Package ‘BIOM.utils’

August 29, 2016

Type Package
Title Utilities for the BIOM (Biological Observation Matrix) Format
Version 0.9
Depends R (>= 3.0), utils
Imports
Suggests RJSONIO, MGRASTer
URL https://github.com/braithwaite/BIOM.utils/
Date 2014-08-22
Description Provides utilities to facilitate import, export and computation with the
BIOM (Biological Observation Matrix) format (http://biom-format.org).
License BSD_2_clause + file LICENSE
Copyright University of Chicago
LazyData yes
Collate source.R
Author Daniel T. Braithwaite [aut, cre]
Maintainer Daniel T. Braithwaite <contact.dtb@gmail.com>
NeedsCompilation no
Repository CRAN
Date/Publication 2014-08-29 01:18:59

R topics documented:

BIOM Constants .................................................. 2
BIOM Dimensions .................................................. 3
BIOM Display ..................................................... 5
BIOM Examples .................................................. 7
BIOM Export ..................................................... 8
BIOM Import ..................................................... 11
Matrix Manipulations for BIOM Format .................................. 13

Index 15
BIOM Constants

| BIOM Constants | Constants related to BIOM format |

Description

These constants enumerate components of BIOM format and their valid values (controlled vocabularies).

Usage

```r
biom_format
biom_format_url
biom_fields
biom_table_types
biom_matrix_element_types
biom_matrix_types
```

Details

BIOM (Biological Observation Matrix) is a simple prescription for storing an annotated table of data. It may be described as a format, standard, or data structure.

The JSON (JavaScript Object Notation) standard for expressing general data objects as text is employed to define BIOM. Therefore the native form of BIOM data is structured text, conforming to the JSON specification in general and the BIOM specification in particular. Familiarity with BIOM is assumed here.

The S3 class `biom` and its methods facilitate analyses by expressing BIOM data as objects in the R environment.

`biom_format` gives the latest BIOM version implemented by this package.

`biom_format_url` gives its associated URL.

`biom_fields` lists the components specified by BIOM.

Each other constant enumerates the allowed values of its corresponding BIOM component.

References

BIOM format
JSON

See Also

biom
**BIOM Dimensions**

**Examples**

- `biom_format_url [biom_format]`
- `biom_fields`
- `biom_table_types`
- `biom_matrix_element_types`
- `biom_matrix_types`

---

**BIOM Dimensions**  
*Row and column information of BIOM data*

---

**Description**

Report shape, unique row and column ids, and row and column metadata annotations of an object of class `biom`.

**Usage**

```r
## S3 method for class 'biom'
dim(x)
```

```r
## S3 method for class 'biom'
dimnames(x)
```

```r
metadata(x, ...)
```

```r
## S3 method for class 'biom'
metadata(x, ...)
```

**Arguments**

- `x`  
an object (biom)
- `...`  
unused

**Details**

BIOM (Biological Observation Matrix) is a simple prescription for storing an annotated table of data. It may be described as a format, standard, or data structure.

The JSON (JavaScript Object Notation) standard for expressing general data objects as text is employed to define BIOM. Therefore the native form of BIOM data is structured text, conforming to the JSON specification in general and the BIOM specification in particular. Familiarity with BIOM is assumed here.

The S3 class `biom` and its methods facilitate analyses by expressing BIOM data as objects in the R environment. The functions above apply to an R object that is already of class `biom`.

- `dim()` returns the dimensions of the BIOM data table (its shape).
- `dimnames()` returns the unique row and column ids required by BIOM.
- `metadata()` returns the row and column metadata annotations provided for by BIOM. They may be empty.
Value

For `dim()`, a length-two integer vector, with additional attribute "nnz" (number not zero) if the object is "sparse", equaling the number of values (rows) in the sparse representation. See note below.

For `dimnames()`, a list of two character vectors, named "rows" and "columns".

