Chapter 14

HDF Command-Line Utilities

14.1 Chapter Overview

A number of command-line utilities are available for working with HDF files. They are described in this chapter.

14.2 The HDF Command-Line Utilities

The HDF command-line utilities are application programs that are executed from the UNIX shell prompt. These utilities serve the following needs of the HDF developer.

- They make it possible to perform, at the command line level, common operations on HDF files without having to resort to custom-programmed utilities to do these operations..
- They provide the capability for performing operations on HDF files that would be very difficult to do with custom-programmed utilities..

Table 14A lists the names and descriptions of the utilities described in this section. Descriptions and examples of these routines follow.

TABLE 14A **The HDF Command-Line Utilities**

Name	Description
hdfls	Displays the tags, reference numbers, and lengths of data elements.
hdfed	Displays the contents of an HDF file and allows limited manipulation of the data.
fp2hdf	Converts floating-point data to HDF floating-point format and/or HDF 8-bit raster image format (RIS8).
ristosds	Converts a set of RIS8 HDF files into a single three-dimensional SDS HDF file.
r8tohdf	Converts one or more 8-bit raster images in raw format to the HDF RIS8 format and writes them to a file, optionally with palettes.
hdftor8	Converts raster images and/or palettes from the HDF format to the raw format and stores them in two sets of files - one for images and the other for palettes.
hdfcomp	Compresses 8-bit raster images from an HDF file, storing them in a new HDF file.
r24hdf8	Converts raw RGB 24-bit images to an RIS8 with a palette.
paltohdf	Converts a raw palette to the HDF format.
hdftopal	Converts a palette in an HDF file to a raw palette format.
hdfpack	Compresses an HDF file, reading all of the objects in the file and writing them to a new HDF file.

Name	Description
vmake	Creates vsets.
vshow	Displays vset information.
hdp	Displays general information about the contents of an HDF file.

14.2.1 Listing Basic Information About an HDF File: hdfls

14.2.1.1 General Description

The `hdfls` utility provides general information about the tags, reference numbers, and if requested, lengths of the data elements.

14.2.1.2 Command-Line Syntax

```
hdfls [-o][-l][-d][-v][-g][-s][-h][-t #] filename
```

`-o`	Ordered	Indicates the reference numbers are to be displayed in ascending order.
`-l`	Long format	Displays more information about the file.
`-d`	Offset/ length	Offset and length information will be displayed for each element in the file.
`-v`	Verbose	Annotation and label text will be displayed, along with the format triggered by the -l flag. (long format)
`-g`	Group	List items by group.
`-s`	Special elements	Display detailed informaton about special elements.
`-h`	DD block	Dump DD block information.
`-t`	Tag	Only list information about the specified tag. Must be followed by a tag number.

14.2.1.3 Examples

A file called "aa.hdf" contains three items associated with a raster image; the image dimensions, a palette, and the raster image. To display information about the contents of this file, the following command is used.

```
hdfls aa.hdf
```

The following output is displayed:

```
aa.hdf:
Image Dimensions-8:      (Raster-8): (tag 200)
       Ref nos: 1
Image Palette-8:         (Raster-8): (tag 201)
       Ref nos: 3
Raster Image-8:          (Raster-8): (tag 202)
       Ref nos: 1
```

The following displays the same information with the length of each data element:

```
hdfls -l aa.hdf
```

This would result in the following information being displayed:

```
aa.hdf:
Image Dimensions-8:  (Raster-8) : (tag 200)
       Ref no 1                  4 bytes

Image Palette-8:     (Raster-8) : (tag 201)
       Ref no 3                  768 bytes

Raster Image-8:      (Raster-8) : (tag 202)
       Ref no 1                  120000 bytes
```

14.2.2 Editing the Contents of an HDF File: hdfed

14.2.2.1 General Description

The hdfed utility allows experienced HDF users to manipulate the elements of an HDF file. These manipulations include

- Selecting groups and showing information about them.
- Dumping group information to output files.
- Writing group data to output files.
- Deleting groups from HDF files.
- Inserting groups in HDF files.
- Replacing elements of HDF files.
- Editing the labels and descriptions of any element in an HDF file.

It is designed primarily for users who need to know about HDF files at the level of individual data elements. It is not designed to provide a comprehensive high-level view of the contents of an HDF file - other tools and utilities should be used for that purpose. To use hdfed one should be familiar with the components of an HDF file covered in the *HDF Specifications* manual.

The hdfed utility is loosely modeled on ed, the UNIX line editor. When hdfed is invoked, it prompts the user for commands, as does ed. Also, basic command syntax and description information is available to the user through hdfed. The most common hdfed commands are used to control the position in the HDF file and the format of the information provided.

The initial view of the file under hdfed consists of a set of tag/reference number pairs. Although hdfed allows modification of tags and reference numbers *within strict constraints,* it will not allow the user to arbitrarily modify binary data in the file.

The following terms and concepts must be understood in order to use hdfed correctly and will be used in the following discussion about hdfed.

- The ***data object*** or ***object*** refers to an HDF data object and the data descriptor of that object. (i.e., tags, reference numbers, offsets, or lengths.)
- The ***data*** or ***data element*** refers to the record that the data descriptor points to. For a precise definition of the data that is associated with a given tag consult the *HDF Specifications* manual.
- The ***group*** refers to a predefined collection of data objects that correspond to a particular application. For example, a raster image group refers to the collection of objects that are used to store all of the information in a raster image set.

Once an HDF file has been opened by hdfed, the following operations can be performed on the data file, among others:

- Select an HDF object to examine more closely.
- Move forward or backward within the HDF file.
- Get information about an object. (tag, reference number, size, label)
- Display a raster image using the ICR protocol.
- Display the contents of any object.
- Delete an object.
- Annotate an object with a label or description.
- Write an object to a second HDF file.
- Write data elements in binary form to a non-HDF file.
- Close the file and exit, or open a new file.

14.2.2.2 Command-Line Syntax

The syntax of `hdfed` is

```
hdfed [-nobackup][-batch] filename
```

If a file named `filename` exists, it is opened and a backup is made of the file. Files may also be opened from within the editor.

