesigns,BenchDesign,BenchDesign-method *(*compareBenchDesigns*)*, 15
compareBenchDesigns,BenchDesign,SummarizedBenchmark-met *(*compareBenchDesigns*)*, 15
compareBenchDesigns,SummarizedBenchmark,BenchDesign-met *(*compareBenchDesigns*)*, 15
compareBenchDesigns,SummarizedBenchmark,missing-method *(*compareBenchDesigns*)*, 15
compareBenchDesigns,SummarizedBenchmark,SummarizedBench *(*compareBenchDesigns*)*, 15

data.frame, *18*
DataFrame, *18*, *26*
dropMethod, 16

estimateMetricsForAssay, 17
estimatePerformanceMetrics, *18*, *22*
estimatePerformanceMetrics
(estimateMetricsForAssay), 17
expandMethod, 19

groundTruths
(SummarizedBenchmark-accessors), 27
groundTruths,SummarizedBenchmark-method
(SummarizedBenchmark-accessors), 27
groundTruths<-
(SummarizedBenchmark-accessors), 27
groundTruths<-,SummarizedBenchmark-method
(SummarizedBenchmark-accessors), 27

hashBDData, 20
hashBDData,BDData-method (hashBDData), 20
hashBDData,BenchDesign-method
(hashBDData), 20

mcols<-,SummarizedBenchmark-method
(SummarizedBenchmark-accessors), 27
modifyMethod, 21

performanceMetrics, 22, *28*
performanceMetrics,SummarizedBenchmark-method
(performanceMetrics), 22
performanceMetrics<-
(performanceMetrics), 22
performanceMetrics<-,SummarizedBenchmark,SimpleList-method
(performanceMetrics), 22
plotMethodsOverlap, 23
plotROC, 24
printMethod, 24
printMethods (printMethod), 24

RangedSummarizedExperiment, *28*

sb, 25
show,BDData-method (BDData-class), 6
show,BDMethod-method (BDMethod-class), 8
show,BDMethodList-method
(BDMethodList-class), 10
show,BenchDesign-method
(BenchDesign-class), 12
SimpleList, *26*, *28*
SummarizedBenchmark, *4*, *17*, *18*, *23*, *24*, *26*, 26, *31*
SummarizedBenchmark-accessors, 27
SummarizedBenchmark-class, 28
SummarizedExperiment, *18*, *26*

tdat, 29
tidyBDMethod, 29
tidyBDMethod,BDMethod-method
(tidyBDMethod), 29
tidyBDMethod,BenchDesign-method
(tidyBDMethod), 29
tidyBDMethod,list-method
(tidyBDMethod), 29
tidyBDMethod,SimpleList-method
(tidyBDMethod), 29
tidyUpMetrics, 30
truthdat (sb), 25
txi (sb), 25

updateBench, 31
upset, *23*