Package 'jmatrix'

November 16, 2022

Type Package

Title Read from/Write to Disk Matrices with any Data Type in a Binary Format

Version 1.1

Date 2022-10-30

Author Juan Domingo [aut, cre] (<https://orcid.org/0000-0003-4728-6256>), Guillermo Ayala [ctb] (<https://orcid.org/0000-0002-6231-2865>), Spanish Ministry of Science and Innovation, MCIN/AEI <doi:10.13039/501100011033> [fnd]

Maintainer Juan Domingo <Juan.Domingo@uv.es>

Description A mainly instrumental package meant to allow other packages whose core is written in 'C++' to read, write and manipulate matrices in a binary format so that the memory used for them is no more than strictly needed. Its functionality is already inside 'parallelpam' and 'scellpam', so if you have installed any of these, you do not need to install 'jmatrix'. Using just the needed memory is not always true with 'R' matrices or vectors, since by default they are of double type. Trials like the 'float' package have been done, but to use them you have to coerce a matrix already loaded in 'R' memory to a float matrix, and then you can delete it. The problem comes when your computer has not memory enough to hold the matrix in the first place, so you are forced to load it by chunks. This is the problem this package tries to address (with partial success, but this is a difficult problem since 'R' is not a strictly typed language, which is anyway quite hard to get in an interpreted language). This package allows the creation and manipulation of full, sparse and symmetric matrices of any standard data type.

License GPL (>= 2)

Imports Rcpp (>= 1.0.8), memuse (>= 4.2.1)

LinkingTo Rcpp

RoxygenNote 7.2.1

Encoding UTF-8

Suggests knitr

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2022-11-16 14:30:02 UTC

R topics documented:

`GetJCol` *GetJCol*

Description

Returns (as a R numeric vector) the requested column number from the matrix contained in a jmatrix binary file

Usage

```
GetJCol(fname, ncol)
```

Arguments

`fname`	String with the file name that contains the binary data.
`ncol`	The number of the column to be returned, in R-numbering (from 1)

Value

A numeric vector with the values of elements in the requested column

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf[,3]
vf<-GetJCol("Rfullfloat.bin",3)
vf
file.remove("Rfullfloat.bin")
```

GetJColByName *GetJColByName*

Description

Returns (as a R numeric vector) the requested named column from the matrix contained in a jmatrix binary file

Usage

```
GetJColByName(fname, colname)
```

Arguments

fname	String with the file name that contains the binary data.
colname	The name of the column to be returned. If the matrix has no column names, or the name is not found, an empty vector is returned

Value

A numeric vector with the values of elements in the requested column

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf[,"c"]
vf<-GetJColByName("Rfullfloat.bin","c")
vf
file.remove("Rfullfloat.bin")
```

GetJColNames *GetJColNames*

Description

Returns a R StringVector with the column names of a matrix stored in the binary format of package jmatrix, if it has them stored.

Usage

```
GetJColNames(fname)
```

Arguments

fname	String with the file name that contains the binary data.

Value

A R StringVector with the column names, or the empty vector if the binaryfile has no row column names as metadata.

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
cn<-GetJColNames("Rfullfloat.bin")
cn
file.remove("Rfullfloat.bin")
```

GetJManyCols *GetJManyCols*

Description

Returns (as a R numeric matrix) the columns with the requested column numbers from the matrix contained in a jmatrix binary file

Usage

```
GetJManyCols(fname, extcols)
```

Arguments

fname	String with the file name that contains the binary data.
extcols	A numeric vector with the indexes of the columns to be extracted, in R-numbering (from 1)

Value

A numeric matrix with the values of elements in the requested columns

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf[,c(1,4)]
vc<-GetJManyCols("Rfullfloat.bin",c(1,4))
vc
file.remove("Rfullfloat.bin")
```

`GetJManyColsByNames` *GetJManyColsByNames*

Description

Returns (as a R numeric matrix) the columns with the requested column names from the matrix contained in a jmatrix binary file

Usage

```
GetJManyColsByNames(fname, extcolnames)
```

Arguments

`fname`	String with the file name that contains the binary data.
`extcolnames`	A numeric vector with the names of the columns to be extracted. If the binary file has no column names, or _any_ of the column names is not present, an empty matrix is returned.