.

`-nobackup`	Specifies that no backup file is to be made. If this option is omitted, a backup file is automatically created.
`-batch`	Specifies that input to `hdfed` is to be input via a stream of `hdfed` commands, rather than interactively.

The `-batch` flag is useful when a group of commonly-used commands are included in a UNIX shell script. The following is an example of such a script, using the C-shell, that lists information about the groups in a specified HDF file.

```
#!/bin/csh -f
set file=$1
shift
hdfed -batch $file -nobackup << EOF
info -all group $*
close
quit
EOF
echo ""
```

To receive usage information, as well as a quick list of the `hdfed` commands, type the command

```
hdfed -help
```

While in `hdfed`, the standard command prompt is displayed.

```
hdfed>
```

Many `hdfed` commands have qualifiers, or flags. For example, the `info` command may be followed by the `-all`, `-long`, `-group`, or `-label` flags.

All of the commands and flags can be abbreviated to the extent that their abbreviations are unique. For example, `-he` is ambiguous as it could stand for either the `-hexadecimal` or the `-help` flags, but `-hel` is not ambiguous.

TABLE 14B

The **`hdfed`** Command Set

Name	Description
help	Displays general `hdfed` help information.
open	Opens an HDF file.
close	Closes an HDF file.
revert	Reverts to the original HDF file.
next	Goes to the next object or group that satisfies the predicate.
prev	Goes to the previous object or group that satisfies the predicate.
info	Displays information about the current data object.
dump	Displays information about the current data object in non-default formats. (i.e., binary, ASCII, etc.). The default is octal.
display	Displays a raster image using ICR.
put	Writes the current data element in a non-HDF file with the specified filename in binary format.
putr8	Writes the current RIS8 group into a non-HDF file with the specified file-name.
getr8	Reads a RIS8 group from a non-HDF file with the specified filename.
delete	Deletes an object or group.
write	Writes an object or group to an HDF file.
annotate	Annotates an object.
if	Conditional statement.
select	Loop for each object.
alias	Defines an alias or display the alias list.
unalias	Deletes an alias.
wait	Prints a message and wait for a carriage return.

To obtain information about the usage of any `hdfed` command, type the following at the `hdfed` prompt.

```
any hdfed command -help
```

Note that usage information cannot be obtained by typing only the command, with no flags. There are `hdfed` commands like `delete` that do not require an argument, so watch out for this kind of error.

There is a subset of `hdfed` commands where *predicates*, *items*, and *comparators* are used. ***Items*** are used to denote an HDF object type and can be any of the following identifiers; `tag`, `ref`, `image_size`, or `label`. A ***comparator*** is an expression used to compare an item with a user-defined value, and can be any of the following; "=" (equal to), "!=" (not equal to), ">" (less than), "<" (greater than), ">=" (greater than or equal to), "<=" (less than or equal to). User-defined values can be either a number (with or without a decimal point) or a string of characters delimited by double-quotes. ***Predicates*** consist of items, comparators and user-defined values and are of the syntax

```
item comparator value
```

Or they may consist of the identifier `group, as` in the `next group` command. Some examples of predicates are:

```
next group
next (same as "next group" as "group" is the default identifier)
next tag = 720
next ref = 2
next image_size < 1000
next label = "abc"
```

The following is a more inclusive description of the `hdfed` commands.

The `help` command

Syntax:	**help**
Flags:	None
Description:	Prints a help screen describing the basic purpose and functionality of the hdfed utility.
Usage Example:	

```
hdfed> help

hdfed allows sophisticated HDF users the ability to manipulate the
elements in an HDF file. These manipulations include selecting groups
...
```

The `open` command

Syntax:	**open [-nobackup] filename**	
Flags:	**-nobackup**	The specified file name is not backed up.
Description:	Opens the specified HDF file.	
Usage Example:		

```
hdfed> open -help
open <file> [-nobackup]
-nobackup Don't make a backup for this file.
hdfed>
hdfed> open h1
hdfed>
```

The `info` command

Syntax:	**info [-all] [-long] [-group] [-label]**	
Flags:	**-all**	Displays information for all of the objects in the current file.
	-long	Displays the long form of the information.
	-group	Organizes the information into groups.
	-label	Shows any labels.
Description:	Displays information for a data object. The listing for special elements will contain a special tag value (in Item 13 below it's 18347, which corresponds to DFTAG_VS) and the text "Unknown Tag".	
Usage Example:		

```
hdfed> info -all -label -long
   (1)     Version Descriptor             : (Tag 30)
           Ref: 1, Offset: 202, Length    :92 (bytes)
   (2)     Scientific Data                : (Tag 702)
           Ref: 2, Offset: 294, Length    : 200 (bytes)
   (3)     Number type                    : (Tag 106)
           Ref: 2, Offset: 494, Length    : 4 (bytes)
   (4)     SciData description            : (Tag 701)
           Ref: 2, Offset: 498, Length    : 2 (bytes)
   (5)     SciData max/min                : (Tag 707)
           Ref: 2, Offset: 520, Length    : 4 (bytes)
```

```
        *(6)     Numeric Data Group               : (Tag 720)
                 Ref: 2, Offset: 524, Length      : 12 (bytes)
                 Label: Experiment #1
         (7)     Data Id Label                    : (Tag 104)
                 Ref: 3, Offset: 536, Length      : 17 (bytes)
         (8)     Scientific Data                  : (Tag 702)
                 Ref: 4, Offset: 553, Length      : 400 (bytes)
         (9)     Number type                      : (Tag 106)
                 Ref: 4, Offset: 953, Length      : 4 (bytes)
         (10)    SciData description              : (Tag 701)
                 Ref: 4, Offset:957, Length       : 22 (bytes)
         (11)    Numeric Data Group               : (Tag 720)
                 Ref: 4, Offset: 979, Length      : 8 (bytes)
                 Label: Experiment #2
         (12)    Data Id Label                    : (Tag 104)
                 Ref: 5, Offset: 987, Length      : 17 (bytes)
         (13)    Unknown Tag                      : (Tag 18347)
                 Ref: 8, Offset: 0, Length        : 40(bytes
hdfed>
hdfed> info -group -all
**Group 1:
                 Numeric Data Group           : (Tag 720) Ref 2
                 Scientific Data              : (Tag 702) Ref 2
                 SciData description          : (Tag 701) Ref 2
                 SciData max/min              : (Tag 707) Ref 2
**Group 2:
                 Numeric Data Group           : (Tag 720) Ref 4
                 Scientific Data              : (Tag 702) Ref 4
                 SciData description          : (Tag 701) Ref 4
**These do not belong to any group:
                 Version Descriptor           : (Tag 30) Ref 1
                 Number Type                  : (Tag 106) Ref 2
                 Data Id Label                : (Tag 104) Ref 3
                 Number Type                  : (Tag 106) Ref 4
                 Data Id Label                : (Tag 104) Ref 5
hdfed>
```