For `metadata()`, a list (invisibly) of two components, named "rows" and "columns". Each is a list equal in length to the data table's corresponding dimension.

Note

When x is "sparse" the dimensions of `as.matrix(x)` are not given by `dim(x)`. Rather, the latter has dimensions c("nnz", 3). See above.

Also note that BIOM requires no exact structure for the metadata annotations of each row (column). All that is guaranteed is a list (possibly of empty, atomic, or nested list elements) with as many entries as the data table has rows (columns).

Author(s)

Daniel T. Braithwaite

References

- BIOM format
- JSON

See Also

- `biom`, `as.matrix.biom`

Examples

```r
## one toy example, one real example:
dd <- biom (dmat, quiet=TRUE)
ff <- biom (li4, quiet=TRUE)

dim (dd)
dim (ff)
dim (as.matrix (ff))
dim (as.matrix (ff, expand=TRUE))

dimnames (dd)
dimnames (ff)

## automatic row and column ids:
dimnames (biom (smat, sparse=TRUE, quiet=TRUE))

## no metadata:
print (metadata (dd))
```
## BIOM Display

*Display BIOM data in full or part*

### Description

Nicely print or summarize an object of class `biom`. A summary omits printing the data table.

### Usage

```r
## S3 method for class 'biom'
str(object, ...)

## S3 method for class 'biom'
summary(object, ...)

## S3 method for class 'biom'
print(x, ...)

## S3 method for class 'biom'
head(x, n=5, p=n, ...)

## S3 method for class 'biom'
tail(x, n=5, p=n, ...)
```

### Arguments

- `x` an object (`biom`)
- `object` an object (`biom`)
- `n` number of rows (single integer)
- `p` number of columns (single integer)
- `...` further arguments to default method (`str()` only)

### Details

BIOM (Biological Observation Matrix) is a simple prescription for storing an annotated table of data. It may be described as a format, standard, or data structure.

The JSON (JavaScript Object Notation) standard for expressing general data objects as text is employed to define BIOM. Therefore the native form of BIOM data is structured text, conforming to
the JSON specification in general and the BIOM specification in particular. Familiarity with BIOM is assumed here.

The S3 class `biom` and its methods facilitate analyses by expressing BIOM data as objects in the R environment. The functions above apply to an R object that is already of class `biom`.

`print()` and `summary()` show the contents of the object, with the difference that the latter omits printing the BIOM data table. The former always prints it in expanded (non-sparse) form.

`str()` shows the raw R structure of its argument using the default `str()` with pleasing options.

`head()` and `tail()` return the data table’s initial or final few rows and columns.

**Value**

A matrix for `head()` and `tail()`. See note below.

For the others, nothing useful.

**Note**

When the object is "sparse" its data table is stored as a three-column sparse representation matrix. (See reference for details, and note that indices begin at zero.) Such data is displayed by `print()` in an expanded (non-sparse) form, but `head()` and `tail()` return parts of the sparse representation unexpanded.

**Author(s)**

Daniel T. Braithwaite

**References**

BIOM format

JSON

**See Also**

`biom`, `dim.biom` `as.matrix.biom`

**Examples**

```r
# one toy example, one real example:
xx <- biom (dmat, quiet=TRUE)
yy <- biom (l14)

summary (xx)
print (xx)
head (xx)
tail (xx)
tail (xx, n=10, p=3)

tail (yy)
tail (yy, n=15)
```
BIOM Examples

\#

str(yy)

## Description

Example objects for getting started, and related utilities.

### Usage

```r
jtxt; smat; dmat; li1; li2; li3; li4
e.exampleBiomFile()
applyBiomMethods(x)
b.buildBiomExamples(rdafile="examples.rda", jsonfile="example-json.txt")
```

### Arguments

- `rdafile`: filename for .rda saves of example objects (string)
- `jsonfile`: filename for saved example JSON text (string)
- `x`: object (biom)

### Details

BIOM (Biological Observation Matrix) is a simple prescription for storing an annotated table of data. It may be described as a format, standard, or data structure.

