IRAF Fits Kernel User's Guide

1 Introduction

The IRAF FITS Kernel available within IRAF v2.11 allows applications that handle images to access directly FITS format files. IRAF will read, modify or create images in standard FITS format as either a single FITS file or as an IMAGE extension within a file.

This user guide is divided into 2 major parts: Single FITS files; i.e. one image per file and MEF files operations. For single FITS files there is no need to learn all the complexities asociated with having more than one unit per file. On the other hand, MEF files provide an easy way to organized associated data files.

2 Single FITS files

IRAF can handle single FITS files in the same way as with 'imh' format. There is one logical disk file with an associated image; logical because the 'imh' image header has a pointer to where the pixel file is located. With a FITS file, we have one disk file; first, a header part ending in a logical boundary of 2880 bytes and following is the data part which end as well in a boundary of 2880 bytes. No more data follows this FITS file

For IRAF a FITS file has a default file extension 'fits' which you can omit by setting the cl variable 'imtype' to the value 'fits', like:

cl> set imtype=fits

You can choose any other arbitrary alphanumeric extension of 2 to 4 characters in length but you will need to let IRAF know that it is a FITS extension by setting the cl variable 'imextn' as:

```
cl> reset imextn = fxf:abf,fits
cl> reset imtype = abf
```

This syntax for 'imextn' tells IRAF that the 'abf' and 'fits' extension correspond to the 'fxf' image kernel (internal name for the FITS Kernel). The 'imtype' was reset to allow the user access to the files without a need to write out the extension.

2.1 FITS Scaling and Datatype

The FITS kernel will scale correctly at reading time an input FITS image based on the values of BSCALE and BZERO. When creating a new image or making a new copy and if the image datatype during computacion is REAL or DOUBLE, the values of BSCALE and BZERO will be one and zero respectively, while the value of BITPIX will be -32 or -64, indicating that the FITS image is in FITS IEEE standard format. When in input image has a BITPIX value of 8, the internal IRAF size will be 16 since this is the minimum size for computation if the values of BSCALE and BZERO are the default, otherwise the size will be 32. If BITPIX is 16, the same principle applies.

For unsigned 16 bits, the IRAF datatype is ushort and the FITS image datatype will be BITPIX 16 and BSCALE of one and BZERO of 32768.

2.1.1 Warning

If you are handling FITS files with BITPIX 16 or 8 and BSCALE and BZERO not the default values, the FK will create a temporary file to convert this image into floating point image. This temporary file is created in the working directory. If you do not have write access to the directory, an error will occur. To solve this problem in the meantime you will need to be in a writeable directory before any operation on those files is attempted.

2.2 Examples for single FITS files

1. Create a fits file from an image section of an 'imh' file.

   ```
   im> imcopy dev$pix.imh[100:250,200:300] pixf.fits
   ```

2. List the header.

   ```
   im> imheader pixf long+
   ```

3. Using imcombine.

   ```
   im> imcombine ctest*.fits comb9 reject="avsigclip"
   ```

4. Delete the images.

   ```
   im> imdelete pixf.fits
   ```

5. Read a FITS image from another machine on the IRAF networking.

   im> imcopy decaxp!/home/data/file.fits[100:200,2] sfile.fits

6. To create FITS file from other image type.

   im> reset imtype='fxf,noinherit'
   im> imcopy file.imh filef

   The 'filef' will have the default extension 'fits' since we have set the 'imtype' to the FITS kernel (fxf) default extension (fits).

   The 'reset imtype' line needs to be done only once in your or set it in your login.cl. If you reset it to handle another format a 'flprc' command is required.

