# HDFView User's Guide

HDFView is a graphic utility designed for viewing and editing the contents of HDF4 and HDF5 files. This document provides the following information:

- User instructions for HDFView
- A brief discussion of the HDF object model (Details of the HDF object model are available from HDF Object Package.)

# Table of Contents

# Chapter 1: Introduction



---

## 1.1 Overview

HDFView is a Java-based tool for browsing and editing HDF4 and HDF5 files. HDFView allows users to browse through any HDF4 and HDF5 file, starting with a tree view of all top-level objects in an HDF file's hierarchy. HDFView allows a user to descend through the hierarchy and navigate among the file's data objects. The content of a data object is loaded only when the object is selected, providing interactive and efficient access to HDF4 and HDF5 files. HDFView editing features allow a user to create, delete, and modify the value of HDF objects and attributes.

The HDFView graphical user interface (GUI) is simple and easy-to-use. First, HDFView was implemented by using the Java™ Platform and SWT™ UI, which is machine-independent. The GUI components have the same native look-and-feel for the machine. Second, HDFView uses conventional folders and icons to display groups and datasets in a tree structure. Users can easily expand or collapse folders to navigate the hierarchical structure of an HDF file. Third, HDFView shows data content as text (table or plain text) or as an image.

## 1.2 About This Release

For information about a specific HDFView release, see the release notes on the Download HDFView page.

## 1.3 Further Information

General information about HDF (HDF4 and HDF5) is available at https://www.hdfgroup.org/. Information about the HDFView tool and related products is available from the HDF-Java Home page.

---

*[Table of Contents]*

# Chapter 2: Getting Started

This chapter assumes that you have installed HDFView and, along with "Chapter 3: The HDF Object Model," is designed to help you get to the point where you are actually looking at HDF datasets.

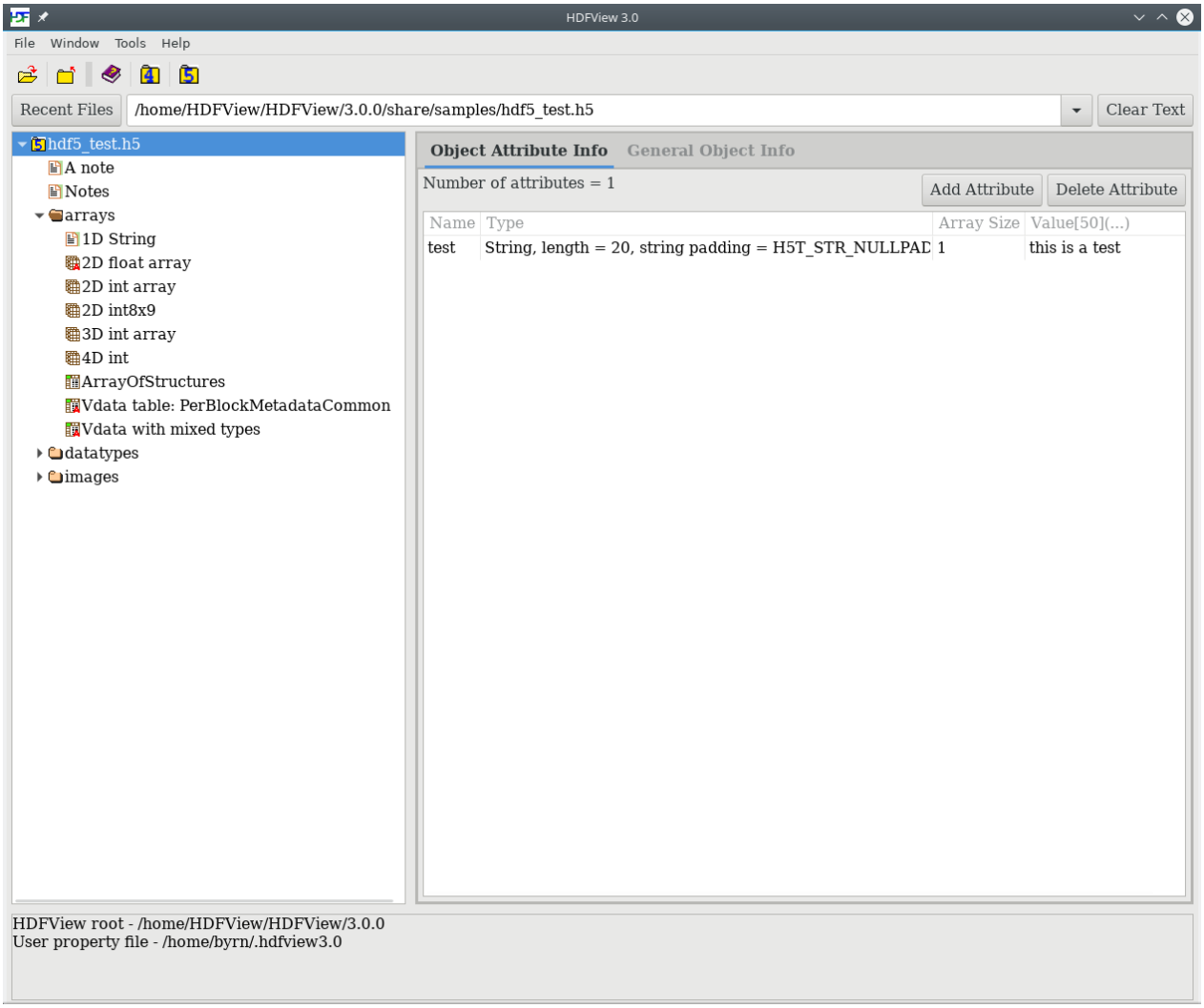


---

## 2.1 The Main Window

When you first open HDFView, the HDFView window appears with an empty tree and metadata panel. After you open an HDF file, the structure of the file is displayed in the Tree Panel. The metadata content of a data object is displayed in the metadata panel by selecting an object.

The main window consists of five components: **Menu bar, Tool bar, File bar, Tree panel, Metadata Panel, Info panel.**




**The main window with a file**

- ***Menu bar***
  The Menu bar is where you choose menu commands: “File”, “Window”, “Tools”, and “Help”.
- ***Tool bar***
  The Tool bar is located under the menu bar and displays buttons with icons that are shortcuts for commonly performed tasks, such as opening and closing a file, zooming in or out of an image, or flipping forward and backward through data pages.
- ***File bar***
  The File bar consists of a Recent Files button which allows you to select among a list of files that you have recently opened. In addition to this is a File/URL text field in which you enter information that leads or points to a file that you want to open. Valid file names include an absolute file name, a path name, and a URL to a remote file, such as /samples/hdf5_test.h5.
  Using a path name and selecting multiple files will open all the selected files.
  Valid remote file prefix:
  - “http://”

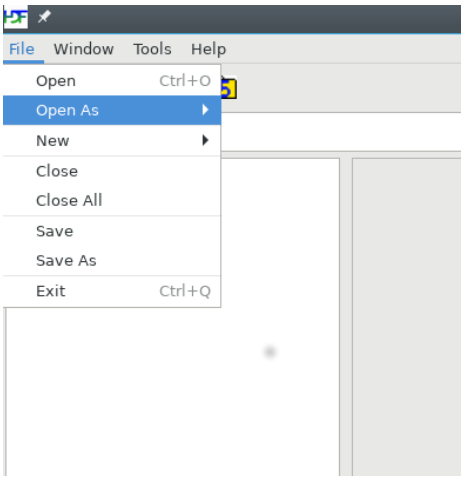
- "https://"
- "ftp://"

- ***Tree panel***
  The Tree panel is the left panel in the HDFView window. It displays the group structure of an HDF file as a tree of "folders." Other objects, such as datasets and images, are displayed as leaf objects in the tree.
- ***Metadata panel***
  The metadata panel is the right panel in the HDFView window where the metadata content of objects are displayed in a tabbed view. The "Object Attribute Info" tab displays a list of attributes attached to the selected object. The attribute name, type, size and a short sample of data is displayed. Double-clicking on an attribute line will open a Data window for the selected attribute. There is an "Add Attribute" button to create a new attribute attached to the currently selected object in the treeview. And a "Delete Attribute" button to remove the selected attribute.
  The "General Object Info" tab displays object information such as names of objects, member names and types, or paths, type and version bounds for file types.
- ***Info panel***
  The Info panel is the bottom panel that spans the width of the HDFView window. It displays status information. A status message includes file or data information, warning or error messages and the status of data processing

### 2.1.1 Menu bar

The Menu bar is at the top of the HDFView window. You can select a menu command from menu items or press key combinations from the keyboard to invoke the menu item's action without navigating through the menu hierarchy.

