Package 'FITSio'

July 21, 2025

Type Package

Title FITS (Flexible Image Transport System) Utilities

Version 2.1-6

Date 2021-04-03

Author Andrew Harris

Maintainer Andrew Harris <harris@astro.umd.edu>

Depends R (>= 3.0.0)

Description Utilities to read and write files in the FITS (Flexible Image Transport System) format, a standard format in astronomy (see e.g. <https://en.wikipedia.org/wiki/FITS> for more information). Present low-level routines allow: reading, parsing, and modifying FITS headers; reading FITS images (multi-dimensional arrays); reading FITS binary and ASCII tables; and writing FITS images (multi-dimensional arrays). Higher-level functions allow: reading files composed of one or more headers and a single (perhaps multidimensional) image or single table; reading tables into data frames; generating vectors for image array axes; scaling and writing images as 16-bit integers. Known incompletenesses are reading random group extensions, as well as complex and array descriptor data types in binary tables.

License GPL (>= 2)

NeedsCompilation no

Repository CRAN

Date/Publication 2021-04-03 19:10:02 UTC

Contents

`FITSio-package` *FITS file input-output functions*

Description

FITS, the Flexible Image Transport System, is a standard data format for astronomy. This package contains functions to read files with FITS file image and binary and ASCII table formats and to write FITS image files. The functions comply with the standards defined by the International Astronomical Union's FITS Working Group.

Details

Package:	FITSio
Type:	Package
Version:	2.1-6
Date:	2021-03-28
License:	GPL (>= 2)
LazyLoad:	yes

At the high level, this package contains functions to read single FITS Header and Data Units (HDUs) containing image and binary table extensions, and one to write FITS image files. `readFITS` automatically recognizes image (multi-dimensional arrays) and binary table extensions, returning a list with data, header, and scaling information. `readFrameFromFITS` returns an R data frame from a single binary table HDU. Both functions accept an argument to pick out the *n*th HDU in a larger file. Binary table complex, and array descriptor data types are not implemented in this release due to a lack of examples for testing. 8, 16, 24, and 32-bit bit arrays return as integers. Other lengths are not supported.

For writing, the package contains the high-level function `writeFITSim` to write FITS images, along with a minor extension to efficiently write data in 16-bit integers, `writeFITSim16i`.

A set of mid-level functions enables reading files with combinations of HDUs: `readFITSheader` reads headers, `readFITSarray` reads image extensions, and `readFITSbintable` reads binary table extensions. `readFITStable` reads ASCII table extensions. `readFITS` and `readFrameFromFITS` invoke these functions to do the actual reading.

Function `axVec` generates a vector for image axis labeling from data in an image FITS file.

Author(s)

Andrew Harris <harris@astro.umd.edu>, with contributions from Eric H. Neilsen, Jr. and Bi Cheng Wu

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

Examples

```
require(FITSio)
## Make test image with axis information, write to disk
Z <- matrix(1:15, ncol = 3)
filename <- paste(tempdir(), "test.fits", sep="")
writeFITSim(Z, file = filename, c1 = "Test FITS file",
            crpix = c(1,1), crvaln = c(10, 100), cdeltn = c(8, 2),
            ctypen = c("Distance", "Time"),
            cunitn = c("Furlongs", "Fortnights"))

## Read image back and display
X <-  readFITS(filename)
ax1 <- axVec(1, X$axDat)          # Make axis vector for image
ax2 <- axVec(2, X$axDat)
xlab <- X$axDat$ctype[1]
ylab <- paste(X$axDat$ctype[2], " [", X$axDat$cunit[2], "]", sep = "")
image(ax1, ax2, X$imDat, xlab = xlab, ylab = ylab)
str(X)
X$axDat                           # Display data frame with axis data
X$hdr[1:10]                       # Header sample
X$hdr[which(X$hdr=="BITPIX")+1]   # BITPIX value from header

### Read back in, modify data, header, and axis data information,
## then write modified version as new file
Z <-  readFITS(filename)
  Z$imDat <- Z$imDat + 300
  Z$header <- addKwv('SCALE', 1.03, 'test header mod', header=Z$header)
  # Z$axDat <- edit(Z$axDat)  # interactive edit
  Z$axDat$cdelt[2] <- 20
filename2 <- paste(tempdir(), "test.fits", sep="")
writeFITSim(Z$imDat, file=filename2, axDat=Z$axDat, header=Z$header)

## Clean up files to avoid clutter
unlink(filename)
unlink(filename2)
```

axVec *Generate axis vector for image*

Description

axVec generates a vector of axis values from variables contained in a FITS image file header.