   A Caveat to this example is that you cannot convert multiple files to fits files, i.e.

   imcopy *.imh .

   will NOT convert the current imh files to 'fits'. This is a shortcoming of the 'imcopy' command. To achieve the above you need to use list files.

   imcopy @in.lis @out.lis

3 Multiple Extensions FITS files (MEF)

A MEF file consists of one or more FITS units. A Unit is defined as either a Primary Header (PHU) plus the data portion or an Extension header (EHU) plus a data portion. The IRAF core system supports multiple extension FITS files but will handle only Primary Units or IMAGE extensions. For TABLE and BINTABLE extension please refer to the TABLES external package supported by ST.

3.1 IRAF filename specification

Version 2.11 has several new features to interpret from a simple to a complex FITS filename specification. The general syntax within IRAF for FITS filenames is:

root.extn[extension_number][kernel_section][im_section] *extn:* All but the 'root' name is required; 'extn' is the extension name that can be omitted only if it 'fits' or 'fit'. Other extensions are supported providing they have from 2 to 4 alphanumeric characters in length and are listed in the 'imextn' CL variable.

The 'imextn' specifies the set of valid extensions for each image type which has a default string:

imextn="oif:imh fxf:fits,fit plf:pl qpf:qp stf:hhh,??h"

oif, fxf, plf, qpf and stf are the names of the different image kernels supported by IRAF and their default extensions. To add new file extensions to the FITS kernel (fxf) just append them to the above. For example:

cl> reset imextn="fxf:fits,fit,z0f,caf,hwf,ag,bx"

All the rest of the kernel will have the default values.

extension_number: The 'extension_number' part of the filename is a decimal specifying which FITS units you are refering to. The numbering of FITS units is zero index. Zero for the Primary unit, 1 for the second unit or first extension, 2 for third unit and so on.

kernel_section: The 'kernel_section' part of the filename consists of a set of comma separated parameter with or without values. The general syntax is:

noparameter+/-=value

The parameter name can be omitted for certain names and they are case insensitive with minimum match. For example:

nic[3,noinherit]

refers to extension number 3 and any operation with the extension will not include the Global header (PHU). Notice that a decimal extension number can be set in the kernel_section only if it is the first item in the list.

image_section: Is the general image section part; e.g. [100:230,10:20].

3.1.1 Kernel Section parameters

EXTNAME [extname=]'string'. If the parameter name is not given, the string needs to start with an alphabetic character. The 'extname' parameter matches the 'EXTNAME' FITS extension keyword. 'extname and 'extver' are very useful to access an IMAGE extension by these values rather than the absolute extension number.

EXTVER [extver=]integer_value. If the parameter name is not given, the integer value needs to be at the second parameter position. The 'extver' parameter matches the 'EXTVER' FITS keyword . In general extname,extver are not duplicated among the units in a given FITS file.

APPEND [no]append[+,-,=yes,=no]. This parameter is required to append a FITS IMAGE extension to a FITS file. If the parameter is omitted or has the 'no' value and the 'imclobber' cl variable is set to yes, the existent file will be deleted before creating the PHU.

OVERWRITE [no]overwrite[+,-,=yes,=no]. Overwrite a FITS IMAGE unit. A temporary file is created in the working directory, copied all data before the unit, the new data then the data after the unit to be overwritten. Rename. No protection for clobbering.

INHERIT [no]inheritance[+,-,=yes,=no]. Adds the Primary header (providing it has NAXIS=0) to any extension header that has the INHERIT keyword with value T. The merging is done by excluding the FITS require keywords, HISTORY and COMMENT plus any keywords that is duplicated in the extension.

If an extension has INHERITANCE = T, and a new copy of this extension is created, the output inheritance value will be F. If you want inheritance on the output extension, the kernel parameter 'inherit' should be used.

DUPNAME [no]dupname[+,-,=yes,=no]. Allows extname,extver to be duplicated in the file.

EXPAND [no]expand[+,-,=yes,=no]. Allows the automatic expansion of headers when they run out of space by creating a temporary file in the working directory, adjusting the size and rewritting. If expand is disabled and the user area is larger than the header in disk, truncation will result and a warning message is issued. To avoid this problem of running out of space use 'ehulines' or 'padlines'.