The `prev` command

Syntax: **prev predicate list**

Flags: None.

Description: Moves to the next object that satisfies the predicate list.

Usage Example:

```
hdfed> info -all
      (1)       Version Descriptor   : (Tag 30) Ref 1
      (2)       Scientific Data      : (Tag 702) Ref 2
      (3)       Number type          : (Tag 106) Ref 2
      (4)       SciData description  : (Tag 701) Ref 2
      (5)       SciData max/min      : (Tag 707) Ref 2
     *(6)       Numeric Data Group   : (Tag 720) Ref 2
      (7)       Data Id Label        : (Tag 104) Ref 3
      (8)       Scientific Data      : (Tag 702) Ref 4
      (9)       Number type          : (Tag 106) Ref 4
      (10)      SciData description  : (Tag 701) Ref 4
      (11)      Numeric Data Group   : (Tag 720) Ref 4
      (12)      Data Id Label        : (Tag 104) Ref 5
hdfed>
hdfed> ! The '*' in the first column marks the current
hdfed> ! position.
hdfed> ! The 'next' and 'prev' commands work with predicates.
hdfed> ! If I want to move to the max/min element,
hdfed> ! I can use the 'tag=' predicate.
hdfed>
hdfed> prev tag=707
hdfed> info
      (5)       SciData max/min (SciData) : (Tag 707) Ref:2
hdfed>
```

The `next` command

Syntax: **next <u>predicate list</u>**
Flags: None.
Description: Moves to the next object that satisfies the predicate.
Usage Example:

```
hdfed> ! Move in the file using next and prev
hdfed> ! The move direction depends on the relative positions.
hdfed> ! so it is often necessary to do an 'info -all' first.
hdfed> info -all
        (1)      Version Descriptor           : (Tag 30) Ref 1
        (2)      Scientific Data              : (Tag 702) Ref 2
        (3)      Number type                  : (Tag 106) Ref 2
        (4)      SciData description          : (Tag 701) Ref 2
       *(5)      SciData max/min              : (Tag 707) Ref 2
        (6)      Numeric Data Group           : (Tag 720) Ref 2
        (7)      Data Id Label                : (Tag 104) Ref 3
        (8)      Scientific Data              : (Tag 702) Ref 4
        (9)      Number type                  : (Tag 106) Ref 4
        (10)     SciData description          : (Tag 701) Ref 4
        (11)     Numeric Data Group           : (Tag 720) Ref 4
        (12)     Data Id Label                : (Tag 104) Ref 5
hdfed>
hdfed> ! This predicate persists for the next and prev
hdfed> ! commands. That means if I now type another 'next'
hdfed> ! command, it will look for a tag that equals 707.
hdfed>
hdfed> next
Reached end of file. Not moved.
hdfed> info
        (5)       SciData max.min (SciData)    : (Tag 707) Ref: 2
hdfed>
hdfed> next group
hdfed> next group
hdfed> info
        (11)     Numeric Data Group           : (Tag 720) Ref 4
hdfed>
```

The `dump` command

Syntax: **dump [-offset <u>offset</u>] [-length <u>length</u>] [-decimal|-short|-byte|-octal|-hexadecimal|-float|-double|-ascii]**

Flags:

-offset	Starting offset
-length	Length of the object to dump.
-decimal	Decimal format (32-bit integers)
-short	Decimal format (16-bit integers)
-byte	Decimal format (8-bit integers)
-octal	Octal format (the default)
-hexadecimal	Hexadecimal format
-float	Single-precision floating-point format (32-bit floats)
-double	Double-precision floating-point format (16-bit floats)
-ascii	ASCII format

Description: Displays the contents of the current object in the specified format.

Usage Example:

```
hdfed> ! to see the binary representation of this element
hdfed>
hdfed> dump
0: 257400004 257200004
hdfed>
hdfed> dump -short
hdfed>
0:      702      4       701      4
```

```
hdfed>
```

The `delete` command

Syntax: **delete**
Flags: None.
Description: Deletes the current object or group.
Usage Example:

```
hdfed> ! deleting groups
hdfed>
hdfed> ! If an element is required by other group it is alone.
hdfed> ! However, this is not perfect as the method by which group
hdfed> ! membership is determined can be pretty ad hoc.
hdfed>
hdfed> delete
hdfed> ! This deletes the Scientific Data Group
hdfed> info -all
      (1)      Version Descriptor       : (Tag 30) Ref 1
      (2)      Scientific Data          : (Tag 702) Ref 2
      (3)      Number type              : (Tag 106) Ref 2
      (4)      SciData description      : (Tag 701) Ref 2
      (5)      SciData max/min          : (Tag 707) Ref 2
      (6)      Numeric Data Group       : (Tag 720) Ref 2
      (7)      Data Id Label            : (Tag 104) Ref 3
      (8)      Number type              : (Tag 106) Ref 4
      (9)      Data Id Label            : (Tag 104) Ref 5
hdfed>
hdfed> ! Notice that the Numeric Data Group with reference
hdfed> ! number 4 is missing, and now there are only 9
hdfed> ! objects in the file.
hdfed>
```

The `annotate` command

Syntax: **annotate [-label] [-descriptor] [-editor <u>editor</u>]**

Flags:
- **-label** Edit a label (the default)
- **-descriptor** Edit a descriptor.
- **-editor** Use an editor. (Default is the editor referred to by the EDITOR environment variable.