Usage

```
axVec(nax = 1, axDat)
```

Arguments

nax	index number of the axis.
axDat	data table axis variables produced by readFITSarray (which is also contained in readFITS).

Details

Run once for each axis needed.

Value

A vector with length equal to the number of pixels along the axis. Vector values $v[i]$ for $i = 1 : NAXISn$ ($NAXISn = len$, the vector length, in the axDat data frame) are:

$$v[i] = (i - CRPIXn)CDELTn + CRVALn$$

where $CRPIXn$, $CDELTn$, and $CVALn$ are the reference pixel, pixel increment, and reference pixel value for each axis n, following the FITS standard.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

readFITS, readFITSarray

Examples

```
require(FITSio)
Z <- matrix(1:15, ncol = 3)
filename <- paste(tempdir(), "test.fits", sep="")
writeFITSim(Z, file = filename, c1 = 'Test FITS file',
            crpix = c(1,1), crvaln = c(10, 100), cdeltn = c(8, 2),
            ctypen = c('Distance', 'Time'),
            cunitn = c('Furlongs', 'Fortnights'))
X <-  readFITS(filename)
ax1 <- axVec(1, X$axDat)
ax2 <- axVec(2, X$axDat)
xlab <- X$axDat$ctype[1]
ylab <- paste(X$axDat$ctype[2], " [", X$axDat$cunit[2], "]", sep = "")
image(ax1, ax2, X$imDat, xlab = xlab, ylab = ylab)

## Clean up files to avoid clutter
unlink(filename)
```

`closeHdr` *Format and close FITS header*

Description

Function adds END statement and closes FITS header

Usage

```
closeHdr(headerName)
```

Arguments

`headerName`	Header card images (vector)

Details

Adds `END` to header and pads with spaces to length defined in FITS standard.

Value

Character vector to write to FITS file as header.

Author(s)

A. Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

modifyHeader, makeFITSimHdr

Examples

```
X <- matrix(1:15, ncol = 3)
# extra header lines (optional)
header <- newKwv('KEYWORD', 'VALUE', 'NOTE') # initialize header
header <- addComment('Add these lines to auto-generated header', header=header)
header <- delKwv('KEYWORD', header=header)  # kill first line
header <- addKwv('test1', 'plot size', header=header)
header <- addKwv('test2', 4294.95397809807, 'number', header=header)
header <- addKwv('test3', 4.29495397809807e50, 'big number', header=header)
# make main header
header <- makeFITSimHdr(X, crpixn = c(1,1), crvaln = c(10, 100),
                        cdeltn = c(8, 2), ctypen = c("Distance", "Time"),
                        cunitn = c("Furlongs", "Fortnights"),
                        header = header)
# finish and close out header
tmp <- closeHdr(header)
```

---

makeFITSimHdr *Generate header for FITS image*

Description

Function generates the header for a FITS image. It determines much of the header information from the file itself and can merge header information with error checking and removal of conflicting reserved keywords.

Usage

```
makeFITSimHdr(X, primaryhdu = TRUE, type = "double", c1 = NA, c2 = NA,
bscale = 1, bzero = 0, crpixn = NA, crvaln = NA, cdeltn = NA, ctypen =
NA, cunitn = NA, axDat = NA, header = "")
```

Arguments

X	Multi-dimensional numeric data array; see Details.
primaryhdu	Logical: TRUE for stand-alone file, FALSE if subordinate header.
type	Type to write: single or double precision.
c1	Character string comment line for header.
c2	Character string comment line for header.
bscale	Global scaling factor, FITS standard meaning.
bzero	Global shift, FITS standard meaning.
crpixn	Vector of reference pixel numbers for axes, FITS standard meaning.

crvaln	Vector of values at reference pixels, FITS standard meaning.
cdeltn	Vector of axis increments per pixel, FITS standard meaning.
ctypen	String vector of descriptive labels for axis, FITS standard meaning.
cunitn	String vector of physical units for axis, FITS standard meaning.
axDat	Data frame with axis data, see details.
header	Optional header of 80-character card images.