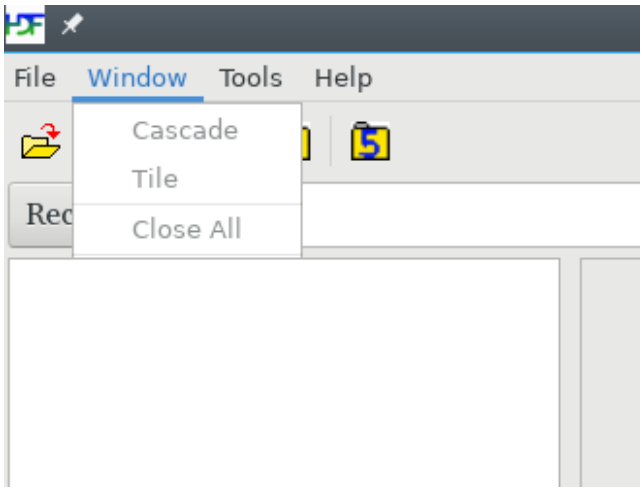
***File menu***
The File menu contains commands to open and close files, create and save files, and exit HDFView.




**File menu**

***Window menu***

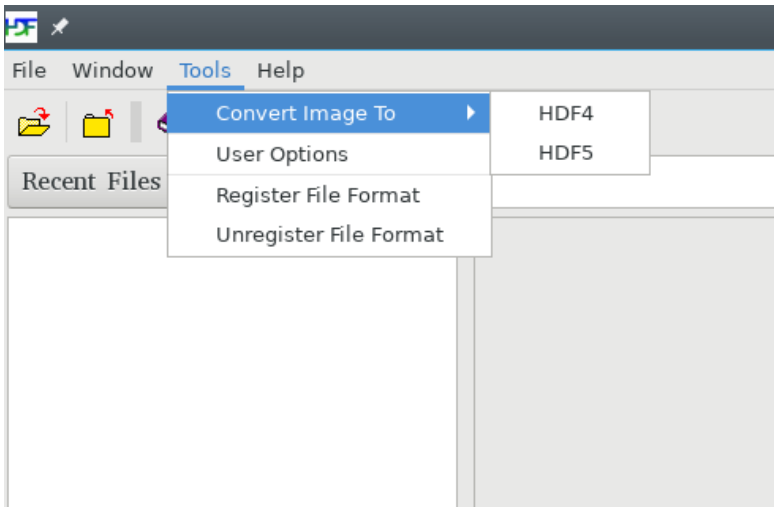
The Window menu displays a list of currently open data windows along with the close, tile, and cascade window commands. You can bring a data window to the front by choosing the data name from the Window menu. The “Cascade” and “Tile” commands cascade or tile all open windows. The “Close” command closes the *active* data window. The “Close All” command closes all open windows. All open data windows are listed at the bottom of the Window menu. Select one of the items on the list to bring that window to the front.




***Window menu***

***Tools menu***

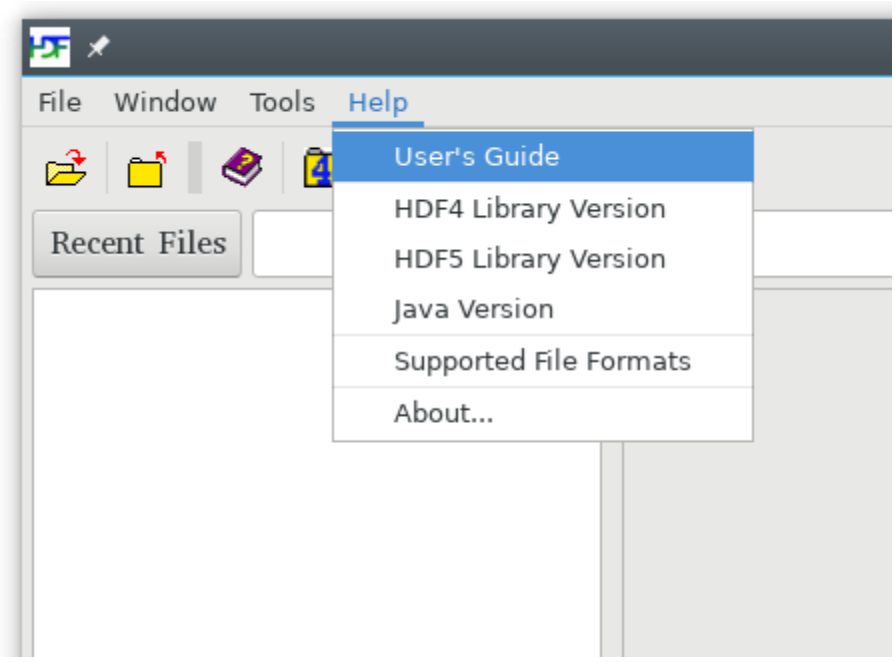
The Tools menu contains a list of commands to launch tools such as image conversion and user options. The “User Options” command launches a dialog box that allows you to change user preferences: the path of the User's Guide, font size, delimiter and more. For more information, see Chapter 7.




**Tools menu**

***Help menu***

The Help menu displays the User's Guide, the version of the HDF4, HDF5 and Java libraries, currently supported file formats and the About box.



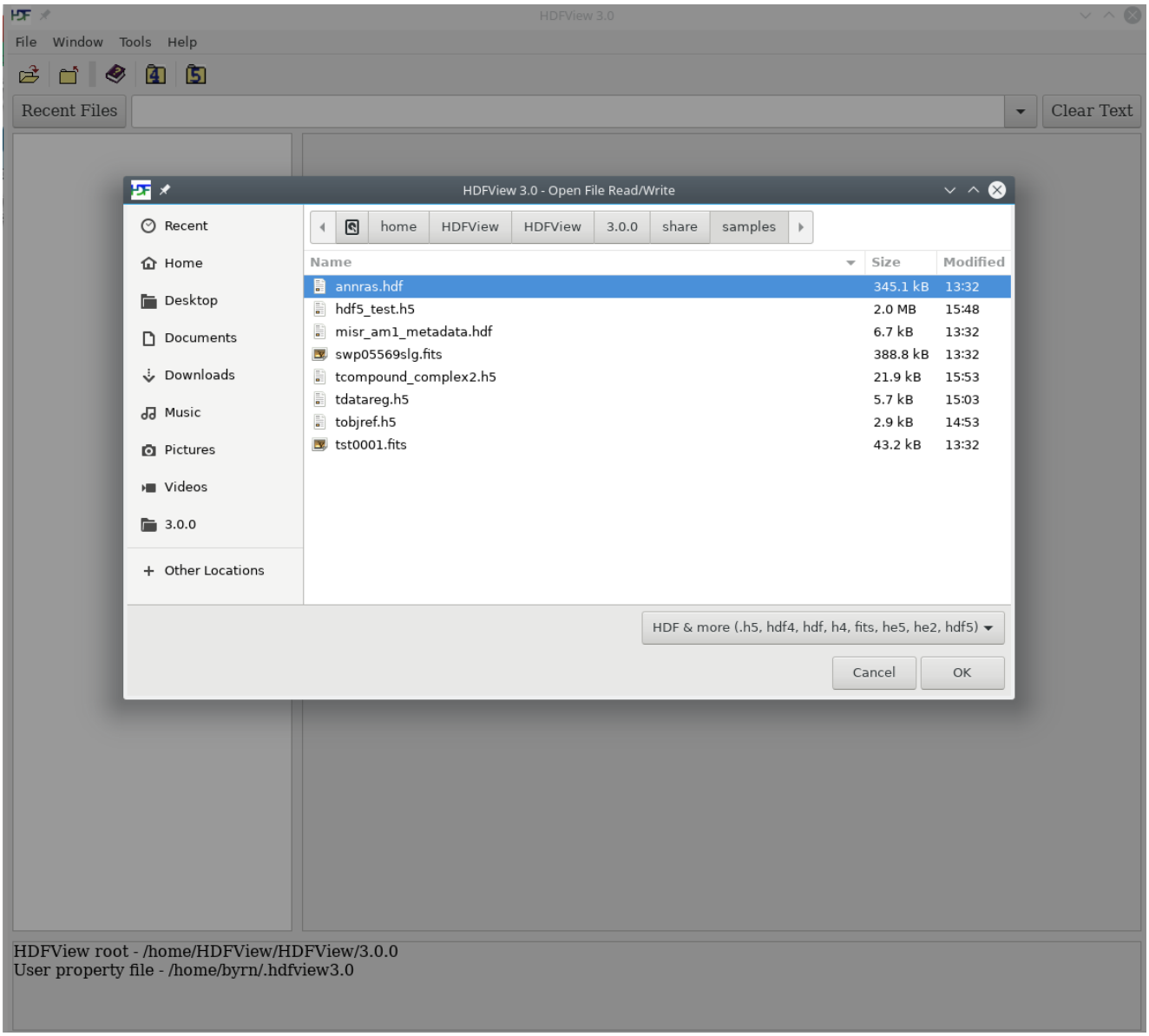

**Help menu**

# 2.2 Opening a File

Select the “Open” command from the file menu or click the “Open” icon in the tool bar to invoke the local file manager, and select a file to open from the local file manager. Selecting “Open As Read-Only” opens a file with read-only permission, which means that editing functions are disabled and changes are not allowed. Selecting “Open As Read/Write” opens a file with read

and write permission, hdf5 will modify the file even if no changes are saved.

