# Package ‘metavizr’

May 3, 2022

**Type** Package

**Version** 1.20.0

**Maintainer** Hector Corrada Bravo `<hcorrada@gmail.com>`

**License** MIT + file LICENSE

**Title** R Interface to the metaviz web app for interactive metagenomics data analysis and visualization

**Description** This package provides Websocket communication to the metaviz web app (http://metaviz.cbcb.umd.edu) for interactive visualization of metagenomics data. Objects in R/bioc interactive sessions can be displayed in plots and data can be explored using a facetzoom visualization. Fundamental Bioconductor data structures are supported (e.g., MRexperiment objects), while providing an easy mechanism to support other data structures. Visualizations (using d3.js) can be easily added to the web app as well.

**VignetteBuilder** knitr

**Depends** R (>= 3.4), metagenomeSeq (>= 1.17.1), methods, data.table, Biobase, digest

**Imports** epivizr, epivizrData, epivizrServer, epivizrStandalone, vegan, GenomeInfoDb, phyloseq, httr

**Suggests** knitr, BiocStyle, matrixStats, msd16s (>= 0.109.1), etec16s, testthat, gss, ExperimentHub, tidyr, rmarkdown

**Collate** 'metavizControl.R' 'startMetaviz.R' 'utils.R' 'EpivizMetagenomicsData-class.R' 'register-methods.R' 'validateMRExperiment.R' 'MetavizApp-class.R' 'MetavizGraph-class.R' 'EpivizMetagenomicsDataInnerNodes-class.R' 'MetavizGraphInnerNodes-class.R' 'EpivizMetagenomicsDataTimeSeries-class.R'

**biocViews** Visualization, Infrastructure, GUI, Metagenomics, ImmunoOncology

**RoxygenNote** 7.1.0

**git_url** https://git.bioconductor.org/packages/metavizr

**git_branch** RELEASE_3_15

**git_last_commit** 596a30e

**git_last_commit_date** 2022-04-26

**Date/Publication** 2022-05-03

**Author** Hector Corrada Bravo [cre, aut],
Florin Chelaru [aut],
Justin Wagner [aut],
Jayaram Kancherla [aut],
Joseph Paulson [aut]

# R topics documented:

---

## `buildMetavizGraph` *Build a MetavizTree object from another object*

---

### Description

Build a MetavizTree object from another object

### Usage

```
buildMetavizGraph(object, ...)

## S4 method for signature 'MRexperiment'
buildMetavizGraph(object, feature_order, ...)
```

## Arguments

| | |
|---|---|
| `object` | The object from which taxonomy data is extracted |
| `...` | Additional arguments |
| `feature_order` | Ordering of leaves (features) in taxonomy tree |

## Value

a `MetavizGraph` object

## Methods (by class)

- `MRexperiment`: Build graph from a `MRexperiment-class` object

---

`buildMetavizGraphInnerNodes`

# *Build a MetavizTree object from another object*

## Description

Build a MetavizTree object from another object

## Usage

```
buildMetavizGraphInnerNodes(object, ...)

## S4 method for signature 'MRexperiment'
buildMetavizGraphInnerNodes(object, feature_order,
  ...)
```

## Arguments

| | |
|---|---|
| `object` | The object from which taxonomy data is extracted |
| `...` | Additional arguments |
| `feature_order` | Ordering of leaves (features) in taxonomy tree |

## Value

a `MetavizGraphInnerNodes` object

## Methods (by class)

- `MRexperiment`: Build graph from a `MRexperiment-class` object

---

`EpivizMetagenomicsData-class`

## *Data container for MRexperiment objects*

---

### Description

Used to serve metagenomic data (used in e.g., icicle plots and heatmaps). Wraps `MRexperiment-class` objects.

### Methods

- `df_to_tree(root, df)` Helper function to recursively build nested response for getHierarchy
  - **root** Root of subtree
  - **df** data.frame containing children to process
- `get_default_chart_type()` Get name of default chart type for this data type
- `get_measurements()` Get description of measurements served by this object
- `getAlphaDiversity(measurements = NULL)` Compute alpha diversity using vegan for the given samples
  - **measurements** Samples to compute alpha diversity
  - **start** Start of feature range to query
  - **end** End of feature range to query
- `getCombined(measurements = NULL, seqName, start = 1, end = 1000, order = NULL, nodeSelection = NULL, select`
  Return the counts aggregated to selected nodes for the given samples
  - **measurements** Samples to get counts for
  - **seqName** name of datasource
  - **start** Start of feature range to query
  - **end** End of feature range to query
  - **order** Ordering of nodes
  - **nodeSelection** Node-id and selectionType pairs
  - **selectedLevels** Current aggregation level
- `getHierarchy(nodeId = NULL)` Retrieve feature hierarchy information for subtree with specified root
  - **nodeId** Feature identifier with level info
- `getPCA(measurements = NULL)` Compute PCA over all features for given samples
  - **measurements** Samples to compute PCA over
  - **start** Start of feature range to query
  - **end** End of feature range to query
- `getRows(measurements = NULL, start = 1, end = 1000, selectedLevels = 3, selections = NULL)`
  Return the sample annotation and features within the specified range and level for a given sample and features