Value

String with format and length matching FITS header standard for an image.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

writeFITSim

Examples

```
X <- matrix(1:15, ncol = 3)
# extra header lines (optional)
header <- newKwv('KEYWORD', 'VALUE', 'NOTE') # initialize header
header <- addComment('Add these lines to auto-generated header', header=header)
header <- delKwv('KEYWORD', header=header)  # kill first line
header <- addKwv('test1', 'plot size', header=header)
header <- addKwv('test2', 4294.95397809807, 'number', header=header)
header <- addKwv('test3', 4.29495397809807e50, 'big number', header=header)
# make main header
header <- makeFITSimHdr(X, crpixn = c(1,1), crvaln = c(10, 100),
                        cdeltn = c(8, 2), ctypen = c("Distance", "Time"),
                        cunitn = c("Furlongs", "Fortnights"),
                        header = header)
# finish and close out header
tmp <- closeHdr(header)
```

modifyHeader *Modify and edit comments and* keyword-value *pairs in FITS header*

Description

Functions to modify and edit keyword = value pairs in FITS header: newKwv creates keyword = value / comment line for header; addKwv adds keyword = value / comment to header; delKwv deletes keyword = value / comment from header; modVal modifies value in keyword = value / comment in header; addComment adds COMMENT line to header; addHistory adds HISTORY line to header.

Usage

```
newKwv(keyw, val, note)

addKwv(keyw, val, note, headerName)

delKwv(keyw, headerName)

modVal(keyw, val, note, headerName)

addComment(comment, headerName)

addHistory(history, headerName)
```

Arguments

keyw	Keyword, 8 character limit (FITS specification)
val	Value for keyword
note	Optional descriptive note following keyword = value
comment	Comment text string
history	History text string
headerName	Header card images (vector)

Details

All keywords will be converted to upper case (FITS convention) and truncated to 8 characters. There is no checking for duplicate keywords. The default numerical format for values is 20.14g; if other formats are needed, format separately and write the text string as the value (see examples). Similar work-arounds may be necessary for long note strings (see examples).

Value

New card image for newKwv, otherwise modified header card images.

Note

It is also possible to operate directly on the vector of header card images. These functions are simply for convenience in maintaining the correct header formatting. No END card is needed in the header card images; that is added by closeHdr, which must be run to complete the header formatting.

Author(s)

A. Harris

See Also

readFITS, readFITSheader, closeHdr

Examples

```
# Make header from scratch, then add, modify, delete card images.
header <- newKwv('KEYWORD', 'VALUE', 'NOTE')
header <- addKwv('test1', 'plot size', header=header)
header <- addKwv('test2', 4294.95397809807, 'number', header=header)
header <- addKwv('test3', 4.29495397809807e50, 'big number', header=header)
header <- addKwv('test4', -4.29495397809807e50, 'big number', header=header)
header <- addKwv('test5', 'this is a very long value', 'number', header=header)
header <- addKwv('test6',
   'this is a very very very very very very very very long value',
   'note', header=header)
header <- addKwv('test7', 'value',
   'very very very very very very very very very very long comment',
   header=header)
header <- addKwv('test8', '0123456789112345678921234567893123456',
   'format long number as string', header=header)
header <- addKwv('apostrophe', 'o\'\'malley',
   'apostrophes are doubled in value strings', header=header)
for (i in 1:4) {
   header <- addKwv(paste('test', 100+i, sep=''), 100+i, header=header)
}
header <- delKwv('test103', header=header)
header <- addComment('TEST103 should be missing from header', header=header)
header <- addHistory('This header generated by modifyHeader example',
   header=header)

# show header
header
```

---

parseHdr *Parse FITS header*

Description

Parse FITS header to extract `keyword = variable` pairs. Mostly useful for finding values given keyword name.

Usage

```
parseHdr(headerName)
```

Arguments

`headerName`	Header card images (vector)

Details

Uses strict FITS convention to identify `keyword = variable` pairs: a string '= ' in card image columns 9 and 10.

Value

Vector with elements `keyword, variable` for all pairs.

Note

Function eliminates leading and trailing spaces.

Author(s)

A. Harris

See Also

`readFITSheader`, `modifyHeader`

Examples

```
header <- newKwv('KEYWORD', 'VALUE', 'NOTE')
header <- addKwv('test1', 'plot size', header=header)
header <- addKwv('test2', 4294.95397809807, 'number', header=header)
header <- addKwv('test3', 4.29495397809807e50, 'big number', header=header)
header <- addKwv('test4', -4.29495397809807e50, 'big number', header=header)
parseHdr(header)
```

readFITS *Read a single data set from a FITS file*

Description

Read a single image (multi-dimensional array) or binary table from a FITS file. The source code `readFITS.r` is a model for creating code to read more complex FITS files.