Note that HDFView defaults to opening a file as read-only unless the default is changed in the “User Options/General Settings” Default File Access Mode section (see 7.1.3 File Access Mode). By default, HDFView will only show files that have a file extension matching those specified in the “User Options/HDF Settings” File Extension list (see 7.2.1 File Extensions), however, the user can click on the drop-down box in the open file dialog box and select "All Files" in order to show all files if needed.




**Local file manager**

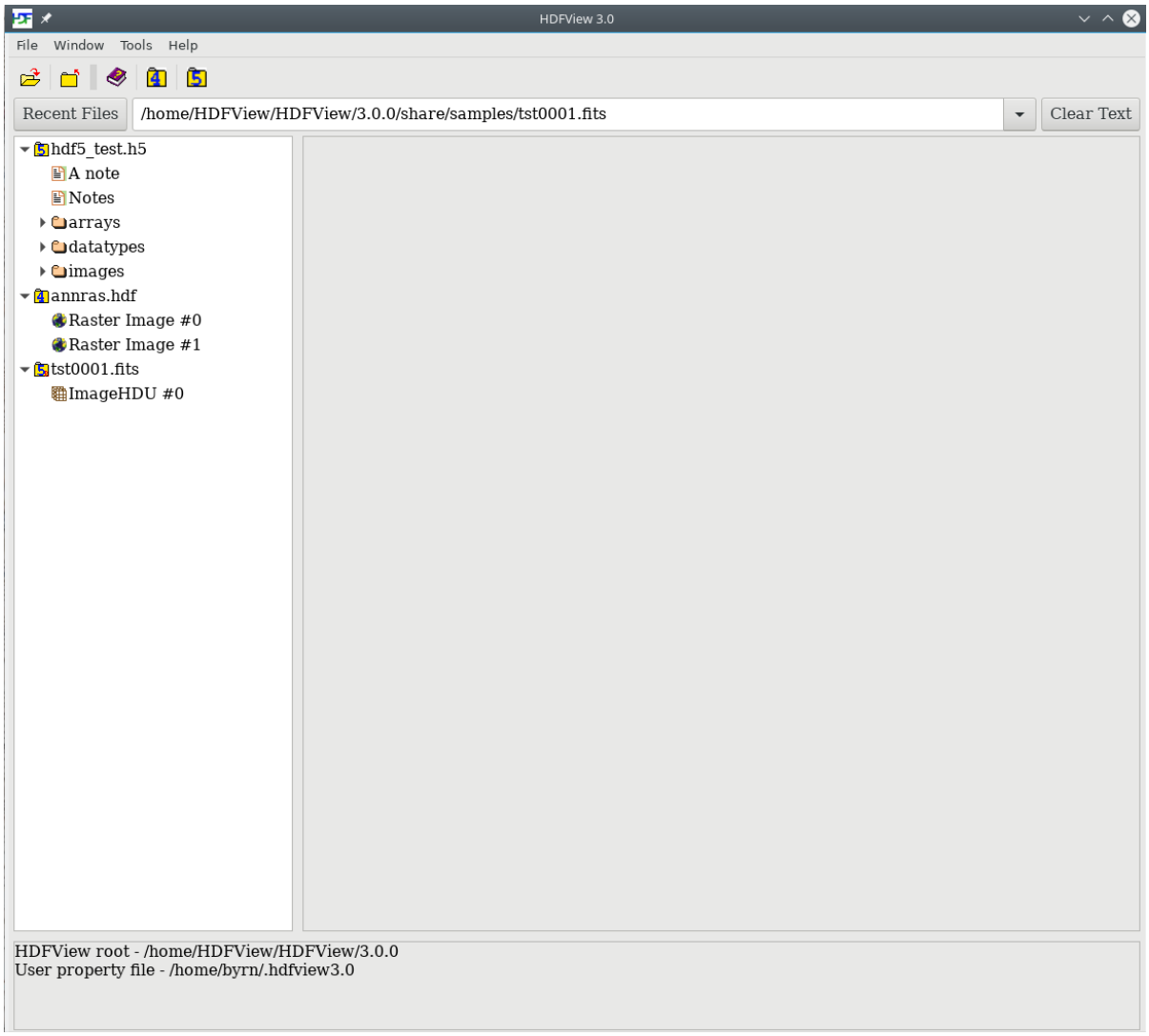
# 2.3 Tree View of File Hierarchy

An HDF file contains one or more objects, optionally grouped in a hierarchy. For general information about the structure and contents of HDF files, see the HDF documentation at HDF5 Documentation and HDF4 Documentation.

The structure of the file is displayed in the Tree panel, which is the left panel in the HDFView window. Data objects are represented as icons, and groups are represented by folders. An HDF file may contain groups, datasets (arrays), committed datatypes, and links.

The hierarchy of the file can be navigated by selecting folders to “open” a group, which displays the objects contained by the group. When an object is selected (by double clicking the object or by right clicking the object and selecting "Open" or "Open As" from the popup menu), the data is displayed appropriately in a window (usually a table or spreadsheet type.).

You can open multiple files at one time. Files are listed as root objects in the tree. The following figure illustrates that an HDF5 file, hdf5_test.h5, and an HDF4 file, annras.hdf, are currently open.




**Tree view**

You can also search for data objects within a file by right-clicking on the file and selecting "Find" from the Context menu. Type in the name of a data object to look for and HDFView will highlight the first occurrence of the object within the file. The wildcard '*', which specifies that any number of characters can come after the text before the '*' character, can be used to broaden your search. As an example, searching for 'dataset*' will find any objects whose name is a variation on 'dataset', i.e. 'dataset', 'dataset_int', 'dataset75', etc.