- **measurements** Samples to retrieve for
- **start** Start of feature range to query
- **end** End of feature range to query
- **selections** Node-id and selectionType pairs
- **selectedLevels** Current aggregation level

`getValues(measurements = NULL, start = 1, end = 1000, selectedLevels = 3, selections = NULL, row_order = NUL`
Return the counts for a sample within the specified range

- **measurements** Samples to get counts for
- **start** Start of feature range to query
- **end** End of feature range to query
- **selections** Node-id and selectionType pairs
- **selectedLevels** Current aggregation level

`propagateHierarchyChanges(selection = NULL, order = NULL, selectedLevels = NULL, request_with_labels = FA`
Update internal state for hierarchy

- **selection** Node-id and selectionType pairs
- **order** Ordering of features
- **selectedLevels** Current aggregation level
- **request_with_labels** For handling requests using fData entries from MRexperiment

`row_to_dict(row)` Helper function to format each node entry for getHierarchy response

- **row** Information for current node.

`searchTaxonomy(query = NULL, max_results = 15)` Return list of features matching a text-based query

- **query** String of feature for which to search
- **max_results** Maximum results to return

`toNEO4JDbHTTP(batch_url, neo4juser, neo4jpass, datasource, description = NULL)` Write an ‘EpivizMetagenomicsData‘ object to a Neo4j graph database

@param batch_url (character) Neo4j database url and port for processing batch http requests @param neo4juser (character) Neo4j database user name @param neo4jpass (character) Neo4j database password @param datasource (character) Name of Neo4j datasource node for this ‘EpivizMetagenomicsData‘ object

@examples library(metagenomeSeq) data("mouseData") mobj <- metavizr:::EpivizMetagenomicsData$new(object=mo
mobj$toNEO4JDbHTTP(batch_url = "http://localhost:7474/db/data/batch", neo4juser = "neo4juser", neo4jpass = "neo4jpass", datasource = "mouse_data")

`update(new_object, send_request = TRUE)` Update underlying data object with new object

## Examples

```
## Not run:
library(metagenomeSeq)
data(mouseData)
obj <- metavizr:::EpivizMetagenomicsData$new(mouseData, feature_order = colnames(fData(mouseData)))

## End(Not run)
```

---

`EpivizMetagenomicsDataInnerNodes-class`
*Data container for MRexperiment objects*

---

## Description

Used to serve metagenomic data (used in e.g., icicle plots and heatmaps). Wraps `MRexperiment-class` objects.

## Methods

`df_to_tree(root, df)` Helper function to recursively build nested response for getHierarchy

- **root** Root of subtree
- **df** data.frame containing children to process

`get_default_chart_type()` Get name of default chart type for this data type

`get_measurements()` Get description of measurements served by this object

`getAlphaDiversity(measurements = NULL)` Compute alpha diversity using vegan for the given samples

- **measurements** Samples to compute alpha diversity
- **start** Start of feature range to query
- **end** End of feature range to query

`getCombined(measurements = NULL, seqName, start = 1, end = 1000, order = NULL, nodeSelection = NULL, select`
Return the counts aggregated to selected nodes for the given samples

- **measurements** Samples to get counts for
- **seqName** name of datasource
- **start** Start of feature range to query
- **end** End of feature range to query
- **order** Ordering of nodes
- **nodeSelection** Node-id and selectionType pairs
- **selectedLevels** Current aggregation level

`getHierarchy(nodeId = NULL)` Retrieve feature hierarchy information for subtree with specified root

- **nodeId** Feature identifier with level info

`getPCA(measurements = NULL)` Compute PCA over all features for given samples

- **measurements** Samples to compute PCA over
- **start** Start of feature range to query
- **end** End of feature range to query

`getRows(measurements = NULL, start = 1, end = 1000, selectedLevels = 3, selections = NULL)`
Return the sample annotation and features within the specified range and level for a given sample and features