Description: Edits an annotation.
Usage Example:

```
hdfed>
hdfed> ! Annotations are labels and descriptors
hdfed>
hdfed> prev -group
hdfed> info -label
      (6)      Numeric Data Group    : (Tag 720) Ref 2
               Label: Experiment #1
hdfed> annotate -editor /usr/ucb/ex
"/tmp/he5091.1" 1 line, 14 characters
:p
Experiment #1
:s/$/      <more stuff>/
Experiment #1<more stuff>
:wq
"/tmp/he5091.1" 1 line 27 characters
hdfed> info -label
      (6)      Numeric Data Group    : (Tag 720) Ref 2
               Label: Experiment #1 <more stuff>
hdfed>
```

The `write` command

Syntax: **write [-attachto <u>tag</u> <u>reference number</u>] <u>filename</u>**

Flags:
- **-attachto** Which element the annotation will be attached to. (only for writing annotations)

Description: Writes an element or group into another HDF file.
Usage Example:

```
hdfed>
hdfed> ! Write object or group to another HDF file.
hdfed>
hdfed> write test
hdfed>
hdfed> ! Let's take a look at the file 'test'
hdfed> close; open test; info -all
      (1)     Version Descriptor    (Tag 30) Ref 1
      (2)     Scientific Data       (Tag 702) Ref 2
      (3)     Number type           (Tag 106) Ref 2
      (4)     SciData description   (Tag 701) Ref 2
      (5)     SciData max/min       (Tag 707) Ref 2
     *(6)     Numeric Data Group    (Tag 720) Ref 2
hdfed>
hdfed> close;
hdfed>
```

The `display` command

Syntax: **display [-position x-position y-position] [-expansion expansion] [-large]**

Flags:

-position	Image position on console screen
-expansion	Image expansion factor
-large	Make image as large as possible.

Description: Displays image on screen.
Usage Example:

```
hdfed> ! We will open a file with some RIS8 images.
hdfed>
hdfed> open denm,HDF
hdfed> display
hdfed>
hdfed> ! The 'display' command displays the current RIS8
hdfed> ! group image via ICR. I.e. if you are using NCSA Telnet
hdfed> ! on a Mac II, this would display the images from denm.HDF
hdfed> ! on your screen.
hdfed> ! NOTE: not guaranteed to work otherwise.
hdfed>
```

The `putr8` command

Syntax: **putr8 [-image image filename palette filename] -verbose]**

Flags:

-image	Image file name template (Default is "img#.@.%")
-palette	Palette file name template (Default is "pal#")
-verbose	To give output of steps taken.

Description: Writes a RIS8 group into raw image and palette files.
Usage Example:

```
hdfed> ! putr8 puts an RIS8 group into raw files
hdfed>
hdfed> putr8 -image my_image.#.@.% -palette testPalettes# -verbose
Writing to file: my_image8.10.10
Writing to file: my_palette
hdfed>
```

The `close` command

Syntax: **close [-keep]**

Flags:

-keep	The backup file is not deleted.

Description: Closes the HDF file opened by the last open command.
Usage Example:

```
hdfed> close
```

```
hdfed>
```

The `select` command

Syntax: **select predicate list command list**

Flags: None.

Description: Step through all the elements in the HDF file that satisfies the predicates, and execute the command list.

Usage Example:

```
hdfed> ! To step through a file and, for example, putr8 on all
hdfed> ! RIS8 groups we can use the select command.
hdfed>
hdfed> select tag=306
>> putr8 -image testImages# -palette testPalettes# -verbose
>> end
Writing to file: testImages8
Writing to file: testPalettes8
Writing to file: test Images14
Writing to file: testPalettes14
Writing to file: testImages21
Writing to file: testPalettes21
hdfed>
hdfed> ! The 'select' and 'if' commands take the same
hdfed> ! predicates as 'next' and 'pref'. There are also
hdfed> ! the predicates 'succeed" and "fail" that test the
hdfed> ! return status of the 'last' command.
hdfed>
```

The `put` command

Syntax: **put [-file filename] [-verbose]**

Flags:

-file	Output file name (Default is "elt#.@")
-verbose	Output diagnostic information.

Description: Writes the raw binary image of the current object to a file.

Usage Example:

```
hdfed> ! The 'put' command writes an element into a binary file.
hdfed> ! This is a dumb routine and does not know about the
hdfed> ! formats of an element.
hdfed>
hdfed> put -file binary#
hdfed> put -file myBinary -verbose
Writing to file: myBinary
hdfed>
```

The `revert` command

Syntax: **revert**

Flags: None.

Description: Discards all changes made in the current hdfed session.

Usage Example:

```
hdfed> revert
hdfed>
```

The `getr8` command

Syntax: **getr8 image file name [x-dimension y-dimension] [-palette palette file name] [-raster|-rle|-imcomp]**

Flags:

-palette	Palette will be read from a binary file.
-raster	No compression will be performed during the write. (the default)
-rle	Run-length compression will be performed during the write.
-imcomp	IMCOMP compression will be performed during the write.

Description: Reads a RIS8 group from binary files.