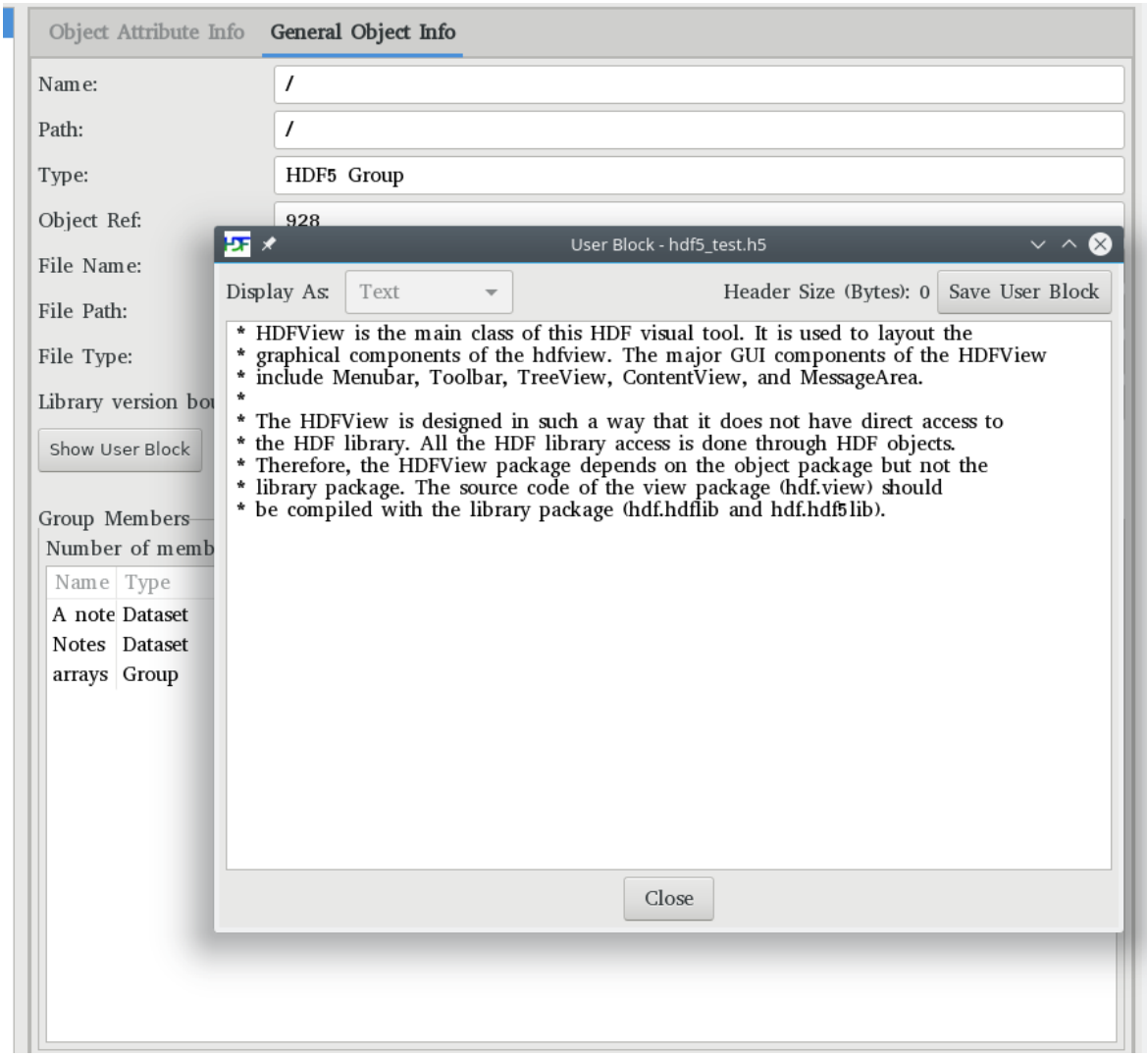
# 2.4 Viewing Status Information

Short status information and feedback messages are displayed in the information panel. Such information includes error messages and feedback from user's actions.

# 2.5 Viewing HDF Metadata

To see the metadata of a data object, left-click on the data object. Then, choose the “Object Attribute Info” tab to display a list of attributes attached to the selected object. The “General Object Info” tab displays object information such as names of objects, member names and types, or paths, type and version bounds for file types.

To view a user block in text, left-click the file node (the root group) and select the “General Object Info” tab. Click on the “Show User Block” button to show the user block information in text.




**User block in text**

# 2.6 Command-line Options

The following command line options are available to HDFView:

`-root  path`
specify the working directory to start HDFView in

`-g, -geometry  WIDTHxHEIGHT+XOFF+YOFF`
specify the dimensions and location of the HDFView window

`-java.version`
display java version information

`<filename>`
open the specified file/s upon start

---

*[Table of Contents]*

# Chapter 3: HDF Object Model

This chapter provides basic information about the HDF Object Model.



---

## 3.1 Overview

HDF files may contain many types of data objects that a scientist might need. HDFView displays the data objects appropriately according to their types. For example, a two-dimension dataset with an associated palette will be displayed as an image. When you open an HDF file with HDFView, you see the tree structure of an HDF file, showing the objects and their groupings. You can select an object from the tree to view its content.

HDF4 (i.e. HDF version 4) is based on the original 1988 version of HDF. Versions 1, 2, 3, and 4 of HDF were all backward compatible, and HDF4 can access files created by all earlier versions. HDF5 is a completely new format and library. Although they are conceptually related, HDF5 files cannot be read by the HDF4 library and vice versa. HDF5 was designed to address some of the limitations of HDF4 and to address and exploit current and anticipated requirements of modern systems and applications.

HDFView is built on a common HDF object model and supports both versions of HDF. The HDF object model was designed in such a way that the HDF4 and HDF5 objects interact with users through a common object layer so the user interface design will be independent of the file format (HDF4 or HDF5). Such a design allows the HDFView to support and convert objects of different formats (HDF4 and HDF5).

## 3.2 The HDF Object Package

The **HDF Object Package** is a Java package that implements HDF4 and HDF5 data objects in an object-oriented form. The HDF Java Object Package provides common standard Java APIs to access both HDF4 and HDF5 files.

The HDF Object Package is **NOT** a “wrapper” for the native HDF libraries, and it requires the HDF4 and HDF5 wrappers. The HDF4 and HDF5 wrappers are separate HDF Java products. For details about the HDF4 and HDF5 native interfaces, read Java HDF Interface (JHI) and the Java HDF5 Interface (JHI5).

The HDF Object Package implements higher level APIs and encapsulates HDF library calls into an object-oriented fashion for easy access to HDF files. For example, to retrieve data content

from an HDF5 dataset by using the HDF5 library APIs directly, you have to make many calls, such as: get the datatype information (datatype class, size, sign, and etc), get the dataspace information (number of dimension, dimension sizes), and allocate the data buffer. The HDF Object Package puts all these calls into a single call - read().

The HDF Object Package, *hdf.object*, provides classes that reflect fundamental concepts to the design of HDF objects. Objects of HDF5 (group and dataset) and HDF4 (group, multi-dimension array, raster image, vdata and annotation) are presented as Java classes.

The HDF Object Package has two major goals. First, it simplifies the process of reading information from or writing data to a file because the details of accessing the HDF library are encapsulated into respective classes. Second, HDF4 and HDF5 objects are inherited from the same common object and interface. Applications can use the HDF Object Package to access objects from either HDF4 or HDF5 in a uniform way, without accessing the libraries directly. The following diagram explains the relationship of the object package, HDF JNI, and application.

**HDF Applications <==> HDF Object Package <==> HDF4/5 Java Wrapper (JNI4/5) <==> HDF File**

## 3.3 Class Hierarchy

The HDF Object Package implements an abstract data model, and the objects of the HDF4 and HDF5 data models are represented as instances of the abstract objects. The abstract class **HObject** has three fundamental abstract classes, **Group**, **Datatype** and **Dataset**, and all HDF5 and HDF4 objects are a sub-type of one of these abstract classes.

For details, see the Java docs.

## 3.4 Using the HDF Object Package

The HDF Object Package is used by Java applications to access HDF4 and HDF5 files without directly calling the HDF4 and HDF5 library APIs. Library calls are encapsulated into respective classes. The HDF Object Package requires the Java HDF Interface (JHI) and the Java HDF5 Interface (JHI5).

The following examples show how to retrieve file hierarchy using the HDF Object Package.

**Example 3.1: Retrieve and print HDF5 objects**

```
import hdf.object.*;     // include the common HDF object package
import hdf.object.h5.*;  // include the HDF5 object package
import hdf.hdf5lib.*;    // include the Java HDF5 interface
```

```
/**
 * Retrieve and print HDF5 objects from file hdf5_test.h5
 * @version 1.3.0 10/26/2001
 * @author Peter X. Cao
 *
 */
public class TestH5File
{
    public static void main(String[] argv)
    {
        // create an H5File object
        H5File h5file = new H5File("hdf5_test.h5",
HDF5Constants.H5F_ACC_RDONLY);

        try
        {
            // open file and retrieve the file structure
            h5file.open();
        }
        catch (Exception ex)
        {
            System.out.println(ex);
        }

        javax.swing.tree.MutableTreeNode root = h5file.getRootNode();
        if (root != null)
        {
            printNode(root, "    ");
        }

        try { h5file.close(); }
        catch (Exception ex ) {}
    }

    // print out the data object recursively
    private static void printNode(javax.swing.tree.TreeNode node, String
indent)
    {
        System.out.println(indent+node);

        int n = node.getChildCount();
        for (int i=0; i<n; i++)
        {
            printNode(node.getChildAt(i), indent+"    ");
        }
    }
}
```