Value

A numeric matrix with the values of elements in the requested columns

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf[,c(1,4)]
vf<-GetJManyColsByNames("Rfullfloat.bin",c("a","d"))
vf
file.remove("Rfullfloat.bin")
```

GetJManyRows *GetJManyRows*

Description

Returns (as a R numeric matrix) the rows with the requested row numbers from the matrix contained in a jmatrix binary file

Usage

```
GetJManyRows(fname, extrows)
```

Arguments

fname	String with the file name that contains the binary data.
extrows	A numeric vector with the indexes of the rows to be extracted, in R-numbering (from 1)

Value

A numeric matrix with the values of elements in the requested rows

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf[c(1,4),]
vc<-GetJManyRows("Rfullfloat.bin",c(1,4))
vc
file.remove("Rfullfloat.bin")
```

GetJManyRowsByNames *GetJManyRowsByNames*

Description

Returns (as a R numeric matrix) the rows with the requested row names from the matrix contained in a jmatrix binary file

Usage

```
GetJManyRowsByNames(fname, extrownames)
```

Arguments

fname	String with the file name that contains the binary data.
extrownames	A numeric vector with the names of the rows to be extracted. If the binary file has no row names, or _any_ of the row names is not present, an empty matrix is returned.

Value

A numeric matrix with the values of elements in the requested rows

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf[c("A","C"),]
vf<-GetJManyRowsByNames("Rfullfloat.bin",c("A","C"))
vf
file.remove("Rfullfloat.bin")
```

GetJNames *GetJNames*

Description

Returns a R list of two elements, rownames and colnames, each of them being a R StringVector with the corresponding names

Usage

```
GetJNames(fname)
```

Arguments

fname	String with the file name that contains the binary data.

Value

N["rownames","colnames"]: A list with two elements named rownames and colnames which are R StringVectors. If the binary file has no row or column names as metadata BOTH will be returned as empty vectors, even if one of them exists. If you want to extract only one, use either GetBinRowNames or GetBinColNames, as appropriate.

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
N<-GetJNames("Rfullfloat.bin")
N["rownames"]
N["colnames"]
file.remove("Rfullfloat.bin")
```

GetJRow *GetJRow*

Description

Returns (as a R numeric vector) the requested row number from the matrix contained in a jmatrix binary file

Usage

```
GetJRow(fname, nrow)
```

Arguments

fname	String with the file name that contains the binary data.
nrow	The number of the row to be returned, in R-numbering (from 1)

Value

A numeric vector with the values of elements in the requested row

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf[3,]
vf<-GetJRow("Rfullfloat.bin",3)
vf
file.remove("Rfullfloat.bin")
```

GetJRowByName *GetJRowByName*

Description

Returns (as a R numeric vector) the requested named row from the matrix contained in a jmatrix binary file

Usage

```
GetJRowByName(fname, rowname)
```

Arguments

fname	String with the file name that contains the binary data.
rowname	The name of the row to be returned. If the matrix has no row names, or the name is not found, an empty vector is returned

Value

A numeric vector with the values of elements in the requested row

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
Rf["C",]
vf<-GetJRowByName("Rfullfloat.bin","C")
vf
file.remove("Rfullfloat.bin")
```

GetJRowNames *GetJRowNames*

Description

Returns a R StringVector with the row names of a matrix stored in the binary format of package jmatrix, if it has them stored.

Usage

```
GetJRowNames(fname)
```

Arguments

fname	String with the file name that contains the binary data.

Value

A R StringVector with the row names, or the empty vector if the binary file has no row names as metadata.