The `if` conditional

```
Syntax:          if predicate list command list end
Flags:           None.
Description:     Executes commands in a loop if predicates are satisfied
                 for each element processed.
```

The `select` loop command

```
Syntax:          select predicate list command list end
Flags:           None.
Description:     Executes the list of commands for each element that
                 satisfies the predicates.
```

The `wait` command

```
Syntax:          wait message
Flags:           None.
Description:     Prints a message, then waits for a carriage return to be
                 typed.
```

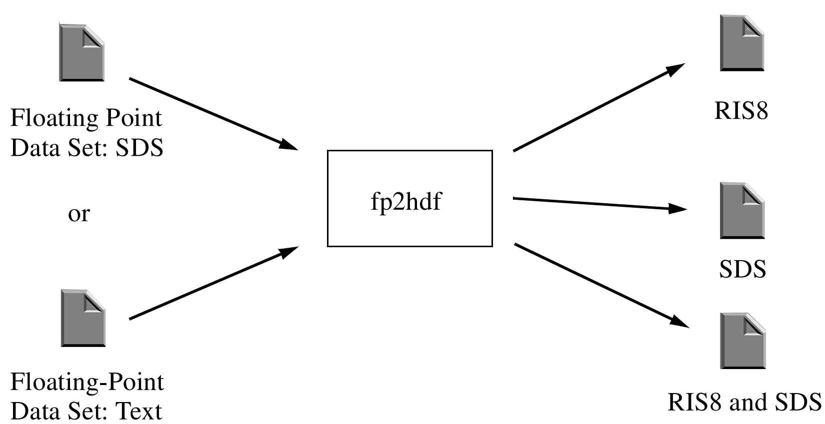
14.2.3 Converting Floating-Point Data to SDS or RIS8: fp2hdf

14.2.3.1 General Description

The `fp2hdf` utility converts 32-bit floating-point arrays from either text files or 32-bit HDF floating-point scientific data sets to either 8-bit HDF raster image sets or 32-bit floating-point HDF scientific data sets, or both, and stores the results in an HDF file. (See Figure 14a.) The images can be scaled on a user-specified mean value.

The term *scale* is used to refer to the area between points on the axes. If these gaps are of equal size, a uniform scale is specified - for example, "1.0, 2.0, 3.0, . . . ". In an HDF file, scales may be omitted, but in a text file (as in the text file shown below) they must be included.

FIGURE 14a **The fp2hdf Utility**



14.2.3.2 Command-Line Syntax

The syntax of `fp2hdf` is as follows.

```
fp2hdf input filename [-o output filename] [-r] [-f] [-e | -i]
   horizontal resolution vertical resolution [-p palette filename]
   [-m mean]
```

The input filename parameter is the name of the file containing the unconverted data set in HDF format. If the format is text, see the discussion on the following page about how it must be organized.

The output filename parameter is the name of the file containing the converted data set in HDF format. Depending on the output options output filename contains a scientific data set and/or raster image set for each of the data sets in the input files.

-e	Expand	Expands floating-point data using pixel replication (Default if -i option is specified)
-r	Raster	Stores the data as a raster image set in the output file.
-f	Float	Stores the data as a scientific data set in the output file. (Default if the "-r" option is not specified)
-i	Interpolation	Applies bilinear interpolation when expanding floating-point data. The horizontal resolution and vertical resolution parameters are, respectively, the horizontal and vertical resolution of the image
-p	Palette	Store the palette with the image. The palette filename parameter is the file name of the file containing the palette data.
-m	Mean	Cause the data to be scaled around the specified mean (supplied by the mean parameter) when generating the image, according to the following formulae: newmax = mean + 0.5*max(abs(max-mean), abs(mean-min) newmin = mean + 0.5*max(abs(max-mean), abs(mean-min))

The `-e` and the `-i` flags cannot be used simultaneously. Either pixel interpolation or bilinear interpolation can be chosen for image expansion, but not both.

If the `-i` option is chosen, the expanded image must have dimensions that are greater than or equal to the dimensions of the original data set.

An optional palette can accompany the image by loading it from an HDF file that contains a palette.

Data from several input files (with one set per input file) are stored as several data sets and/or images in one output file. A shell script can be used as another option in calling `fp2hdf` repeatedly to convert data from multiple input files to their corresponding output HDF files.

If an HDF file is used for input, it must contain an SDS. The SDS need only contain a dimension record and the data, but if it also contains maximum and minimum values and/or scales for the horizontal and vertical axes, these will be used as well.

If a text file is used for input, it must adhere to the following format.

```
number of rows number of columns
maximum value minimum value
scale for the vertical axis in an array
scale for the horizontal axis in an array
data element 1 data element 2 data element 3
...
```

The arrays that contain the scale for the vertical and horizontal axes must have a size equal to the values specified in the number of rows and number of columns positions, respectively. The data elements are floating-point data and are assumed to be ordered by rows, left-to-right and top-to-bottom.

14.2.3.3 Examples

This is the `fp2hdf` command-line syntax used to convert floating-point data in the file named "infile1.txt" to the SDS format, and to store it as an SDS in the HDF output file "outfile1".

```
fp2hdf infile1.txt -o outfile1
```

This `fp2hdf` command is used to convert floating-point data in the file named "infile2.txt" to an 8-bit raster image and store it in RIS8 format in the HDF output file named "outfile2".

```
fp2hdf infile2 -o outfile2 -r
```

This `fp2hdf` command is used to convert floating-point data in the file named "infile3.txt" to the RIS8 and SDS formats and store both converted data groups in the HDF output file "outfile3".

```
fp2hdf infile3.txt -o outfile3 -r -f
```

This `fp2hdf` command is used to convert floating-point data in the file named "infile4.txt" to a 500 x 600 raster image, storing the RIS8 in the HDF file "outfile4". This also stores the palette data read from the file name "palfile" with the image data.

```
fp2hdf input4.txt -o outfile4 -r -e 500 600 -p palfile
```

This `fp2hdf` command is used to convert floating-point data in all files whose names begin with the letter "f" to 500 x 600 RIS8 images and store them in the output file "output5".

```
fp2hdf f* -o outfile5 -r -i 500 600
```

14.2.4 Converting Several RIS8 Images to One 3D SDS: ristosds

14.2.4.1 General Description

The `ristosds` utility creates a single HDF file consisting of a three-dimensional SDS from a set of HDF files containing one or more raster images. All images in the input HDF files must have the same dimensions. If a palette is to be included with the images, it should be in the first HDF input file. Only one palette can be associated with the images; any additional palette data encountered by the utility after the first palette has been processed will be ignored.