### Example 3.2: Retrieve and print HDF4 objects

```
import hdf.object.*;    // include the common HDF object package
import hdf.object.h4.*; // include the HDF4 object package
import hdf.hdflib.*;    // include the Java HDF5 interface

/**
 * Retrieve and print HDF4 objects from file annras.hdf.
 * @version 1.3.0 10/26/2001
```

```
 * @author Peter X. Cao
 *
 */

public class TestH4File
{
    public static void main(String[] argv)
    {
        // create an H4File object
        H4File h4file = new H4File("annras.hdf", HDFConstants.DFACC_READ);

        try
        {
            // open file and retrieve the file structure
            h4file.open();
        }
        catch (Exception ex)
        {
            System.out.println(ex);
        }

        javax.swing.tree.MutableTreeNode root = h4file.getRootNode();
        if (root != null)
        {
            printNode(root, "    ");
        }

        try { h4file.close(); }
        catch (Exception ex ) {}
    }

    // print out the data object recursively
    private static void printNode(javax.swing.tree.TreeNode node, String
indent)
    {
        System.out.println(indent+node);

        int n = node.getChildCount();
        for (int i=0; i<n; i++)
        {
            printNode(node.getChildAt(i), indent+"    ");
        }
    }
}
```

---

*[Table of Contents]*

# Chapter 4: The Tree Viewer



---

## 4.1 Overview

The Tree Viewer (or panel) shows the objects in each open file, and supports the navigation and editing of those objects. Multiple files can be viewed and edited, and both HDF4 and HDF5 files can be opened. The viewing and editing operations work for both HDF4 and HDF5, although some operations cannot be implemented for HDF4.

Every HDF5 object appears in at least one group. A set of objects can be stored together in a group. You can use the group object API functions to create and manipulate the groups. With the groups, you can organize the data objects in an HDF file. Objects in HDF4 may or may not belong to a group. An object that does not belong to any group is called a lone object, such as lone Vdata. To map HDF4 structure to HDF5 structure, a "dummy" root group is created in the tree view and lone objects are put at the root group.

Since objects can have names in more than one group, the set of all objects in an HDF file is a directed graph. It is difficult to manipulate and browse the graph on a 2-D virtual screen. Instead of showing the directed graph of the HDF file structure, HDFView displays HDF objects in a conventional structure as a tree of "folders," breaking the "loops" in the graph when necessary. The tree viewer of HDFView provides users an easy way to browse and manage HDF data objects in an HDF file. The following icons are used to represent HDF objects:

 --- HDF4 file
 --- HDF5 file
 --- Collapsed group folder
 --- Expanded group folder
 --- HDF4 SDS or HDF5 scalar dataset
 --- HDF4 Vdata or HDF5 compound dataset
 --- HDF4 SDS or HDF5 scalar dataset of type String

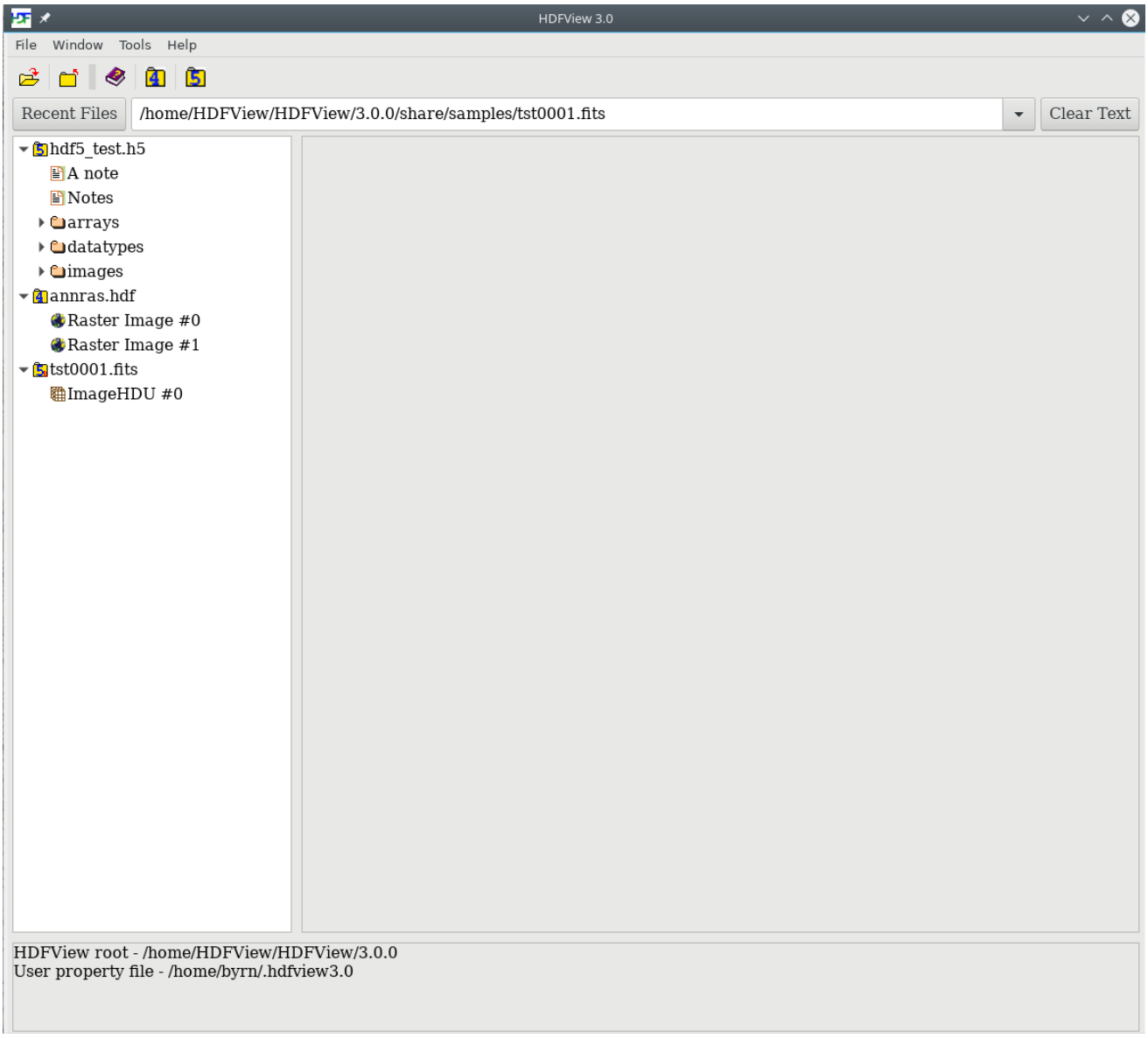
 --- HDF4 GR image or HDF5 image
 --- HDF5 committed datatype
 --- HDF5 object association

---

## 4.2 Tree Structure

Upon opening an HDF file, HDFView displays the tree structure of the HDF file in the tree viewer, which is the left panel in the HDFView window. Every HDF file has exactly one root group that is created when the file is created. The root group is added to the root of the tree. When two or more files are open, the root groups of the files are listed at the root in the tree view.

Groups are presented as folders in the tree viewer. Opening and closing folders, you can browse the individual HDF objects in the HDF file. Datasets, images and tables are leaf objects in the tree. Double clicking a leaf object allows you to see the content of the object displayed in a data/document window.

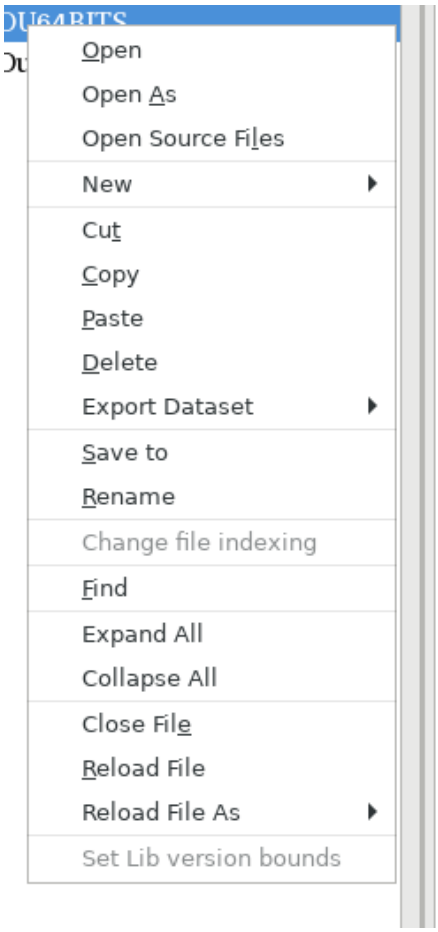



**The tree viewer**

# 4.3 View Data Content

The Tree View allows the user to navigate the structure of the HDF file, to select individual objects, and to view the content of the selected object.