Usage

```
readFITS(file = "R.fits", hdu = 1, maxLines = 5000,
fixHdr = c('none', 'remove', 'substitute'), phdu = 1)
```

Arguments

`file`	Existing FITS file name, or remote file; see examples.
`hdu`	Position of Header and Data Unit (HDU) in FITS file: 1 for the first HDU, 2 for the second HDU, etc.
`maxLines`	Integer: maximum number of header lines to read.
`fixHdr`	Deal with (illegal) nonprinting characters in the header.
`phdu`	Rarely needed; see Details.

Details

`readFITS` is a simple but complete FITS file reader, automatically detecting image and binary table data extensions (random groups are not supported in this release). It reads a single Header and Data Unit (HDU) pair from a file and returns a list with data, header, and axis information. Files with more complicated structures or isolated header units can be read with an appropriate combination of `readFITSheader`, `readFITSarray`, and `readFITSbintable`. See the Example section for `readFITSimage` for a step-by-step example that includes file opening and closing.

phdu = 0 forces a read of a secondary HDU in a fairly pathological case (specifically: indicating a read of a secondary HDU by setting NAXISn = 0 for n > 1, but NAXIS != 0 and NAXIS1 != 0.) This does not seem to be forbidden by the FITS standard but would be unlikely coding.

... passes additional values for file reading. The only use at present is to pass values to *fixHdr* in `readFITSheader`. *fixHdr* attempts to fix headers with non-printing characters, either by removing them with `fixHdr = 'remove'`, reading further into the file until 2880 valid characters are present, or by substituting spaces for non-printing characters with `fixHdr = 'substitute'`. This option should be used with caution, as non-printing characters should not be in the header in the first place, so this option may or may not corrupt the following data. The default is `fixHdr = 'none'`. Partial matching is allowed.

Binary table complex, and array descriptor data types are not implemented in this release due to a lack of examples for testing. 8, 16, 24, and 32-bit bit arrays return as integers. Other lengths are not supported.

Value

Return values from `readFITS` are in a list. Depending on the data type, list entries are:

`header`	Vector with input file header up to END statement.
`hdr`	Vector with `keyword = value` pairs parsed from header.
`imDat`	Data array (image).
`axDat`	Data frame with axis scaling and labels (image).
`F`	Data frame containing a table (table).
`col`	Data from each column (bintable).
`colNames`	Vector of column names, TTYPEn FITS variable (bintable, table).
`colUnits`	Vector of column units, TUNITn FITS variable (bintable, table).
`TNULLn`	Vector of undefined value definitions, FITS variable (bintable, table).
`TSCALn`	Vector of multipliers for scaling, FITS variable (bintable, table).
`TZEROn`	Vector of zeros for scaling, FITS variable (bintable, table).
`TDISPn`	Vector of format information, FITS variable (bintable).

Note

Graphical FITS viewers such as *fv* (https://heasarc.gsfc.nasa.gov/ftools/fv/) and *SAOImage DS9* (http://ds9.si.edu/) have excellent facilities for displaying FITS data, headers, and file structure. Having one or more graphical viewers available will prove extremely useful for working with FITS files, even when the data are read into R for further processing. *fv* and *SAOImage DS9* are in active development with support for unix, Windows, and Mac OS-X operating systems, and are available at no cost.

See `readFrameFromFITS` to read a binary table directly into an R data frame.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

`readFITSarray`, `readFITSbintable`, `readFITSheader`, `readFrameFromFITS`, `modifyHeader`, `image`, `par`