14.2.4.2 Command-Line Syntax

```
ristosds input_filename_1, input_filename_2, ... input_filename_n
    [-o output_filename]
```

14.2.4.3 Examples

The contents of a directory consists of 20 files named "storm001.hdf", "storm002.hdf". ... "storm020.hdf". Each file contains a single RIS8 with a 100 x 200 raster image. A file that combines these 20 raster images into a 32-bit floating-point SDS with the dimensions 100 x 200 x 20 can be created with the following `ristosds` command:

```
ristosds storm*.hdf -o storm.hdf
```

14.2.5 Converting 8-Bit Raster Images to the HDF Format: r8tohdf

14.2.5.1 General Description

The `r8tohdf` utility converts a set of raw raster images to the HDF RIS8 format and writes them to a file.

14.2.5.2 Command-Line Syntax

```
r8tohdf [number_of_rows number_of_columns] output_filename
    [-p palette_filename] [-c|-r|-i] raw_raster_image_filename_1,
    raw_raster_image_filename_2, ... raw_raster_image_filename_n
```

`-p`	Palette File	Inserts a palette stored in the file palette_filename in the RIS8. If the -p flag is not specified, a palette is not stored with the RIS8.
`-c`	Run-length Encoding	Compresses the output data using run-length encoding.
`-i`	IMCOMP Compression	Compresses the output data using the IMCOMP method.
`-r`	No Compression	No compression is applied to the output data. (the default)

14.2.5.3 Examples

A file named "rawras" contains a 256 x 512-byte raw raster image, and its palette is stored in a file name "mypal". To convert the information in these files to an RIS8 without compression and store the RIS8 in a file named "ras.hdf", enter the following `r8tohdf` command:

```
r8tohdf 256 512 ras.hdf -p mypal rawras
```

A 800 x 1000-byte raw raster image is stored in a file named "bigpic". This data must be converted to a RIS8 without a palette, compressing it using run-length encoding, then stored in a file named "bigpic.hdf". The following command will do this:

```
r8tohdf 800 1000 bigpic.hdf -c bigpic
```

A 300 x 400 raw raster image is contained in each of the files named "pic1", "pic2", and "pic3". To convert all three files to RIS8s, compress them using the IMCOMP method, and store them in a file named "pic.hdf", enter

```
r8tohdf 300 400 pic.hdf -i pic1 pic2 pic3
```

Different types of raster image data are to be stored in a file named "ras.hdf". The image data in the file "rawras1" will be stored without a palette. The image data sets from the file named "rawras2" are to be stored with a palette extracted from a file named "mypal". The images from the "rawras1" and "rawras2" files are to be compressed using run-length encoding, and the image in the "rawras3" file is not to be compressed. The size of all images are 256 x 512 bytes. The following command is used to do this:

```
r8tohdf 256 512 ras.hdf -c rawras1 -p mypal rawras2 -r rawras3
```

14.2.6 Extracting 8-Bit Raster Images and Palettes from HDF Files: hdftor8

14.2.6.1 General Description

The `hdftor8` utility extracts the raster images and/or palettes from an HDF file and stores them in one file that contains the raster image data and another that contains the palette data.

14.2.6.2 Command-Line Syntax

```
hdftor8 input HDF filename [-i] [-v] [-r raster image filename]
    [-p palette filename]
```

`-i`	Interactive Mode	Program is executed in interactive mode.
`-v`	Verbose Mode	Program is executed in verbose mode. Diagnostic messages are displayed during the session.
`-r`	Raster Image File Name	The raster image file name immediately follows this flag.
`-p`	Palette File Name	The palette file name immediately follows this flag.

The names given as the HDF format file, raster image file, and palette file are interpreted by `hdftor8` as follows: For each raster image file, the file name is given the extension

```
.#.@.%
```

where "#" represents the raster image number from the HDF file, "@" represents the x-dimension of the raster image and "%" represents the y-dimension of the raster image. For each palette file, the file name is given the extensions ".#", where "#" represents the palette number from the HDF format file.

If no name is given for the raster image file, the default name "img.#.@.%" is assigned, where "#", "@", and "%" are defined as in the preceding paragraph. The default name for a palette file, if no name is specifically given in the command, is "pal.#".

14.2.6.3 Examples

A file named "denm.hdf" contains three 512 x 256 raster images and three palettes. To store these images and palettes in separate raster image and palette files, use the following `hdftor8` command:

```
hdftor8 denm.hdf
```

Six files are created, named "img1.512.256:", "img2.512.256", "img3.512.256", "pal.1", "pal.2", and "pal.3".

14.2.7 Compressing RIS8 Images in an HDF File: hdfcomp

14.2.7.1 General Description

The `hdfcomp` utility reads RIS8 images from a set of HDF files, compresses them and stores the compressed data in a second HDF file. If the output HDF file exists, the compressed images will be appended to it.

14.2.7.2 Command-Line Syntax

```
hdfcomp output_filename [-c|-r|-i] input_filename_1,
   [-c|-r|-i] input_filename_2, ... [-c|-r|-i] input_filename_n
```

-r	No compression	The raster image data is not compressed. (the default)
-c	Run-length Encoding	The raster image data is compressed using run-length encoding.
-i	IMCOMP Compression	The raster image data is compressed using the IMCOMP algorithm.

14.2.7.3 Examples

A directory contains twenty files named "storm001", "storm002", ... "storm020". Each of these files contains a single RIS8 image. To compress these images using run-length encoding and store them in a file named"altcomp.hdf", use the following `hdfcomp` command:

```
hdfcomp allcomp.hdf -c storm*.hdf
```

14.2.8 Converting 24-Bit Raw Raster Images to RIS8 Images: r24hdf8

14.2.8.1 General Description

The `r24hdf8` utility quantizes a raw RGB 24-bit raster image, creating an 8-bit image with a 256-color palette, then it stores the palette and raster image data in an HDF file.