You can open a data object in the tree by a double-mouse-click or selecting the “Open” command from the object's popup menu (accessed by right clicking on the object). By default, data content is displayed as an image, a table or text based on its datatype. See the following chapters for a detailed discussion of the data viewers.




**The object popup menu**

If the data value is not in memory, the “Open” action will load the data from the file and display it. If the data is already loaded into memory, it will just display the data. If you want to refresh data in memory or open a dataset with a different selection (subset or display option), you have to use the “Open As” command.

Using “Open As”, you can select a subset of the dataset to display or change the default display options. For example, you can display the data values of an image in a spreadsheet, or show a scalar dataset as an image. For multi-dimension datasets, you can select any of two dimensions to display. The mouse-drag navigator allows you to select a subset by dragging the mouse over the preview image. For more details about how to select a subset and dimensions, see 5.2 Subset and Dimension Selection.

The “Start” field(s) determines the starting coordinate of the selected area.

The “End” field(s) determines the ending coordinate of the selected area.

The “Stride” field(s) chooses array locations from the dataspace with each value in the stride array determining how many elements to move in each dimension. Setting a value in the stride array to 1 moves to each element in that dimension of the dataspace; setting a value of 2 in the

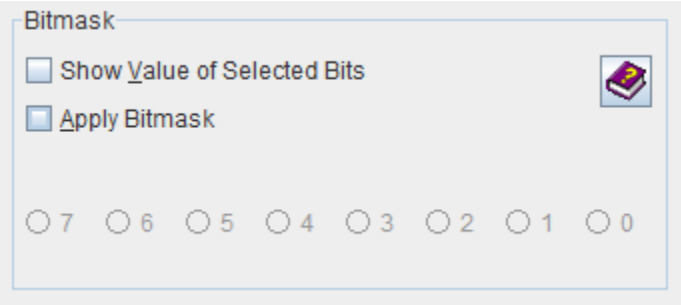
stride array moves to every other element in that dimension of the dataspace. In other words, the stride determines the number of elements to move from the start location in each dimension. Stride values of 0 are not allowed. If the stride parameter is NULL, a contiguous hyperslab is selected (as if each value in the stride array was set to all 1's).

### 4.3.1 Show Bit Values

HDFView allows users to view bit values of integers in two ways:

- applying a bitmask to data values
- showing values of selected bits

The figure below shows the GUI layout of the two options:




**The bitmask options**

Applying a bitmask means to use the bitwise operation 'AND' to the original data. For example,

```
    10010101 (data)
AND 00011100 (mask)
  = 00010100 (result)
```

Showing values of selected bits means to apply the bitmask to the data and throw away any unnecessary bits that are not selected by the bitmask, i.e. the '0' bits in the bitmask. In most of the cases, this will be the beginning and ending '0' bits in the bitmask. For example,

**Applying the mask:**

```
    10010101 (data)
AND 00011100 (mask)
  = 00010100 (result)
```

**Throwing away unnecessary bits:**

~~000~~101~~00~~ → 101 (decimal value 5)

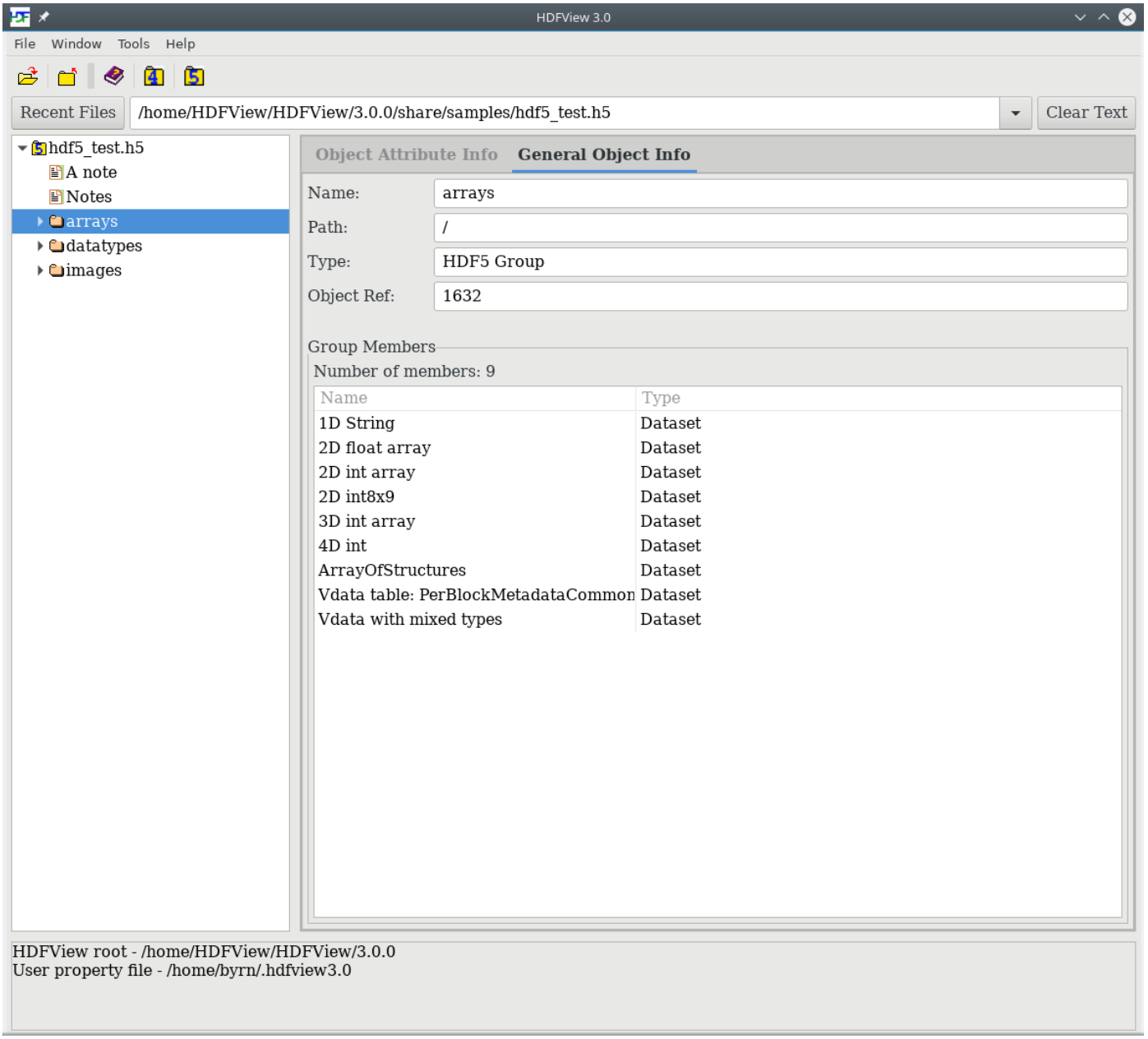
For the current implementation, only contiguous bits are allowed when the option of "Show Value of Selected Bits" is selected.

## 4.4 Display Metadata and Attributes

The metadata of an object appears by selecting the object (icon) in the tree. The metadata panel is the right panel in the HDFView window where the metadata content of objects are displayed in a tabbed view. The “Object Attribute Info” tab displays a list of attributes attached to the selected object. The attribute name, type, size and a short sample of data is displayed. Double-clicking on an attribute line will open a Data window for the selected attribute. There is an “Add Attribute” button to create a new attribute attached to the currently selected object in the treeview. And a “Delete Attribute” button to remove the selected attribute.
The “General Object Info” tab displays object information such as names of objects, member names and types, or paths, type and version bounds for file types.