Examples

```
require(FITSio)

### Image example
## Make test image with axis information, write to disk
Z <- matrix(1:15, ncol = 3)
filename <- paste(tempdir(), "test.fits", sep="")
writeFITSim(Z, file = filename, c1 = "Test FITS file",
            crpix = c(1,1), crvaln = c(10, 100), cdeltn = c(8, 2),
            ctypen = c("Distance", "Time"),
            cunitn = c("Furlongs", "Fortnights"))
## Read back in and display
X <-  readFITS(filename)
ax1 <- axVec(1, X$axDat)          # Make axis vector for image
ax2 <- axVec(2, X$axDat)
xlab <- X$axDat$ctype[1]
ylab <- paste(X$axDat$ctype[2], " [", X$axDat$cunit[2], "]", sep = "")
image(ax1, ax2, X$imDat, xlab = xlab, ylab = ylab)
str(X)
X$axDat                           # Display data frame with axis data
X$hdr[1:10]                       # Header sample
X$hdr[which(X$hdr=="BITPIX")+1]   # BITPIX value from header
## No axis scale markings
image(X$imDat, xlab = xlab, ylab = ylab, xaxt = "n", yaxt = "n")
## Clean up to avoid clutter
unlink(filename)

### Binary table examples
## Bintable with one row and differently multi-dimensional columns
## Either download example file from
## <https://fits.gsfc.nasa.gov/fits_samples.html>
## and use
## Not run: filename <- "IUElwp25637mxlo.fits"
## or, for local example use
filename <- system.file("fitsExamples", "IUElwp25637mxlo.fits",
                        package = "FITSio")

Y <- readFITS(filename)
## Look at contents
str(Y)
Y$colNames
str(Y$col)
Y$hdr[which(Y$hdr=="BITPIX")+1]   # BITPIX value from header
plot(Y$col[[5]], ylab = "Value", main = Y$colNames[5], type = "l")

### Simple flat file example
filename <- system.file("fitsExamples", "2008_03_11_203150.fits",
                        package = "FITSio")
Z <- readFITS(filename)
str(Z)
Z$colNames
str(Z$col)
```

```
attach(Z)
xc <- which(colNames == "DMJD")
yc <- which(colNames == "TiltX")
xlab <- paste(colNames[xc], " [", colUnits[xc], "]", sep = "")
ylab <- paste(colNames[yc], " [", colUnits[yc], "]", sep = "")
plot(col[[xc]], col[[yc]], xlab = xlab, ylab = ylab, type = "l")
detach(Z)

### Read FITS file directly from URL (thanks to Bi Cheng Wu)
## Not run:
  require(httr)      # provides RETRY() and content()
  require(magrittr)  # provides %>% pipe operator, (part of tidyverse)
#
  fits <- RETRY(verb="GET", url=fits_url) %>%
    content(type="raw") %>%
    rawConnection %>%
    readFITS

## End(Not run)
```

`readFITSarray` *Read an image (multi-dimensional array) from a FITS file*

Description

Read an image (multi-dimensional array) from an open connection to a FITS file.

Usage

```
readFITSarray(zz, hdr)
```

Arguments

`zz`	File handle; see Example.
`hdr`	Header card images, raw or parsed.

Details

`readFITSarray` reads the data from the image part of a FITS Header and Data Unit (hdu) containing image data. The header must be read first by `readFITS` or `readFITSheader`; either this header or the parsed version from `parseHdr` are valid for the *hdr* variable.

Value

A list containing

`imDat`	Data array.
`axDat`	Data frame with axis scaling and labels.
`hdr`	Vector with the parsed header.

Note

Function assigns values of 1 to CRPIX, CRVAL, and CDELT if they are unspecified in the header.

Graphical FITS viewers such as *fv* (https://heasarc.gsfc.nasa.gov/ftools/fv/) and *SAOImage DS9* (http://ds9.si.edu/) have excellent facilities for displaying FITS data, headers, and file structure. Having one or more graphical viewers available will prove extremely useful for working with FITS files, even when the data are read into R for further processing. *fv* and *SAOImage DS9* are in active devlopement with support for unix, Windows, and Mac OS-X operating systems, and are available at no cost.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

readFITS, readFITSheader, readFITSbintable, file

Examples

```
require(FITSio)
## Make a test file.
Z <- matrix(1:15, ncol = 3)
filename <- paste(tempdir(), "test.fits", sep="")
writeFITSim(Z, filename)
## Open file, read header and array, close file and delete.
zz <- file(description = filename, open = "rb")
header <- readFITSheader(zz)  # image data off primary header
D <- readFITSarray(zz, header)
close(zz)
## Look at data list, header file, and parsed header
str(D)
image(D$imDat)
str(header)
str(parseHdr(header))
```

---

readFITSbintable *Read a FITS binary table*

Description

Read a FITS binary table from an open connection to a FITS file.

Usage

```
readFITSbintable(zz, hdr)
```

Arguments

zz	File handle; see Example.
hdr	Header card images, raw or parsed.

Details

readFITSbintable reads the data from the binary table part of a FITS Header and Data Unit (hdu) containing binary table data. The header must be read first by readFITSheader; either this header or the parsed version from parseHdr are valid for the *hdr* variable. Binary tables are multi-column files with one or more rows. Each column has an individual data type and number of entries per cell (i.e., a cell may contain a scalar or vector).

64-bit integers are read as pairs of 32-bit integers, for a vector twice the length of most other vectors. Files read properly, but reconstructing the 64-bit representation is untested. The CRAN package int64 may be of use here.

8, 16, 24, and 32-bit bit arrays return as integers. Other lengths are not supported.

Binary table complex and array descriptor data types are not implemented in this release due to a lack of examples for testing.