14.2.8.2 Command-Line Syntax

```
r24hdf8 [x-dimension length y-dimension length] raw 24-bit image file-
    name hdf ris8 image filename
```

The pixel order in the raw 24-bit image file is left-to-right and top-to-bottom. Each pixel data element consists of three contiguous bytes, the first representing the red intensity value, the second the green intensity value, and the third the blue intensity value. Use the `ptox` filter to convert the raster image data from a pixel-interlaced format to scan-plane interlaced.

14.2.8.3 Examples

A file named "rawraster" containing 24-bit raw raster images with x and y-dimensions of 480 x 640, respectively, must be converted to the HDF RIS8 format and stored in a file named "hdfraster". The following command is used to do this:

```
r24hdf8 480 640 rawraster hdfraster
```

14.2.9 Converting an HDF RIS24 Image to an HDF RIS8 Image: hdf24hdf8

14.2.9.1 General Description

The `hdf24hdf8` utility quantizes an HDF RGB RIS24 pixel-interlaced image, producing an HDF RIS8 image with a 256-color palette and stores the palette and raster image data in an HDF file.

14.2.9.2 Command-Line Syntax

```
hdf24hdf8 ris24 image filename ris8 image filename
```

14.2.10 Converting Raw Palette Data to the HDF Palette Format: paltohdf

14.2.10.1 General Description

The `paltohdf` utility converts raw palette data to the HDF palette format. The raw palette data must have 768 bytes organized in the following order: first, 256 contiguous red intensity values, then 256 contiguous green intensity values, then 256 contiguous blue intensity values. The palette in the HDF file will have the RGB values pixel-interlaced, as follows.

```
red value green value blue value red value green value
    blue value ...
```

This is the standard HDF format for 8-bit palettes.

14.2.10.2 Command-Line Syntax

```
paltohdf raw format palette filename HDF format palette filename
```

If a HDF palette format file is specified that doesn't exist, it is created before the converted data is stored. If an HDF palette format file is specified that already exists, the converted data is appended to the file.

14.2.11 Extracting Palette Data from an HDF File: hdftopal

14.2.11.1 General Description

The `hdftopal` utility converts a palette in an HDF file to a raw palette in an non-HDF file. The raw palette will have 768 bytes with the first 256 bytes representing red intensity values, the second 256 bytes representing green intensity values, and the third 256 bytes representing blue intensity values. The utility performs the converse operation of the `paltohdf` utility.

14.2.11.2 Command-Line Syntax

```
hdftopal HDF_format_palette_filename raw_format_palette_filename
```

14.2.12 Compressing an HDF File: hdfpack

14.2.12.1 General Description

The `hdfpack` utility compresses all of the data in an HDF file and writes the compressed data to a second HDF file.

14.2.12.2 Command-Line Syntax

```
hdfpack [-i|-b] [-d number_of_data_descriptors_per_block]
    [-t number_of_linked_blocks_per_table_entry] input_HDF_filename
    output_HDF_filename
```

`-b`	Non-coalesced Blocks	The utility will not coalesce linked-block elements.
`-i`	Interactive Mode	The utility will prompt for each linked-block element.
`-d`	Data descriptors per block	The output file will be created with the specified number of data descriptors per block of data descriptors.
`-t`	Linked-blocks per table entry	The output file will be created with the specified number of linked blocks per table entry.

14.2.12.3 Examples

To compress the data in the file named "aa.hdf" and store the compressed data in the file named "aa.cmp", use the following `hdfpack` command:

```
hdfpack aa.hdf aa.cmp
```

Suppose a file named "bb.hdf" contains data elements stored as sequences of linked blocks. The following `hdfpack` command compresses the file while leaving the linked-block elements intact, and writes the compressed data to a file named "bb.blk".

```
hdfpack -b bb.hdf bb.blk
```

14.2.13 Displaying Vdata Information: vshow

14.2.13.1 General Description

Displays information about either one Vdata object in an HDF file, or all Vdata objects in the file.

14.2.13.2 Command-Line Syntax

```
vshow input HDF filename [+|+vdata_id]
```

+	All Vdatas	The utility will display information about all Vdata objects in the HDF file.
+vdata_id	One Vdata	The utility will display information about the Vdata object corresponding to the specified vdata id.

14.2.13.3 Examples

Information about all of the Vdata objects in the HDF file named "image012.hdf" must be examined. The following command will display this information.

```
vshow image012.hdf +
```

14.2.14 Displaying General Information About the Contents of an HDF File: hdp

14.2.14.1 General Description

The `hdp` utility provides quick and general information about all objects in the specified HDF file. It can list the contents of HDF files at various levels with different details. It can also dump the data of one or more specific objects in the file.

14.2.14.2 Command-Line Syntax

```
hdp [-H command] filename
```

-H	Help	Displays usage information about the specified command. If no command is listed, information about all commands are displayed.

Like `hdfed`, hdp provides a set of commands that allow the user to determine what kind of information is to be displayed.

TABLE 14C **The `hdp` Command Set**

Name	Description
list	Displays the contents of the HDF files in the specified format.
dumpsds	Displays the contents of the SDSs in the listed files.
dumpvd	Displays the contents of the vdata objects in the listed files.
dumpvg	Displays the contents of the vgroup objects in the listed files.
dumprig	Displays the contents of the RIGs in the listed files.

The `list` command

Syntax: **list [-s|-l|-d] [-n|-c|-a] [-g|-t __number__ __name__] [-ot|-of|-og|-on] __message__**

Flags:

-s	Short format.
-l	Long format.
-d	Debug format.
-n	Display the object name.
-c	Display the object class.
-a	Display the object description.
-g	Display groups only.
-t	Display objects with the specified tag number or name.
-ot	Sort by tag.
-of	Sort by position in the data descriptor list.
-og	Sort by group.
-on	Sort by name.

Description: Display the contents of the HDF files in the specified format. As with the 'info' command, the listing for specialelements will contain a special tag value (for DFTAG_VS it's 18347) and the text "Unknown Tag".