EHULINES ehulines=integer. Preallocate a minimum number of header lines when creating a new image or making a new copy of one. If more lines are allocated than actual header lines, trailing space is added to the header in disk; if less, it has no effect.

PADLINES padlines=integer. Add a number of header lines to a new copy image header. When making a new copy of an image, the application can add more header lines to the olf header and there is a possibility that the output header size is larger than the old one. To avoid rewriting the file to make the header larger a padlines value to cover this problem can be given.

3.1.2 Default values for KS

When a FITS image specification is given to an IRAF task, the following are the default default values of the kernel section:

```
extname   = EOS
extver    = INDEFI
append    = NO
overwrite = NO
inherit   = YES
dupname   = NO
expand    = YES
ehulines  = 0
padlines  = 0
```

3.1.3 *fkinit*

The 'fkinit' Cl variable can have all of the above kernel section parameter values and can serve as the IRAF session default values, which overwrite the above default values and saves the user repetitiuos typing. For example:

cl> set fkinit='append,inherit'

Will make an application 'append' a FITS IMAGE extension into an existing file when new or new copy image is wanted. If reading an IMAGE extension, the application will merge the global (PHU) with the extension header (inherit). This 'inherit' value applies only to input filenames. To request 'inherit' on output you'll have to put the parameter in the output filename. See examples below.

3.2 FITS Header blank lines

When creating a new FITS unit, the FK will write the header END keyword at the end of the last header 2880 bytes block; hence extra blank cards can exists before the END card. The FK will use this space to add new keyword and it will not readjust if there is more than one block (2880) of blanks.

3.3 INHERIT on the Fits KerneL

Inherit means the ability for an IMAGE extension to have access to the Primary Header Unit for editing or making a copy of the combine global and extension header to another file. The Global header contains all those keywords that are common to all the extensions that have the keyword INHERIT with the value T. Hence there are no repeated keywords between the Global header and the extension that contains the keyword INHERIT with value T. If some application creates a new keyword on the extension that already exist on the global header the only way for it to be in the extension header is to have a different value. For inheritance to take effect, the PHU should be dataless; i.e. NAXIS = 0.

The agreement so far has been that inherit should be relevant only in the context of the local file and should not be carry to another files when copying individuals extension. Is up to the aplication or the user to be responsable in carrying the global header when making a copy of an extension.

There are a certain number of keywords that will not be copied from the global header to the combine global+extension header. These are the FITS required keywords plus the HISTORY and COMMENT keywords. A combined header will have then the required keywords and HISTORY and COMMENT from the extension header only.

As an usage note, the way to add a keyword to an extension that already exists in the global header but with a different value is to edit the extension header with the kernel section parameter 'noinherit'. This way the value of the INHERIT keyword in the extension header is preserve.

3.4 FITS Scaling and Datatype

The FITS kernel will scale correctly at reading time an input FITS image based on the values of BSCALE and BZERO. When creating a new image or making a new copy and if the image datatype during computacion is REAL or DOUBLE, the values of BSCALE and BZERO will be one and zero respectively, while the value of BITPIX will be -32 or -64, indicating that the FITS image is in FITS IEEE standard format. When in input image has a BITPIX value of 8, the internal IRAF size will be 16 since this is the minimum size for computation if the values of BSCALE and BZERO are the default, otherwise the size will be 32. If BITPIX is 16, the same principle applies.

For unsigned 16 bits, the IRAF datatype is ushort and the FITS image datatype will be BITPIX 16 and BSCALE of one and BZERO of 32768.