The JSON (JavaScript Object Notation) standard for expressing general data objects as text is employed to define BIOM. Therefore the native form of BIOM data is structured text, conforming to the JSON specification in general and the BIOM specification in particular. Familiarity with BIOM is assumed here.

The S3 class `biom` and its methods facilitate analyses by expressing BIOM data as objects in the R environment. Simple data objects of several kinds are provided to experiment with handling BIOM in R. See the examples below.

`buildBiomExamples()` draws from MG-RAST (see below) to construct these objects and saves them in designated files. (This utility is used to build the package.)

`exampleBiomFile()` returns the path to a file of JSON text correctly structured as BIOM.

`applyBiomMethods()` is a utility for testing that simply applies all `biom` methods to a given object.

### Value

Only `exampleBiomFile()` has a useful return value, as described above.
BIOM Export

Convert BIOM data from formal to basic type (export)

Description

Convert an object of class biom into a basic type: matrix, list, or character.
BIOM Export

Usage

```r
## S3 method for class 'biom'
as.matrix(x, expand=NULL, ...)

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

## S3 method for class 'biom'
as.character(x, ..., file=NULL)
```

Arguments

- `x` an object (biom)
- `expand` if sparse, force return of not sparse matrix? (logical)
- `...` unused
- `file` filename for writing object

Details

BIOM (Biological Observation Matrix) is a simple prescription for storing an annotated table of data. It may be described as a format, standard, or data structure. The JSON (JavaScript Object Notation) standard for expressing general data objects as text is employed to define BIOM. Therefore the native form of BIOM data is structured text, conforming to the JSON specification in general and the BIOM specification in particular. Familiarity with BIOM is assumed here.

The S3 class `biom` and its methods facilitate analyses by expressing BIOM data as objects in the R environment. Each function above transforms an object that is already of class `biom` into a basic R type.

- `as.matrix()` returns the BIOM data table as a matrix. If the object is "dense", then `dimnames()` of the result are equal to the BIOM row and column ids. Otherwise, the three-column sparse representation matrix is returned, with ids given by attached attributes "rownames" and "colnames".

- However, using `expand=TRUE` expands a sparse representation. (Setting `expand=FALSE` has no effect when the object is "dense".) Also, see below for an example using class "sparseMatrix" from the R package Matrix.

- `as.character()` returns BIOM properly speaking, that is, data and annotations written in JSON text conforming to the BIOM specification. That text is written to file, if provided.

- See below for an example of saving the data table, only, in CSV or TSV format. Of course, it is possible to bring `biom` objects in and out of the session with `save()` and `load()`.

- `as.list()` returns a list corresponding closely element-by-element to BIOM. The differences are: list element data is a matrix not list; elements rows and columns hold metadata only, and so do not include ids; instead, additional elements `row.ids` and `column.ids` are present; and `format_url` is missing.
Value

For `as.matrix()` and `as.list()`, as described.

For `as.character()`, a single string of JSON text, or simply file if it is not NULL.

Except that last case, these functions return invisibly.

Author(s)

Daniel T. Braithwaite

References

BIOM format
JSON

See Also

biom

Examples

\[
\begin{align*}
\text{tt} & \leftarrow \text{tempfile()} \\
\text{xx} & \leftarrow \text{biom (li3)} \\
\text{yy} & \leftarrow \text{biom (smat, sparse=TRUE, quiet=TRUE)}
\end{align*}
\]

## extract objects of basic types:
print (as.matrix (xx))
head (as.matrix (yy, expand=TRUE))
as.character (xx)
as.character (xx, file=tt)
str (as.list (xx))

## export to a CSV file:
write.table (as.matrix (xx), file=tt, sep=",")

## Not run:
## a classed sparse matrix (for computation or what have you):
\[
\begin{align*}
\text{zz} & \leftarrow \text{as.matrix (yy)} \\
\text{zz}[1:2] & \leftarrow 1 + \text{zz}[1:2] \\
\text{Matrix}::\text{sparseMatrix (i=zz[,1], j=zz[,2], x=zz[,3])}
\end{align*}
\]

## End(Not run)
unlink(tt)
Description

Construct an object of class `biom` from given data, supplementing missing fields as necessary.

