FITS Embedded Function Proposal
ASC-FITS-FUNCTION-1.0
SDS-11.1

Jonathan McDowell and Arnold Rots

1998 Mar 29

1 Introduction

This document describes a proposal the storage of a function or functional in a FITS binary table HDU. The HDU provides a recipe for returning a function or function value given the values of its parameters. The proposal will have the status of an ASC internal convention, although it is intended to be general enough that it may gain wider acceptance.

Changes to the initial proposal include:

- FTYPE keywords to include independent variables as well as tabulated function parameters; previously independent variables were left implicit.
- VFUNC/VTYPE keywords to provide in-line function macros for complicated function definitions.
- Parameters are enumerated in the function definition, replacing the PARAMi keywords.
- Key name changes: FNAXIS to FAXIS, FNAXISi replaced by FAXISi, MODEL replaced by FUNCTION

2 Definition

The HDU represents a function-valued function, called the table function. The table function is a mapping from an n-dimensional space, whose axes are called the table function parameters, to a space of functions, called the return function space. In other words, you give the HDU an n-tuple of numbers and it gives you back a function, which we call the return function. The return function space is not arbitrary; it is required to be the set of functions which are restrictions of a single, specified, function (the model function) of m+p parameters to an m-dimensional domain. We call the m parameters of the return function the 'independent variables' and the remaining p parameters of the model function the 'model function parameters'.

For example: Consider modelling an instrument point spread function P(x,y) as a circular gaussian G(x,y,sigma,norm) whose sigma and norm depend further on the external parameters energy and off-axis angle theta. Then we have:

- Model function $G : R^4 \rightarrow R = Gauss(x, y, sigma, norm)$ (i.e. G is a function from 4D space to 1D space).
- Independent variables: x,y
- Model function parameters: sigma, norm
- Return function $P : R^2 \rightarrow R = P(x, y)$, the restriction of G(x,y,sigma,norm) to the 2-D subdomain given by a particular choice of sigma and norm.
- Table function parameters: energy, theta
- Table function: $T : R^2 \rightarrow F(R^2 \rightarrow R)$, (i.e. T is a function from 2D space to the space of functions which map 2D space to 1D); T(energy,theta) = P(x,y) returns a function of x and y for any given energy and theta.

The model function may be a enumerated function with the single parameter of its index, in which case the table function is a simple map from an n-tuple to a function value. In this case, if it is further true that the table function parameters are all tabulated on regular constant-spaced intervals, the standard FITS IMAGE storage mechanism can and should be used.

An alternative view of the FUNCTION binary table is an n-dimensional image, not enumerated as in a FITS IMAGE extension, but specified analytically in a binary table. Hence, one can envision an interface implementation where the input to the interface is the specification of an n-dimensional grid, and the returned output an n-dimensional image on that grid, as specified by the FUNCTION binary table.

3 Specification

The HDU shall be recognized as containing a function by the (mandatory) presence of the keywords

```
HDUCLASS = 'ASC'
HDUCLAS1 = 'FUNCTION'
```

The value of HDUCLAS2 may indicate the function type.

A good, but not compulsory, choice of EXTNAME is 'FUNCTION'.

The mandatory keyword FUNCNAME gives the name of the table function. This name will be recognized by other software as the user name of this function.

The keyword FUNCTION gives the definition of the model function. Use of COMMENT keywords to explain the model function definition is encouraged. The syntax for FUNCTION specifications is a series of tokens combined with the operators +, -, *, **, /, (). The tokens must be either

- names, in which case they must be one of the TTYPEn, FTYPEn, DTYPEn, or VTYPEn.
- function calls, in which case they must be followed immediately by a parenthesis containing zero or more names as arguments, The function name must be one of a list which will be specified elsewhere (although we may later allow the name to be the name of another FITS-defined FUNCTION.)
- numeric constants.

Examples:

```
FUNCTION = 'Norm - Scale * ( X**2 + Y**2 )'  /
FUNCTION = '1.0 + exp( -X / k * T )'            /
FUNCTION = '1.0 + Cutoff'                    /
FTYPE3 = 'X '          / Independent variable
TTYPE2 = 'Y '          / Column of Y values
TTYPE3 = 'Scale'       / Column of scale values
DTYPE1 = 'Norm'        / Constant defined in NORM
DTYPE2 = 'k'           / Constant defined in K
VTYPE1 = 'Cutoff'      / Specifies an in-line function
VFUNC1 = 'exp( -X / k * T )' / Definition of VTYPE1
```

The VTYPEn/VFUNCn key pairs provide an abbreviation for a functional expression to be used in FUNCTION. VTYPE contains the abbreviation, VFUNC the expression which replaces it.

