Package 'rmatio'

March 18, 2019

Title Read and Write 'Matlab' Files

Version 0.14.0

Description Read and write 'Matlab' MAT files from R. The 'rmatio' package supports reading MAT version 4, MAT version 5 and MAT compressed version 5. The 'rmatio' package can write version 5 MAT files and version 5 files with variable compression.

Copyright The package includes the source code of matio written by Christopher Hulbert (http://sourceforge.net/projects/matio/) (License: Simplified BSD). The matio io routines have been adopted to use R printing and error routines.

License GPL-3

URL https://github.com/stewid/rmatio

SystemRequirements zlib headers and library.

Type Package

LazyData true

Biarch true

Imports Matrix, methods, utils

Depends R(>= 3.0.2)

Collate 'have_lib.R' 'read_mat.R' 'rmatio.R' 'write_mat.R'

Encoding UTF-8

RoxygenNote 6.1.1

NeedsCompilation yes

Author Stefan Widgren [aut, cre] (Author of the R interface to the C-library matio),
Christopher Hulbert [aut] (Author of the C-library matio, http://sourceforge.net/projects/matio/)

Maintainer Stefan Widgren <stefan.widgren@gmail.com>

Repository CRAN

Date/Publication 2019-03-18 20:00:07 UTC

R topics documented:

read.mat 2
rmatio 3
write.mat 4

Index **7**

`read.mat` *Read Matlab file*

Description

Reads the values in a mat-file to a list.

Usage

```
read.mat(filename)
```

Arguments

`filename`	Character string, with the MAT file or URL to read.

Details

Reads the values in a mat-file and stores them in a list.

Value

A list with the variables read.

Note

- A sparse complex matrix is read as a dense complex matrix.
- A sparse logical matrix is read as a 'lgCMatrix'
- A sparse matrix is read as a 'dgCMatrix'
- A matrix of dimension `1 x n` or `n x 1` is read as a vector
- A structure is read as a named list with fields.
- A cell array is read as an unnamed list with cell data
- A function class type is read as NULL and gives a warning.

See Also

See `write.mat` for more details and examples.

Examples

```
## Read a version 4 MAT file with little-endian byte ordering
filename <- system.file('extdata/matio_test_cases_v4_le.mat', package='rmatio')
m <- read.mat(filename)

## View content
str(m)

## Read a version 4 MAT file with big-endian byte ordering.
filename <- system.file('extdata/matio_test_cases_v4_be.mat', package='rmatio')
m <- read.mat(filename)

## View content
str(m)

## Read a compressed version 5 MAT file
filename <- system.file('extdata/matio_test_cases_compressed_le.mat', package='rmatio')
m <- read.mat(filename)

## View content
str(m)
```

---

rmatio **rmatio**: *reading and writing Matlab MAT files from R*

Description

Reading and writing Matlab MAT files from R

Details

`rmatio` supports reading MAT version 4, MAT version 5 and MAT compressed version 5.

`rmatio` can write version 5 MAT files and version 5 files with variable compression.

References

- Christopher C. Hulbert, MATIO User Manual for version 1.5.2. https://sourceforge.net/projects/matio/files/matio/1.5.2/matio_user_guide.pdf
- The MathWorks Inc., MATLAB - MAT-File Format, version R2013b, September 2013. https://www.mathworks.com/help/pdf_doc/matlab/matfile_format.pdf

write.mat *Write Matlab file*

Description

Writes the values in a list to a mat-file.

Usage

```
write.mat(object, filename = NULL, compression = TRUE,
  version = c("MAT5"))

## S4 method for signature 'list'
write.mat(object, filename = NULL, compression = TRUE,
  version = c("MAT5"))
```

Arguments

object	The object to write.
filename	The MAT file to write.
compression	Use compression when writing variables. Defaults to TRUE.
version	MAT file version to create. Currently only support for Matlab level-5 file (MAT5) from rmatio package.

Details

Writes the values in the list to a mat-file. All values in the list must have unique names.

Value

invisible NULL

Note

- A vector is saved as a `1 x length` array
- Support for writing a sparse matrix of type 'dgCMatrix' or 'lgCMatrix' to file

Author(s)

Stefan Widgren

Examples

```
## Not run:
library("Matrix")
filename <- tempfile(fileext = ".mat")

## Example how to read and write an integer vector with rmatio
write.mat(list(a=1:5), filename=filename)
a <- as.integer(read.mat(filename)[["a"]])

stopifnot(identical(a, 1:5))

unlink(filename)

## Read a compressed version 5 MAT file
m <- read.mat(system.file("extdata/matio_test_cases_compressed_le.mat",
                          package='rmatio'))

## Write an uncompressed version 5 MAT file
write.mat(m, filename="test-uncompressed.mat", compression=FALSE, version="MAT5")

## Write a compressed version 5 MAT file
write.mat(m, filename="test-compressed.mat", compression=TRUE, version="MAT5")

## Check that the content of the files are identical
identical(read.mat("test-uncompressed.mat"), read.mat("test-compressed.mat"))

unlink("test-uncompressed.mat")
unlink("test-compressed.mat")

## Example how to read and write a S4 class with rmatio
## Create 'DemoS4Mat' class
setClass("DemoS4Mat",
         representation(a = "dgCMatrix",
                        b = "integer",
                        c = "matrix",
                        d = "numeric"))

## Create a function to coerce a 'DemoS4Mat' object to a list.
setAs(from="DemoS4Mat",
      to="list",
      def=function(from)
      {
        return(list(a=from@a,
                    b=from@b,
                    c=from@c,
                    d=from@d))
      }
)

## Create a function to coerce a list to a 'DemoS4Mat' object.
setAs(from="list",
      to="DemoS4Mat",
```

```
      def=function(from)
      {
        return(new("DemoS4Mat",
                   a=from[["a"]],
                   b=as.integer(from[["b"]]),
                   c=from[["c"]],
                   d=from[["d"]]))
      }
)

## Define a method to write a 'DemoS4Mat' object to a MAT file.
setMethod("write.mat",
          signature(object = "DemoS4Mat"),
          function(object,
                   filename,
                   compression,
                   version)
          {
            ## Coerce the 'DemoS4Mat' object to a list and
            ## call 'rmatio' 'write.mat' with the list.
            return(write.mat(as(object, "list"),
                             filename,
                             compression,
                             version))
          }
)

## Create a new 'DemoS4Mat' object
demoS4mat <- new("DemoS4Mat",
                 a = Matrix(c(0, 0, 0, 0, 0, 0, 1, 0, 0,
                              0, 0, 0, 0, 0, 0, 0, 1, 0,
                              0, 0, 0, 0, 0, 0, 0, 0, 1),
                              nrow=3,
                              ncol=9,
                              byrow=TRUE,
                              sparse=TRUE),
                 b = 1:5,
                 c = matrix(as.numeric(1:9), nrow=3),
                 d = c(6.0, 7.0, 8.0))

## Write to MAT file
write.mat(demoS4mat, filename)

## Read the MAT file
demoS4mat.2 <- as(read.mat(filename), "DemoS4Mat")

## Check result
stopifnot(identical(demoS4mat, demoS4mat.2))

unlink(filename)

## End(Not run)
```

Index

∗Topic **methods**
    write.mat, 4

read.mat, 2
rmatio, 3
rmatio-package *(*rmatio*)*, 3

write.mat, *2*, 4
write.mat,list-method *(*write.mat*)*, 4