Value

col	Data from each column, either a vector or an array.
hdr	Vector with parsed header.
colNames	Vector of column names, TTYPEn FITS variable.
colUnits	Vector of column units, TUNITn FITS variable.
TNULLn	Vector of undefined value definitions, FITS variable.
TSCALn	Vector of multipliers for scaling, FITS variable.
TZEROn	Vector of zeros for scaling, FITS variable.
TDISPn	Vector of format information, FITS variable.

Note

Graphical FITS viewers such as *fv* (https://heasarc.gsfc.nasa.gov/ftools/fv/) and *SAOImage DS9* (http://ds9.si.edu/) have excellent facilities for displaying FITS data, headers, and file structure. Having one or more graphical viewers available will prove extremely useful for working with FITS files, even when the data are read into R for further processing. *fv* and *SAOImage DS9* are in active devlopement with support for unix, Windows, and Mac OS-X operating systems, and are available at no cost.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

readFITS, readFITSheader, readFITSarray, file

Examples

```
require(FITSio)

## Either download example file from
## <https://fits.gsfc.nasa.gov/fits_samples.html>
## and use
## Not run: filename <- "IUElwp25637mxlo.fits"
## or, for local example use
filename <- system.file("fitsExamples", "IUElwp25637mxlo.fits",
                        package = "FITSio")

## Open file, read header and table, close file.
zz <- file(description = filename, open = "rb")
header0 <- readFITSheader(zz) # read primary header
header <- readFITSheader(zz) # read extension header
D <- readFITSbintable(zz, header)
close(zz)

## Look at contents
str(D)
str(header)
str(parseHdr(header))
D$hdr[which(D$hdr=="BITPIX")+1]   # BITPIX value from header
D$colNames
plot(D$col[[5]], ylab = "Value", main = D$colNames[5], type = "l")
```

---

readFITSheader *Read a FITS header*

Description

Read a FITS header from an open connection to a FITS file.

Usage

```
readFITSheader(zz, maxLines = 5000, fixHdr = 'none')
```

Arguments

zz	file handle; see Example.
maxLines	maximum number of header lines to read; see Details.
fixHdr	deal with non-printing characters in header; see Details.

Details

readFITSheader reads the data from the header part of a FITS Header and Data Unit. In addition to general header information, it provides parameters needed to read image and binary table files by functions readFITSarray and readFITSbintable. A header unit may exist without a corresponding data unit, for instance to carry additional coordinate information.

The *maxLines* variable limits the number of header lines readFITSheader will read to prevent endless reading if the header is flawed and the END statement is missing. The function generates a message an halts when the number of reads exceeds *maxLines*. Increase the value as needed for very large headers.

fixHdr attempts to fix headers with non-printing characters, either by removing them with fixHdr = 'remove', reading further into the file until 2880 valid characters are present, or by substituting spaces for non-printing characters with fixHdr = 'substitute'. This option should be used with caution, as non-printing characters should not be in the header in the first place, so this option may or may not corrupt the following data. The default is fixHdr = 'none'. Partial matching is allowed.

The header vector does not have an easy format for people to read (graphical FITS viewers like *fv* and *SAOImage DS9* are good for this), but is designed for further processing by R. The header vector has a kind of *keyword value keyword value* ... format, where keywords without values are simply followed by the next keyword. This means a search for a keyword will give the corresponding value as the next element in the vector; see the Examples.

Value

hdr is a character vector *hdr* containing the header information in a format that is easy for R to parse further. See Details.

Note

Graphical FITS viewers such as *fv* (https://heasarc.gsfc.nasa.gov/ftools/fv/) and *SAOImage DS9* (http://ds9.si.edu/) have excellent facilities for displaying FITS data, headers, and file structure. Having one or more graphical viewers available will prove extremely useful for working with FITS files, even when the data are read into R for further processing. *fv* and *SAOImage DS9* are in active development with support for unix, Windows, and Mac OS-X operating systems, and are available at no cost.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

parseHdr, the usual complement to readFITSheader; readFITS, readFITSarray, readFITSbintable, file

Examples

```
require(FITSio)
## Make test image with axis information, write to disk
Z <- matrix(1:15, ncol = 3)
filename <- paste(tempdir(), "test.fits", sep="")
writeFITSim(Z, file = filename, c1 = "Test FITS file",
            crpix = c(1,1), crvaln = c(10, 100), cdeltn = c(8, 2),
            ctypen = c("Distance", "Time"),
            cunitn = c("Furlongs", "Fortnights"))
## Read back in
## Open file, read header and array.
    zz <- file(description = filename, open = "rb")
    header <- readFITSheader(zz)
    hdr <- parseHdr(header)
    D <- readFITSarray(zz, hdr)
    close(zz)
    hdr[1:10]                       # Header sample
    hdr[which(hdr=="BITPIX")+1]   # BITPIX value from header

## Clean up files to avoid clutter
unlink(filename)
```

---

readFITStable *Read a FITS ASCII table*

Description

Read a FITS ASCII table from an open connection to a FITS file.