3.5 IRAF tasks and MEF

Multiple FITS extension files can be processed by any IRAF Tasks that already support images but with one image extension at a time. MEF files can be visualized as subdirectories with as many single FITS files as the number of extensions in each MEF file; hence to do 'imheader' on a subdirectory, the user would do:

```
imheader spec*.fits

but with a MEF file is necessary to list each extension:

imheader spec[1],spec[2],spec[3],spec[4]

or create a list file: 'spec.lis'

spec.fits[1]
spec.fits[2]
spec.fits[3]
spec.fits[4]

and
```

```
imheader @spec.lis
```

To create the above list files among other things related with IMAGE extensions, there is a new task in the PROTO package called 'imextensions'.

3.6 Examples for FITS extensions handling

1. Make a copy of a subsection of dev$pix in FITS format.

   im> imcopy dev$pix[100:255,*] pixf.fits

2. List the header.

   ```
       im> imheader pixf long+
   ```

3. Append an image to the above file. The image extension will have the EXTNAME keyword value 'sci' and EXTVER value 1. Notice that 'append' is necesary.

   ```
   im> imcopy nicm.fits pixf[extname\=sci,extver\=1,append]

   we can also type

   im> imcopy nicm.fits pixf[sci,1,append]

   This will overwrite the input EXTNAME and EXTVER values
   for 'sci' and 1 respectively.
   ```

4. If we have a FITS file with more than one extension, is it necessary to specify which one.

   ```
       im> splot spf.fits[ne,3]

       or if we know that this is extension 4 we can type

       im> splot spf[4]
   ```

5. Make a difference of 2 images and put the result in diff.fits.

   imarith image5_cal.fits[SCI,1] - image6_cal.fits[SCI,1] diff1

   Notice that the reference to the images is done by using the values of EXTNAME and EXTVER keywords.

6. Run imstatistics on all extension in a MEF file. Since the task can handle only one input image at a time, a small script is necessary:

```
        cl> for (i=1;i<4;i=i+1)
   imstat ("calf.fits["//i//"]")


        Or we can create a list file: "calf.lis"
        calf.fits[1]
        calf.fits[2]
        calf.fits[3]

        cl> imstat @calf.lis

We can use the new task 'imextension' in the proto
package to create the list file:

        cl> proto
        pr> imexten calf output=file > calf.list
        pr> imstat @calf.lis
```

7. There is an external package 'fitsutil' (see below for retrieval information) that allows us to see all the extension in a file.

```
    ta> fitsutil
    fi> fxheader pixf.fits
```

   Will list one line description per extension.

4 Non IMAGE Extension

IRAF task that handle images cannot deal with TABLE or BINTABLE extensions. A Warning message is issued when an task tried to process such extensions.

5 Shortcomings between IRAF and MEF

Most of the following problems can be solved by using tools available in other IRAF packages that deals exclusively with FITS files. For a new external package that has several fits utilities please install the external package from iraf.noao.edu/iraf/extern/fitsutil.tar.Z

1. It is important to use 'imdelete' and NOT 'delete' when deleting a MEF file, since the FK has a cache of the last 10 most recently files accessed by any IRAF task. To keep the cache in sync wiht the files on disk, imdelete will clear the cache entry indicating that the file is no longer on disk.

2. Cannot delete an IMAGE extension.

   imdelete spec.fits[3]

   is not implemented yet. A warning message will be issued. The task 'fxfdelete' from the fitsutil package will accomplish this.

3. Imdelete will delete the entire MEF file. With the present IRAF version:

   imdelete spec.fits

   will delete the whole file.

4. Cannot insert an image in the middle of a MEF file. The fitsutil task 'fxfinsert' will do this.

5. To get a listing of all the extensions in a MEF file use 'fxfheader'.

6. To copy more than one IMAGE extension from a MEF file use 'fxcopy'.

6 FITS Standard

The FITS standard that IRAF uses is based in the NOST document of September 29, 1995 which is accessible via ftp at nssdca.gsfc.nasa.gov in the subdirectory 'fits'. For more information about FITS, the web site is 'http://ssdoo.gsfc.nasa.gov/astro/fits/fits_home.html'.

Nelson Zarate
NOAO
zarate@noao.edu or iraf@noao.edu
August 1997