The standard FITS IMAGE keyword BUNIT shall be used to denote the unit of the function's return value, where appropriate.

The mandatory keyword FAXIS gives the number of table function parameters and independent variables.

The mandatory indexed keywords FTYPEi (i = 1 to FAXIS) give the name of the ith function variable. A function variable may be either an independent variable or a table function parameter. In the former case, it may be accompanied by an FUNITi keyword giving the unit and an

FLMINi/FLMAXi keyword giving the range accepted by the function. In the latter case (table function parameter), FTYPEi must be identical to one of the TTYPE values (We key off the name rather than the column number so as to be more robust to changes by file manipulation tools.) and FUNITi, if present, must be the same as TUNITi.

The optional indexed keywords FAXISi give the number of tabulated values for the ith function variable. This only makes sense if the same number of values for FTYPE1 are tabulated at each value of FTYPE2, etc., in other words the grid of points at which the function is tabulated is regular. The keywords FLMINi and FLMAXi may be used to indicate the range of values. For our initial implementation, we require the product of the provided FAXISi does in fact equal NAXIS2, so that the grid is filled.

The model parameters are given as table columns. The names of the parameters are tied to the function definition by their appearance in the model expressions (VFUNC). However, if a model parameter has a constant value for all the rows, it may be stored as a keyword instead (possibly as a DTYPE/DVAL/DUNIT combination).

For a model in which the function value is simply tabulated directly in a column, the column name (TTYPE) should be the same as the FUNCNAME, so e.g.

```
TTYPE4   = 'HRMA_AREA'
FUNCNAME = 'HRMA_AREA'
FUNCTION = 'HRMA_AREA'
```

Other columns may be present in the table and do not affect the interpretation of the function. For instance, if the table is used to store the results of a fitting process, one of the table function parameters might be 'iteration number' and an additional column might be the goodness of fit parameter.

4 Implementation

Simple linear interpolation for tabulated parameters is assumed in the current implementation. Future revisions to this proposal may add keywords to support other recommended interpolation types. Particular implementations of the FUNCTION interface may offer other interpolations, provided that the default behaviour is to use a specified interpolation function if and when indicated in the FUNCTION table, or else to use linear interpolation.

This document does not define a standard set of interface functions; such a list will be constructed in the near future.

In designing FUNCTION tables, attention should be given to composition of VFUNCi keywords, noting that some function expressions make it hard for interface parsers to optimize the function's evaluation on a grid of parameter values.

EXAMPLE 1: Simple function. We tabulate the HRMA effective area as a function of energy, off-axis angle, and assumed focal length: HRMA_AREA(ENERGY,THETA,F).

```
XTENSION = 'BINTABLE'
BITPIX   =         8
NAXIS    =         2
NAXIS1   =        16
NAXIS2   =        12
PCOUNT   =         0
GCOUNT   =         1
TFIELDS  =         4
TTYPE1   = 'ENERGY  '     / Energy
TUNIT1   = 'keV     '
TFORM1   = '1E      '
TTYPE2   = 'THETA   '      / Off-axis angle
TUNIT2   = 'arcmin  '
TFORM2   = '1E      '
TTYPE3   = 'F       '      / Focal length
TUNIT3   = 'mm       '
TFORM3   = '1E       '
TTYPE4   = 'HRMA_AREA'
TUNIT4   = 'cm**2   '
TFORM4   = '1E      '
EXTNAME  = 'AREA_TABLE'
HDUCLASS = 'ASC     '
HDUCLAS1 = 'FUNCTION'
HDUCLAS2 = 'TABLE'
COMMENT
COMMENT  Mission and instrument are relevant when units include counts
COMMENT  and pixels
COMMENT
MISSION  = 'AXAF    '
TELESCOP = 'AXAF    '
FUNCNAME = 'HRMA_AREA'   /
FAXIS    =           3  /
FTYPE1   =  'ENERGY'    /
FTYPE2   =  'THETA'     /
FTYPE3   =  'F    '     /
FUNIT1   =  'keV ' /
FUNIT2   =  'arcmin' /
FAXIS1   =           2  /
FAXIS2   =           3  /
FAXIS3   =           2  /
FUNCTION = 'None'       /
```