Usage

```
readFITStable(zz, hdr)
```

Arguments

zz	File handle; see Example.
hdr	Header card images, raw or parsed.

Details

readFITStable reads the data from the ASCII table part of a FITS Header and Data Unit (hdu) containing ASCII table data. The header must be read first by readFITSheader; either this header or the parsed version from parseHdr are valid for the *hdr* variable. ASCII tables are multi-column files with one or more rows.

Value

hdr	Vector with parsed header.
DF	Data frame containing table data.
colNames	Vector of column names, TTYPEn FITS variable.
colUnits	Vector of column units, TUNITn FITS variable.
TNULLn	Vector of undefined value definitions, FITS variable.
TSCALn	Vector of multipliers for scaling, FITS variable.
TZEROn	Vector of zeros for scaling, FITS variable.

Note

Graphical FITS viewers such as *fv* (https://heasarc.gsfc.nasa.gov/ftools/fv/) and *SAOImage DS9* (http://ds9.si.edu/) have excellent facilities for displaying FITS data, headers, and file structure. Having one or more graphical viewers available will prove extremely useful for working with FITS files, even when the data are read into R for further processing. *fv* and *SAOImage DS9* are in active devlopement with support for unix, Windows, and Mac OS-X operating systems, and are available at no cost.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

readFITS, readFITSheader, readFITSarray, file

Examples

```
require(FITSio)

filename <- system.file("fitsExamples", "vizier.fits",
                        package = "FITSio")

## Simple read
D <- readFITS(filename)

## Explicit read: open file, read header and table, close file.
zz <- file(description = filename, open = "rb")
header0 <- readFITSheader(zz) # read primary header
header <- readFITSheader(zz) # read extension header
D <- readFITStable(zz, header)
close(zz)
```

```
## Either way, look at contents
str(D)
str(D$DF)
str(header)
str(parseHdr(header))
D$hdr[which(D$hdr=="BITPIX")+1]   # BITPIX value from header
D$colNames
```

`readFrameFromFITS` *Read a single data set from a FITS file into a data frame*

Description

Read a binary table from a FITS file directly into an R data frame.

Usage

```
readFrameFromFITS(file, hdu = 1)
```

Arguments

`file`	existing FITS file name.
hdu	position of Header and Data Unit (HDU) in FITS file: 1 for the first HDU, 2 for the second HDU, etc.

Details

`readFrameFromFITS` reads a single binary table Header and Data Unit (HDU) pair from a file and returns the values as a data table.

Binary table bit, complex, and array descriptor data types are not implemented in this release due to a lack of examples for testing. 8, 16, 24, and 32-bit bit arrays return as integers. Other lengths are not supported.

Value

An R data frame with the contents of the requested binary table.

Note

Graphical FITS viewers such as *fv* (https://heasarc.gsfc.nasa.gov/ftools/fv/) and *SAOImage DS9* (http://ds9.si.edu/) have excellent facilities for displaying FITS data, headers, and file structure. Having one or more graphical viewers available will prove extremely useful for working with FITS files, even when the data are read into R for further processing. *fv* and *SAOImage DS9* are in active devlopement with support for unix, Windows, and Mac OS-X operating systems, and are available at no cost.

Author(s)

Eric H.\ Neilsen, Jr.

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

readFITS

Examples

```
require(FITSio)

## Either download example file from
## <https://fits.gsfc.nasa.gov/fits_samples.html>
## and use
## Not run: filename <- "IUElwp25637mxlo.fits"
## or, for local example use
filename <- system.file("fitsExamples", "IUElwp25637mxlo.fits",
                        package = "FITSio")

## Get data and display
F <- readFrameFromFITS(filename)
names(F)
plot(F$NET, ylab = "Value", main = names(F)[5], type = "l")

### Simple flat file example
filename <- system.file("fitsExamples", "2008_03_11_203150.fits",
                        package = "FITSio")
F <- readFrameFromFITS(filename)
names(F)
attach(F)
plot(DMJD, TiltX, xlab = "Time [DMJD]", ylab = "X Tilt [degr]", type = "l")
detach(F)
```

---

writeFITSim *Write a FITS image (multi-dimensional numeric array) to disk*

Description

Write a FITS image (multi-dimensional numeric array) to disk.