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
rn<-GetJRowNames("Rfullfloat.bin")
rn
file.remove("Rfullfloat.bin")
```

GetSubdiag *GetSubdiag*

Description

Takes a symmetric matrix and returns a vector with all its elements under the main diagonal (without those at the diagonal itself) Done as an instrumental function to check the PAM in package cluster. To be removed in final version of the package.

Usage

```
GetSubdiag(fname)
```

Arguments

fname	The name of the file with the dissimilarity matrix in jmatrix binary format.

Value

The vector with the values under the main diagonal, sorted by columns (i.e.: m(2,1) .. m(n,1), m(3,2)..m(n,2),..., m(n-1,n))

Examples

```
Rns <- matrix(runif(49),nrow=7)
Rsym <- 0.5*(Rns+t(Rns))
rownames(Rsym) <- c("A","B","C","D","E","F","G")
colnames(Rsym) <- c("a","b","c","d","e","f","g")
JWriteBin(Rsym,"Rsymfloat.bin",dtype="float",dmtype="symmetric")
d<-GetSubdiag("Rsymfloat.bin")
Rsym
```

```
d
file.remove("Rsymfloat.bin")
```

JMatInfo *JMatInfo*

Description

Shows in the screen or writes to a file information about a matrix stored in the binary format of package jmatrix

Usage

```
JMatInfo(fname, fres = "")
```

Arguments

fname	String with the file name that contains the binary data.
fres	String with the name of the file to write the information. Default: "" (information is written to the console)

Value

No return value, called for its side effects (writes on screen or creates a file)

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
JMatInfo("Rfullfloat.bin")
file.remove("Rfullfloat.bin")
```

JMatrixSetDebug *JMatrixSetDebug*

Description

Sets debugging in jmatrix package to ON (with TRUE) or OFF (with FALSE).
On package load the default status is OFF.
Setting debugging to ON shows a message. Setting to OFF does not show anything (since debugging is OFF...)

Usage

```
JMatrixSetDebug(deb = TRUE)
```

Arguments

deb	boolean, TRUE to generate debug messages and FALSE to turn them off. Default: true

Value

No return value, called for side effects (internal boolean flag changed)

Examples

```
JMatrixSetDebug(TRUE)
JMatrixSetDebug(FALSE)
```

JWriteBin *JWriteBin*

Description

Writes a R matrix to a disk file as a binary matrix in the jmatrix format

Usage

```
JWriteBin(M, fname, dtype = "float", dmtype = "full", comment = "")
```

Arguments

M	The R matrix to be written
fname	The name of the file to write
dtype	The data type of the matrix to be written: one of the strings 'short', 'int', 'long', 'float' or 'double'. Default: 'float'
dmtype	The matrix type: one of the strings 'full', 'sparse' or 'symmetric'. Default: 'full'
comment	A optional string with the comment to be added as metadata. Default: "" (empty string, no added comment)

Details

Use this function cautiously. Differently to the functions to get one or more rows or columns from the binary file, which book only the memory strictly needed for the vector/matrix and do not load all the binary file in memory, this function books the full matrix in the requested data type and writes it later so with very big matrices you might run out of memory.
Type 'int' is really long int (8-bytes in most modern machines) so using 'int' or 'long' is equivalent.
Type is coerced from double (the internal type of R matrices) to the requested type, which may provoke a loose of precision.
If M is a named-R matrix, row and column names are written as metadata, too.
Also, if you write as symmetric a matrix which is not such, only the lower-diagonal part will be written. The rest of the data will be lost. In this case, if the matrix has row and column names, only row names are written.

Value

No return value, called for side effects (creates a file)

Examples

```
Rf <- matrix(runif(48),nrow=6)
rownames(Rf) <- c("A","B","C","D","E","F")
colnames(Rf) <- c("a","b","c","d","e","f","g","h")
JWriteBin(Rf,"Rfullfloat.bin",dtype="float",dmtype="full",comment="Full matrix of floats")
file.remove("Rfullfloat.bin")
```

Index