The `dumpsds` command

Syntax: **dumpsds [-i __index__ | -r __ref list__ | -n __name list__ | -a] [-v|-h|-d] [-o __filename__ | -b | -t]**

Flags:

-i	Dump SDSs with the specified index.
-r	Dump SDSs with the specified reference numbers.
-n	Dump SDSs with the specified names.
-a	Dump all SDSs.
-v	Display all SDS contents including annotations.
-h	Display the SDS header only.
-d	Display the SDS data only.
-o	Print information to the specified file.
-b	Specify output file as binary.
-t	Specify output file as text.

Description: Displays SDS information in the specified format.

The `dumpvd` command

Syntax: **dumpvd [-i __index__ | -r __ref list__ | -n __name list__ | -c __class__ | [-a] [-v|-h|-f|-d] [-o __filename__ | -b | -t]**

Flags:

-i	Dump vdatas with the specified index.
-r	Dump vdatas with the specified reference numbers.
-c	Dump vdatas with the specified class.
-n	Dump vdatas with the specified names.
-a	Dump all vdatas.
-v	Display all vdata contents including annotations.
-h	Display the vdata header only.
-d	Display the vdata data only.
-f	Display the vdata field data only.

-o	Print information to the specified file.
-b	Specify output file as binary.
-t	Specify output file as text.

Description: Displays vdata information in the specified format.

The dumpvg command

Syntax: **dumpvg [-i index | -r ref list | -n name list | -c class | [-a] [-v|-h|-d] [-o filename | -b | -t]**

Flags:

-i	Dump vgroups with the specified index.
-r	Dump vgroups with the specified reference numbers.
-c	Dump vgroups with the specified class.
-n	Dump vgroups with the specified names.
-a	Dump all vgroups.
-v	Display all vgroup contents including annotations.
-h	Display the vgroup header only.
-d	Display the vgroup data only.
-o	Print information to the specified file.
-b	Specify output file as binary.
-t	Specify output file as text.

Description: Displays vgroup information in the specified format.

The dumprig command

Syntax: **dumprig [-i index | -r ref list | -m <8, 24> | [-a] [-v|-h|-d] [-o filename | -b | -t]**

Flags:

-i	Dump RIGs with the specified index.
-r	Dump RIGs with the specified reference numbers.
-m	Dump RIGs with the specified data length - 8- or 24-bit.
-a	Dump all RIGs.
-v	Display RIG contents including annotations.
-h	Display the RIG header only.
-d	Display the RIG data only.
-o	Print information to the specified file.
-b	Specify output file as binary.
-t	Specify output file as text.

Description: Displays RIG information in the specified format.

14.2.15 The HDF User-Contributed Utilities

In addition to the command-line utilities supported by NCSA, a number of utilities have been contributed by HDF users. Although they are not supported by NCSA, these utilities are distributed by NCSA via anonymous ftp at hdf.ncsa.uiuc.edu in the "/pub/dist/HDF/contrib" directory. These utilities, which are listed and briefly described in Table 14D, will not be discussed in this manual.

Note that this list is current as of the release date of this manual. A updated list of all of the user-contributed HDF utilities is available in the "README" file at the HDF ftp site and directory location mentioned above.

TABLE 14D

HDF User-Contributed Utilities

Name	Description
yuvconvert	Converts raster images in an HDF file to mpeg format.
LinkWinds	Interactively accesses, displays and performs analysis on multidisciplinary data sets. Includes its own graphical user interface.
fixver	Removes garbage data from the end of the version tag in the specified file for the benefit of applications, like the Macintosh port of NCSA Collage, that do not deal with this garbage data well.
ReadDF	Reads the contents of an HDF or a netCDF file into the Silicon Graphics Iris Explorer.
hdfinfo	Displays general information about the contents of an HDF file.
hdf24seq	Sequentially displays 24-bit raster images in HDF files. Tested on IBM RS/6000 and some SGI workstations.
mhdf24seq	Provides the same functionality as `hdf24seq`, but allows the user to display raster image data from more than one HDF file at a time.
vs2ris	Converts Vset data in the specified file to HDF RIS format.
vs2ps	Converts Vset data in the specified file to PostScript format.
inspectHDF	Displays general information about the contents of an HDF file.
fits2hdf	Converts FITS-formatted data to HDF format.
fix32luf	Fixes a bug in HDF version 3.2 revisions 1, 2 and 3 where SDS strings are written incorrectly.
sds2ris	Converts SDS data in the specified input file to RIS format, then writes the converted data to the specified output file.
cgmct.to.raster.npoc	Converts graphical data from a CGM clear text metafile, as outputted by CA-DISSPLA and CA-GKS to either raw raster images and CLUTs and stores it in a generic output file, or to RIS8 data and stores it in an HDF file.
cgmct.to.raster.withpc	Converts graphical data from a CGM clear text metafile, as outputted by CA-DISSPLA and CA-GKS to either raw raster images and CLUTs and stores it in a generic output file, or to RIS8 data and stores it in an HDF file, or a NERSC compressed movie file.
hdfrseq_new	New `hdfrseq` routine with SunView features.
hdftodico	Converts an HDF-formatted file to a Dicomed DCN file.
hdftops	Converts HDF raster image data to PostScript format.
iristohdf	Converts data in the Silicon Graphics image format to the HDF format.
isistoddc	Converts data in the Silicon Graphics image format to the Dicomed D48 DDC format.
xwdtohdf	Converts data in the X-Windows display format to the HDF RIS format.
hdftoxwd	Converts data in the HDF RIS format to the X-Windows display format.
qdvtohdf	Converts data in the qdv format to the HDF RIS format.
paltohdf	Converts raw palette data to the HDF palette format.
hdfxdis	Displays HDF RIS data on an X-Windows server.
hdf2tiff	Converts HDF RIS data to the TIFF image format.
tiff2hdf	Converts data in the TIFF image format to the HDF RIS format.
hdf2ras	Converts HDF RIS data to the Sun Rasterfile format.
ras2hdf	Converts raster image data in the Sun Rasterfile format to the HDF RIS format.
vs2ps	Converts polygonal Vset data to contour plots in PostScript format.
vs2ris	Converts polygonal Vset data to HDF RIS format.