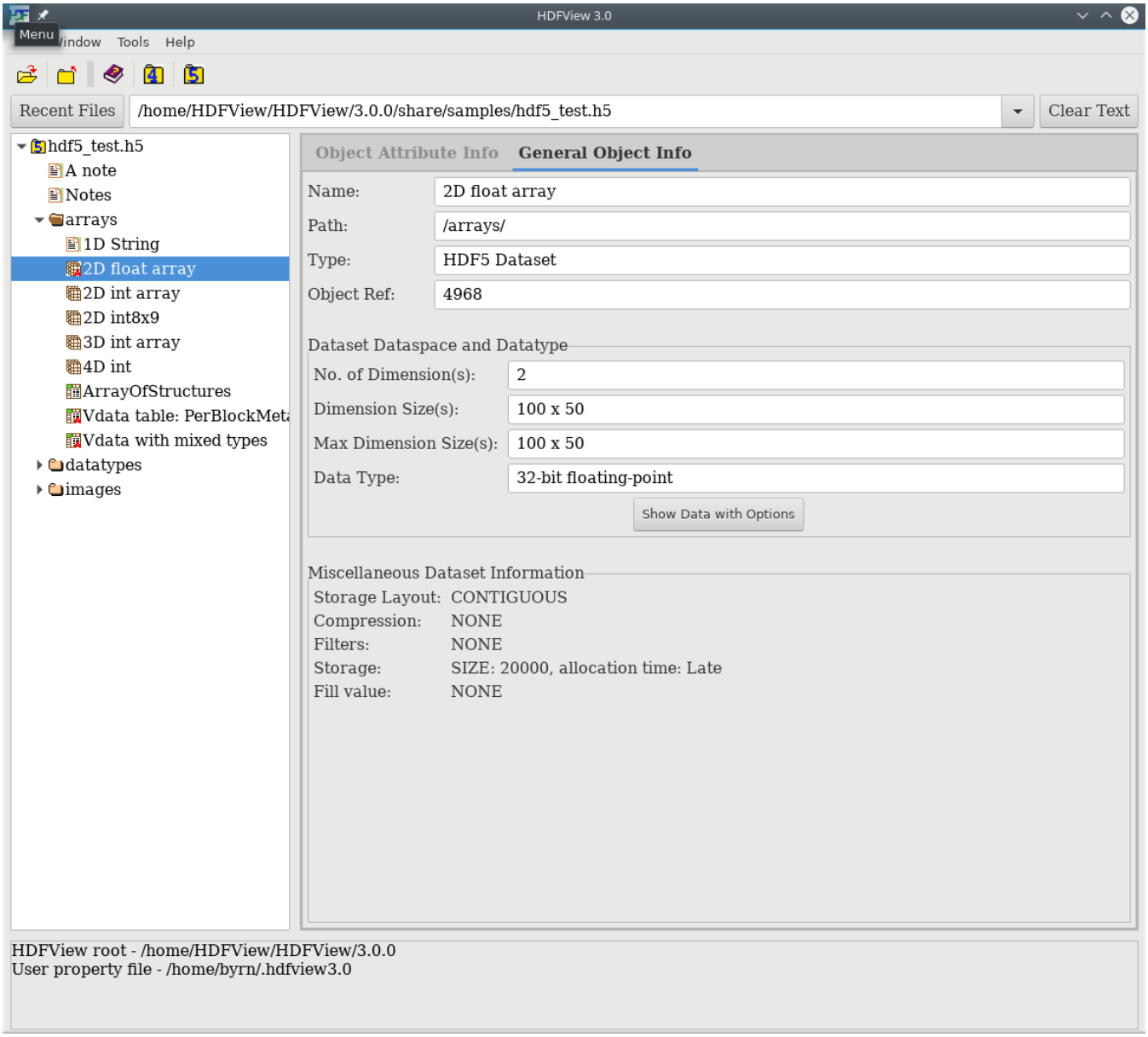
General metadata includes the name, type, and path of the data object.

For groups, general metadata also includes the name, type, and size of all the members of the group. The following figure shows an example of the general information of a group.



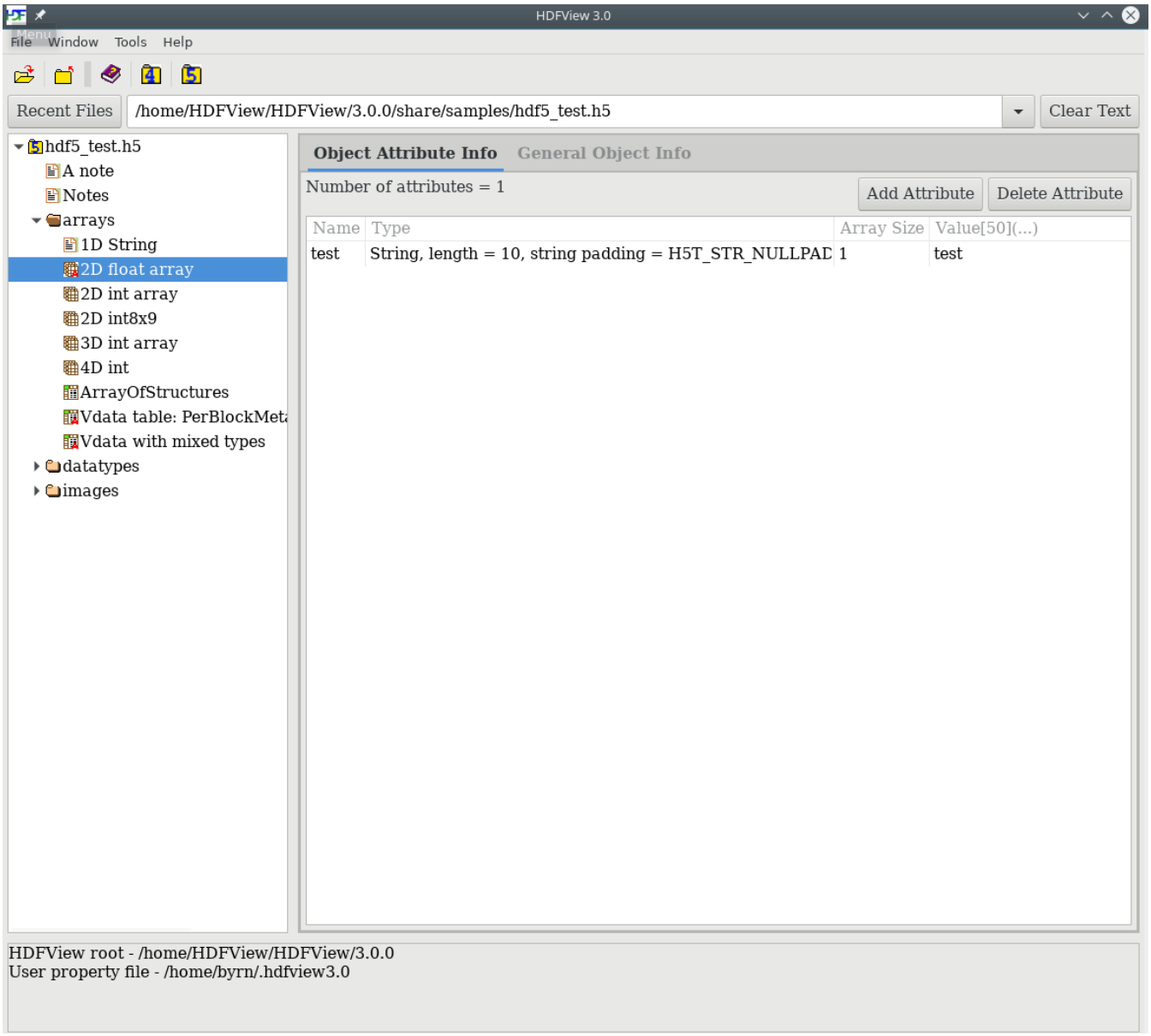

**General properties of an HDF group**

For a dataset (or image), the general metadata also includes the dataspace information (rank, current, and maximum size of each dimension) and datatype information (type, size, and order). The following figure shows an example of the metadata for a dataset.




**General properties of a dataset**

To see the attributes of an object, click the “Object Attribute Info” tab in the object's metadata panel. The name, type, size, and value of all attributes of the selected object are displayed in a table. The following figure shows an example of the attribute display.