Usage

```r
biom(x, ...)  
## S3 method for class 'character'
biom(x, ..., file=NULL, quiet=FALSE)

## S3 method for class 'matrix'
biom(x, type=biom_table_types, sparse=NULL, ..., quiet=FALSE)

## S3 method for class 'list'
biom(x, ..., quiet=FALSE)
```

Arguments

- `x`: an object convertible to `biom`
- `type`: the type of BIOM table to be constructed (string)
- `sparse`: dims or dimnames when sparse (integer or list, length-two)
- `...`: arguments to `fromjson()`
- `file`: file containing JSON text for BIOM data (string)
- `quiet`: print messages and warnings? (logical)

Details

BIOM (Biological Observation Matrix) is a simple prescription for storing an annotated table of data. It may be described as a format, standard, or data structure.

The JSON (JavaScript Object Notation) standard for expressing general data objects as text is employed to define BIOM. Therefore the native form of BIOM data is structured text, conforming to the JSON specification in general and the BIOM specification in particular. Familiarity with BIOM is assumed here.

The S3 class `biom` and its methods facilitate analyses by expressing BIOM data as objects in the R environment. Each function above transforms an object of a basic R type into class `biom`.

The character method is suitable for importing BIOM data properly speaking, that is, JSON text conforming to the BIOM specification. The data is read from `file`, when provided. Arguments from `...` are passed to `fromJSON()` in order to allow for different character encodings.

To make a `biom` object from a TSV or CSV file, see examples below. Of course, it is possible to bring `biom` objects in and out of the session with `save()` and `load()`.
The matrix method accepts an ordinary ("dense") matrix or a three-column sparse matrix representation (using indices starting at zero, per BIOM). If the latter, sparse should also be provided, as a length-two integer giving the data table’s shape, or a length-two list giving its row and column ids. It may also be simply TRUE, in which case the smallest possible shape is assigned.

The list method accepts a list of components allowed or required by BIOM, inventing something reasonable for missing components, with certain qualifications:

- shape is required if matrix_type is specified as "sparse"; data may be given as a list of rows, a list of triples sparsely representing a matrix, or a matrix; in the last case, its dimnames() may be used to provide row and column ids; but ids from rows and columns, if provided, supercede dimnames().

Value

An object of class biom, invisibly.

Note

A biom object reflects its matrix_element_type implicitly with the storage.mode() of its data. Accordingly, applying storage.mode<-() prior to biom() can be useful. See examples below.

Also, note that values seeming to be integers in R often are not.

Author(s)

Daniel T. Braithwaite

References

BIOM format
JSON

See Also

as.character.biom, as.matrix.biom, dim.biom, print.biom, storage.mode

Examples

tt <- tempfile()

## two ways to the same result:
ff <- exampleBiomFile()
txt <- readLines (ff)
biom (txt)
biom (file=ff)

## choose what fields to include with a list:
biom (list (data=smat, matrix_type="sparse", shape=c(266,4),
matrix_element_type="unicode", comment="no comment"), quiet=TRUE)

xx <- matrix2list (cbind (LETTERS[1:20], paste ("some metadata for row", 1:20)))
Matrix Manipulations for BIOM Format

```r
xx <- lapply(xx, "names"="", c("id", "metadata"))
biom (list (data=dmat, type="Gene table", rows=xx, id="1234567890"), quiet=TRUE)

## the same result in two ways, again:
write.table (dmat, file="tt", sep="","
biom (dmat, "Function table")
biom (as.matrix (read.table (file="tt", sep="","), "Func")

## all the same:
biom (smat, sparse=TRUE, quiet=TRUE)
biom (smat, sparse=c(266,4), quiet=TRUE)
biom (smat, sparse=list (paste ("row", 1:266), paste ("column", 1:4)), quiet=TRUE)

## enforce matrix_element_type to be "int":
mm <- dmat
storage.mode (mm) <- "integer"
biom (mm, quiet=TRUE)

unlink (tt)
```

---

Matrix Manipulations for BIOM Format

*Convert matrix data between dense, sparse, and list representations*

Description

Convert two-dimensional data between ordinary (dense) matrix type, sparse representation, and a list of rows (or columns).