ENERGY	THETA	F	HRMA_AREA
1.5	0.0	10065.0	4.8
6.4	0.0	10065.0	2.3
1.5	1.0	10065.0	4.6
6.4	1.0	10065.0	2.1
1.5	2.0	10065.0	4.0
6.4	2.0	10065.0	1.8
1.5	0.0	10258.0	5.8
6.4	0.0	10258.0	3.3
1.5	1.0	10258.0	5.6
6.4	1.0	10258.0	3.1
1.5	2.0	10258.0	5.0
6.4	2.0	10258.0	3.8

EXAMPLE 2: Functional case. We give the ACIS PSF, which is modelled as a functional of summed circular gaussians. The table indicates the values of the gaussian parameters as a function of the independent variables. (In practice, a circular gaussian might be explicitly defined as the product of 2 1-D gaussians).

```
XTENSION = 'BINTABLE'
BITPIX   =         8
NAXIS    =         2
NAXIS1   =        52
NAXIS2   =         6
PCOUNT   =         0
GCOUNT   =         1
TFIELDS  =         9
TTYPE1   = 'ENERGY  '    / Energy
TUNIT1   = 'keV     '
TFORM1   = '1E      '
TTYPE2   = 'THETA   '     / Off-axis angle
TUNIT2   = 'arcmin  '
TFORM2   = '1E      '
TTYPE3   = 'SIGMA1'
TUNIT3   = 'pixel   '
TFORM3   = '1E      '
TTYPE4   = 'NORM1'
TUNIT4   = 'count    '
TFORM4   = '1E       '
TTYPE5   = 'HALO.X0  '        / Halo center X0
TUNIT5   = 'pixel    '
TFORM5   = '1E       '
TTYPE6   = 'HALO.Y0  '        / Halo center Y0
TUNIT6   = 'pixel    '
TFORM6   = '1E       '
TTYPE7   = 'SIGMA2'
TUNIT7   = 'pixel    '
TFORM7   = '1E       '
TTYPE8   = 'NORM2'
TUNIT8   = 'count    '
TFORM8   = '1E       '
TTYPE9   = 'TRW_ID   '  / Test used to derive this data point
TFORM9   = '20A      '  / (Example of an extraneous column)
EXTNAME  = 'FUNCTION'
HDUCLASS = 'ASC     '
HDUCLAS1 = 'FUNCTION'
HDUCLAS2 = 'COMPOUND'
COMMENT
COMMENT  Mission and instrument are relevant when units include counts
```

```
COMMENT  and pixels
COMMENT
MISSION  = 'AXAF    '   /
TELESCOP = 'AXAF    '   /
INSTRUME = 'ACIS    '   /
DTYPE1   = 'X0'    / Function parameter
DVAL1    =        0.0   / [pixel]
DTYPE2   = 'Y0'    / Function parameter
DVAL2   =        0.0   / [pixel]
DUNIT1  = 'pixel'
DUNIT2  = 'pixel'
FUNCNAME = 'ACIS_PSF'   /
FAXIS   =          4  /
FTYPE1   =  'ENERGY'    /
FTYPE2   =  'THETA'     /
FTYPE3   =  'X    '     /
FTYPE4   =  'Y    '     /
FUNIT1   =  'keV '     /
FUNIT2   =  'arcmin  '  /
FUNIT3   =  'pixel '   /
FUNIT4   =  'pixel '   /
FAXIS1  =          2  /
FAXIS2  =          3  /
FUNCTION = 'Core + Halo'
VFUNC1   = 'Core'
VTYPE1   = 'Circ2D(X,Y,Sigma1,Norm1,X0,Y0)'
VFUNC2   = 'Halo'
VTYPE2   = 'Circ2D(X,Y,Sigma2,Norm2,HALO.X0,HALO.Y0)'
ENERGY      THETA     SIGMA1  NORM1 HALO.X0 HALO.Y0 SIGMA2 NORM2  TRW_ID

1.5          0.0
6.4          0.0
1.5          1.0
6.4          1.0
1.5          2.0
6.4          2.0
```