Usage

```
writeFITSim(X, file = "R.fits", type = "double",
bscale = 1, bzero = 0, c1 = NA, c2 = NA,
crpixn = NA, crvaln = NA, cdeltn = NA, ctypen = NA, cunitn = NA,
axDat = NA, header = '')

writeFITSim16i(X, file = "R.fits", ...)
```

Arguments

X	Multi-dimensional numeric data array; see Details.
file	Output filename.
type	Type to write: single or double precision.
bscale	Global scaling factor, FITS standard meaning.
bzero	Global shift, FITS standard meaning.
c1	Character string comment line for header.
c2	Character string comment line for header.
crpixn	Vector of reference pixel numbers for axes, FITS standard meaning.
crvaln	Vector of values at reference pixels, FITS standard meaning.
cdeltn	Vector of axis increments per pixel, FITS standard meaning.
ctypen	String vector of descriptive labels for axis, FITS standard meaning.
cunitn	String vector of physical units for axis, FITS standard meaning.
axDat	Data frame with axis data (optional), see details.
header	Header (optional) as 80-character card images.
...	Arguments as defined above, as needed, for writeFITSim16i.

Details

writeFITSim and writeFITSim16i write multi-dimensional data arrays and header information to FITS-format files. A single image is a two-dimensional array; data cubes contain two dimensions plus one additional dimension for each (often velocity) plane.

Axis data may be given as a data frame axDat, typically from readFITS or, if that is missing, by individual vectors for crpixn, crvaln, etc. axDat has priority over individual vectors. axDat may be edited easily with e.g. axDat <- edit(axDat).

Header data may be added to that the function automatically generates. No or little editing is needed if the header is taken from an existing file by e.g. readFITSheader: writeFITSim will remove reserved keywords in the header that conflict with those generated and add the END statement. See modifyHeader for functions to add to or modify the header.

writeFITSim writes integer or float data matching the data type in the input array. writeFITSim16i scales and shifts the input to write 16-bit integer data with the maximum precision allowed by word length. FITS variables BSCALE and BZERO are automatically updated to allow reconstruction of the original data values. In cases where full precision is not needed, this can reduce file sizes by a factor of about four compared with a double-precision float.

Value

FITS file written to disk.

Note

Graphical FITS viewers such as *fv* (https://heasarc.gsfc.nasa.gov/ftools/fv/) and *SAOImage DS9* (http://ds9.si.edu/) have excellent facilities for displaying FITS data, headers, and file structure. Having one or more graphical viewers available will prove extremely useful for working with FITS files, even when the data are read into R for further processing. *fv* and *SAOImage DS9* are in active devlopement with support for unix, Windows, and Mac OS-X operating systems, and are available at no cost.

Author(s)

Andrew Harris

References

Hanisch et al., *Astron.\ Astrophys.* 376, 359-380 (2001)

https://fits.gsfc.nasa.gov/

See Also

readFITS, readFITSheader, modifyHeader

Examples

```
## Make data array with axis information, write to disk
X <- matrix(1:15, ncol = 3)
filename <- paste(tempdir(), "test.fits", sep="")
writeFITSim(X, file = filename, c1 = "Test FITS file",
            crpixn = c(1,1), crvaln = c(10, 100), cdeltn = c(8, 2),
            ctypen = c("Distance", "Time"),
            cunitn = c("Furlongs", "Fortnights"))
## Read back in, modify data offset, header, and axis data information,
## then write modified version as new file
Z <-  readFITS(filename)
Z$imDat <- Z$imDat + 300
Z$header <- addKwv('SCALE', 1.03, 'test header mod', header=Z$header)
# Z$axDat <- edit(Z$axDat)  # interactive edits
Z$axDat$cdelt
Z$axDat$cdelt[2] <- 20
Z$axDat$cdelt
writeFITSim(Z$imDat, file=filename, axDat=Z$axDat, header=Z$header)


### 3-dimensional array example
## Write sample file
X <- array(1:210, dim = c(10, 7, 3))
writeFITSim(X, filename)
## Read back in and display plane 2, no axis scale markings
```

```
Z <-  readFITS(filename)
dim(Z$imDat[,,2])
image(Z$imDat[,,2], xaxt = "n", yaxt = "n")
str(Z)
print(Z$axDat)

## Clean up files after examples to avoid clutter
unlink(filename)

### Note: either of the writeFITSim() calls here could be replaced
###       with writeFITSim16i() calls with the identical argument.
```

Index