Usage

- `sparse2dense(x, dim=NULL)`
- `dense2sparse(x)`
- `matrix2list(x)`

Arguments

- `x` object to convert (matrix)
- `dim` number of rows and columns (length-two integer)

Details

BIOM (Biological Observation Matrix) is a simple prescription for storing an annotated table of data. It may be described as a format, standard, or data structure.

The JSON (JavaScript Object Notation) standard for expressing general data objects as text is employed to define BIOM. Therefore the native form of BIOM data is structured text, conforming to the JSON specification in general and the BIOM specification in particular. Familiarity with BIOM is assumed here.
The S3 class codebiom and its methods facilitate analyses by expressing BIOM data as objects in the R environment. These manipulations of two-dimensional (matrix) data are sometimes handy when working with BIOM data. The sparse format used by BIOM is conventional, but see the reference for exact details.

All names() and dimnames() are removed.

Value

sparse2dense() and dense2sparse() return a matrix.
matrix2list() returns a list of the rows of x.

Note

Like R these functions begin indices at 1, whereas BIOM counts from 0, so adjust if necessary.

Author(s)

Daniel T. Braithwaite

References

BIOM format
JSON

See Also

biom, as.matrix.biom

Examples

xx <- smat
xx[,1:2] <- xx[,1:2] + 1
sparse2dense (xx)
sparse2dense (xx, c(266,10))

matrix2list (dmat)
matrix2list (t(dmat))
Index

*Topic **datasets**
  BIOM Constants, 2

applyBiomMethods (BIOM Examples), 7
as.character.biom, 12
as.character.biom (BIOM Export), 8
as.list.biom (BIOM Export), 8
as.matrix.biom, 4, 6, 12, 14
as.matrix.biom (BIOM Export), 8

biom, 2, 4, 6, 8, 10, 14
biom (BIOM Import), 11
BIOM Constants, 2
BIOM Dimensions, 3
BIOM Display, 5
BIOM Examples, 7
BIOM Export, 8
BIOM Import, 11
biom_fields (BIOM Constants), 2
biom_format (BIOM Constants), 2
biom_format_url (BIOM Constants), 2
biom_matrix_element_types (BIOM Constants), 2
biom_matrix_types (BIOM Constants), 2
biom_table_types (BIOM Constants), 2
buildBiomExamples (BIOM Examples), 7
dense2sparse (Matrix Manipulations for BIOM Format), 13
dim.biom. 6, 12
dim.biom (BIOM Dimensions), 3
dimnames.biom (BIOM Dimensions), 3
dmat (BIOM Examples), 7
dsum (BIOM Examples), 7

head.biom (BIOM Display), 5
jtxt (BIOM Examples), 7
li1 (BIOM Examples), 7
li2 (BIOM Examples), 7
li3 (BIOM Examples), 7
li4 (BIOM Examples), 7
Matrix Manipulations for BIOM Format, 13
matrix2list (Matrix Manipulations for BIOM Format), 13
metadata (BIOM Dimensions), 3
print.biom, 12
print.biom (BIOM Display), 5
print.biomsummary (BIOM Display), 5

smat (BIOM Examples), 7
sparse2dense (Matrix Manipulations for BIOM Format), 13
storage.mode, 12
str.biom (BIOM Display), 5
summary.biom (BIOM Display), 5
tail.biom (BIOM Display), 5

15