**measurements** Samples to retrieve for

**start** Start of feature range to query

**end** End of feature range to query

**selections** Node-id and selectionType pairs

**selectedLevels** Current aggregation level

`getValues(measurements = NULL, start = 1, end = 1000, selectedLevels = 3, selections = NULL)` Return the counts for a sample within the specified range

**measurements** Samples to get counts for

**start** Start of feature range to query

**end** End of feature range to query

**selections** Node-id and selectionType pairs

**selectedLevels** Current aggregation level

`propagateHierarchyChanges(selection = NULL, order = NULL, selectedLevels = NULL, request_with_labels = FA` Update internal state for hierarchy

**selection** Node-id and selectionType pairs

**order** Ordering of features

**selectedLevels** Current aggregation level

**request_with_labels** For handling requests using fData entries from MRexperiment

`row_to_dict(row)` Helper function to format each node entry for getHierarchy response

**row** Information for current node.

`searchTaxonomy(query = NULL, max_results = 15)` Return list of features matching a text-based query

**query** String of feature for which to search

**max_results** Maximum results to return

## Examples

```
## Not run:
library(curatedMetagenomicData)
zeller.eset = ZellerG_2014.metaphlan_bugs_list.stool()
zeller_MR <- ExpressionSet2MRexperiment(zeller.eset)
feature_order <- colnames(fData(zeller_MR))
sampleId<- "CCIS98482370ST-3-0"
mObj <- metavizr:::EpivizMetagenomicsDataInnerNodes$new(zeller_MR, feature_order = feature_order)

## End(Not run)
```

---

`EpivizMetagenomicsDataTimeSeries-class` *Data container for MRexperiment objects*

---

## Description

Used to serve metagenomic data (used in e.g., icicle plots and heatmaps). Wraps `MRexperiment-class` objects.

## Examples

```
## Not run:
library(metagenomeSeq)
data(mouseData)
obj <- metavizr:::EpivizMetagenomicsData$new(mouseData, feature_order = colnames(fData(mouseData)))

## End(Not run)
```

---

`generateSelection` *Method to select and set aggregation type to nodes in FacetZoom*

---

## Description

Method to select and set aggregation type to nodes in FacetZoom

## Usage

```
generateSelection(feature_names, aggregation_level, selection_type,
  feature_order = NULL)
```

## Arguments

| | |
|---|---|
| `feature_names` | Selected Features |
| `aggregation_level` | Level in the hierarchy |
| `selection_type` | Expanded, aggregated, or removed |
| `feature_order` | Order of features at that level |

## Value

A selection object for a metavizControl object to accept

## Examples

```
generateSelection("Bacteroidales", 1L, 2L)
```

---

## `MetavizApp-class` *Class managing connection to metaviz application.*

---

### Description

Class managing connection to metaviz application.

---

## `metavizControl` *metavizr settings*

---

### Description

Default settings for the various plotting functions in metavizr.

### Usage

```
metavizControl(aggregateAtDepth = 3, aggregateFun = function(x)
  colSums(x), valuesAnnotationFuns = NULL, maxDepth = 4,
  maxHistory = 3, maxValue = NULL, minValue = NULL, title = "",
  n = 10000, rankFun = stats::sd, norm = TRUE, log = FALSE,
  featureSelection = NULL)
```

### Arguments

| | |
|---|---|
| `aggregateAtDepth` | Level of the tree to aggregate counts at by default. |
| `aggregateFun` | Function to aggregate counts by at the aggregateAtDepth level. |
| `valuesAnnotationFuns` | Function for error bars. |
| `maxDepth` | Level of the tree to display by default in icicle view. |
| `maxHistory` | Value for caching. |
| `maxValue` | Maximum value to display. |
| `minValue` | Minimum value to display. |
| `title` | title. |
| `n` | Number of OTUs to include in ranking. |
| `rankFun` | Ranking function - single vector function. |
| `norm` | Normalize MRexperiment object. |
| `log` | Log tranformation of MRexperiment object. |
| `featureSelection` | List of features to set as nodeSelections |

### Value

List of setting parameters.

### Examples

```
settings = metavizControl()
```

---

## `MetavizGraph-class` *Graph implementation to query hierarchical feature data*

### Description

Used to manage aggregation and range queries from the Metaviz app UI.

---

## `MetavizGraphInnerNodes-class` *Graph implementation to query hierarchical feature data*

### Description

Used to manage aggregation and range queries from the Metaviz app UI.

---

## `register,MRexperiment-method` *Generic method to register data to the epiviz data server*

### Description

Generic method to register data to the epiviz data server

### Usage

```
## S4 method for signature 'MRexperiment'
register(object, type = "LeafCounts",
  columns = NULL, ...)
```

### Arguments

| | |
|---|---|
| `object` | The object to register to data server |
| `type` | leafCounts, if data objects has counts at leaf level or innerNodeCounts, if data object has counts at inner nodes |
| `columns` | Name of columns containing data to register |
| `...` | Additonal arguments passed to object constructors |

### Value

An `EpivizMetagenomicsData-class` object

---

## `register,phyloseq-method` *Generic method to register data to the epiviz data server*

### Description

Generic method to register data to the epiviz data server

### Usage

```
## S4 method for signature 'phyloseq'
register(object, type = "LeafCounts", ...)
```

### Arguments

| | |
|---|---|
| `object` | The object to register to data server |
| `type` | leafCounts, if data objects has counts at leaf level or innerNodeCounts, if data object has counts at inner nodes |
| `...` | Additonal arguments passed to object constructors |

### Value

An `phyloseq-class` object

---

## `replaceNAFeatures` *Method to replace NA or null feature labels with Not_Annotated_hierarchy-level*

### Description

Method to replace NA or null feature labels with Not_Annotated_hierarchy-level

### Usage

```
replaceNAFeatures(replacing_na_obj_fData, feature_order)
```

### Arguments

| | |
|---|---|
| `replacing_na_obj_fData` | fData from MRexperiment object to replace NA or null |
| `feature_order` | Order of features |

### Value

hierarchy with NA or null feature labels replaced

### Examples

```
library(metagenomeSeq)
data(mouseData)
feature_order <- colnames(fData(mouseData))
replaceNAFeatures(fData(mouseData),feature_order)
```

---

## `setMetavizStandalone` *set metaviz app standalone settings*

### Description

set metaviz app standalone settings

### Usage

```
setMetavizStandalone(url = "https://github.com/epiviz/epiviz.git",
  branch = "metaviz-4.1", local_path = NULL, non_interactive = FALSE)
```

### Arguments

| | |
|---|---|
| `url` | (character) github url to use. defaults to (`"https://github.com/epiviz/epiviz.git"`). |
| `branch` | (character) branch on the github repository. defaults to (master). |
| `local_path` | (character) if you already have a local instance of metaviz and would like to run standalone use this. |
| `non_interactive` | (logical) don't download repo, used for testing purposes. |

### Value

path to the metaviz app git repository

### Examples

```
## Not run:
#' # see package vignette for example usage
setMetavizStandalone()

## End(Not run)
```

---

## `startMetaviz` *Start metaviz app and create `MetavizApp` object to manage connection.*

---

### Description

Start metaviz app and create `MetavizApp` object to manage connection.

### Usage

```
startMetaviz(host = "http://metaviz.cbcb.umd.edu",
  register_function = .register_all_metaviz_things, ...)
```

### Arguments

| | |
|---|---|
| `host` | (character) host address to launch. |
| `register_function` | (function) function used to register actions and charts on the metaviz app. |
| `...` | additional parameters passed to `startEpiviz`. |

### Value

An object of class `MetavizApp`

### See Also

`MetavizApp`

### Examples

```
# see package vignette for example usage
app <- startMetaviz(non_interactive=TRUE, open_browser=FALSE)
app$stop_app()
```

---

## `startMetavizStandalone` *Start metaviz app in standalone (locally) and create `MetavizApp` object to manage connection.*

---

### Description

Start metaviz app in standalone (locally) and create `MetavizApp` object to manage connection.

### Usage

```
startMetavizStandalone(register_function = .register_all_metaviz_things,
  use_viewer_option = FALSE, ...)
```

### Arguments

| | |
|---|---|
| `register_function` | (function) function used to register actions and charts on the metaviz app. |
| `use_viewer_option` | (function) run application in viewer defined by `getOption("viewer")`. This allows standalone app to run in Rstudio's viewer (FALSE by default) |
| `...` | additional parameters passed to `startStandalone`. |

### Value

An object of class `MetavizApp`

### Examples

```
#' # see package vignette for example usage
app <- startMetavizStandalone(non_interactive=TRUE)
app$stop_app()
```

---

## `validateObject` *validate* `MRexperiment-class` *object*

### Description

validate `MRexperiment-class` object

### Usage

```
validateObject(object)
```

### Arguments

| | |
|---|---|
| `object` | an object of class `MRexperiment-class` |

### Value

TRUE or FALSE

### Examples

```
library(metagenomeSeq)
data(mouseData)
validateObject(mouseData)
```

# Index