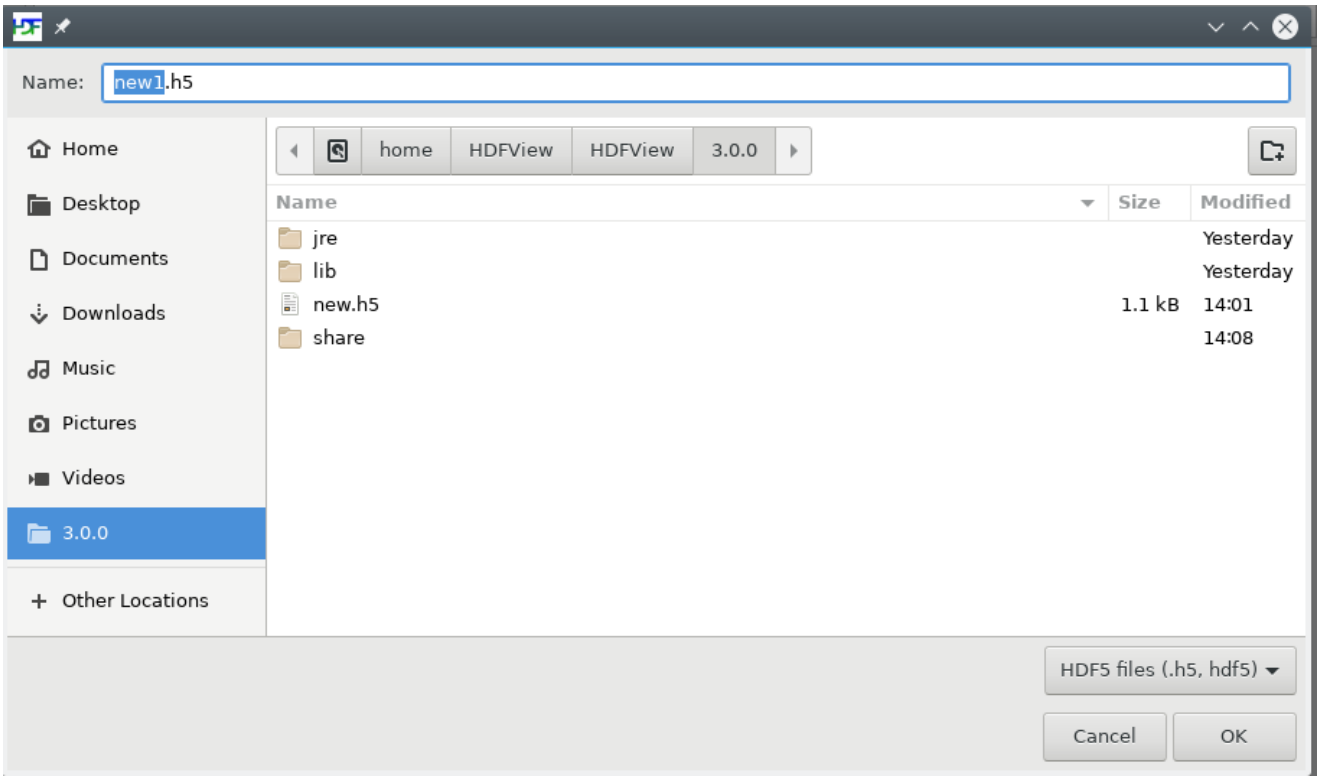

**The Attributes tab**

# 4.5 Edit File and File Structure

HDFView allows you to create and save files and to add and delete objects in the files. **Users should be aware that all the changes are made to the actual stored file. This version does not support the undo operation; editing results cannot be automatically recovered**.

### 4.5.1 Create and Save File

The file creation dialog box appears when you choose the “New -> HDF4/5” command from the File menu. Choose the path and the file name you want to create and click the “Save” button from the New File dialog box. A new file will be created with an empty root group.



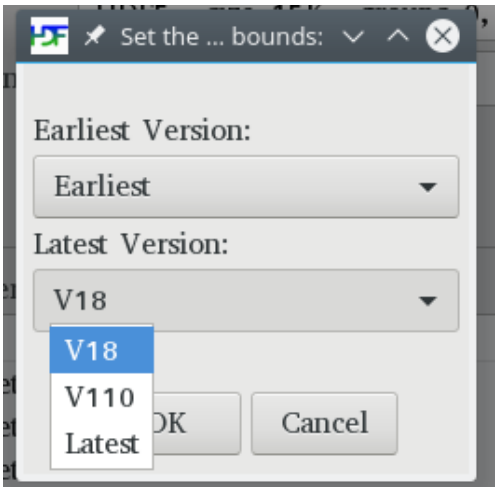

**Create New File**

Select “Save As” from the File menu to save the currently selected file into a new file of the same format, i.e. HDF4 to HDF4 or HDF5 to HDF5. (HDFView does not support HDF4 to HDF5 conversion.)

For HDF5, a new file is written that does not contain the inaccessible objects, and it packs the unusable space. Thus, the new file may be smaller than the original. **Dataset or attribute values that are object references cannot be updated.** For HDF4, a new file is an exact copy with the same file content and file size.

### 4.5.2 Setting the Library Version Bounds

The library version bounds of an HDF5 file can be set by right clicking the file from treeview, then selecting “Set Lib version bounds” from the Context menu and setting the earliest version and latest version. In order to see the currently set library versions of an HDF5 File, select the “General Object Info” tab of the metadata panel and it will display the library versions that had been set earlier.



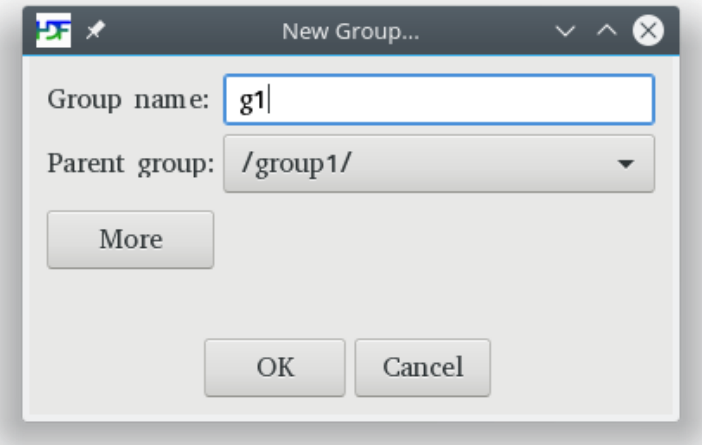

**Setting the library version bounds of an HDF5 file**

### 4.5.3 Add and Delete Object

All objects in HDFView are created by selecting the "New" command from the Context menu, accessed by right clicking on any object in the TreeView panel.

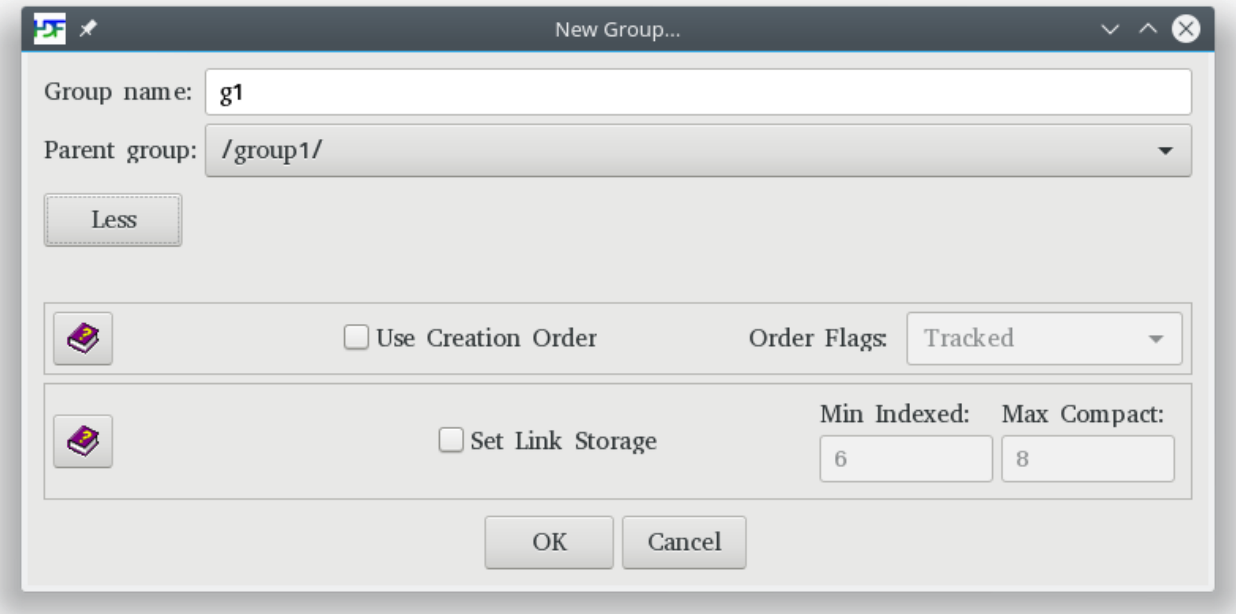
#### *4.5.3.1 Create a New Group*

To create a new group, select the “New -> Group” command from the Context menu, then give the name of the new group and the parent group to which the new group is to be added.




**The New Group dialog box**

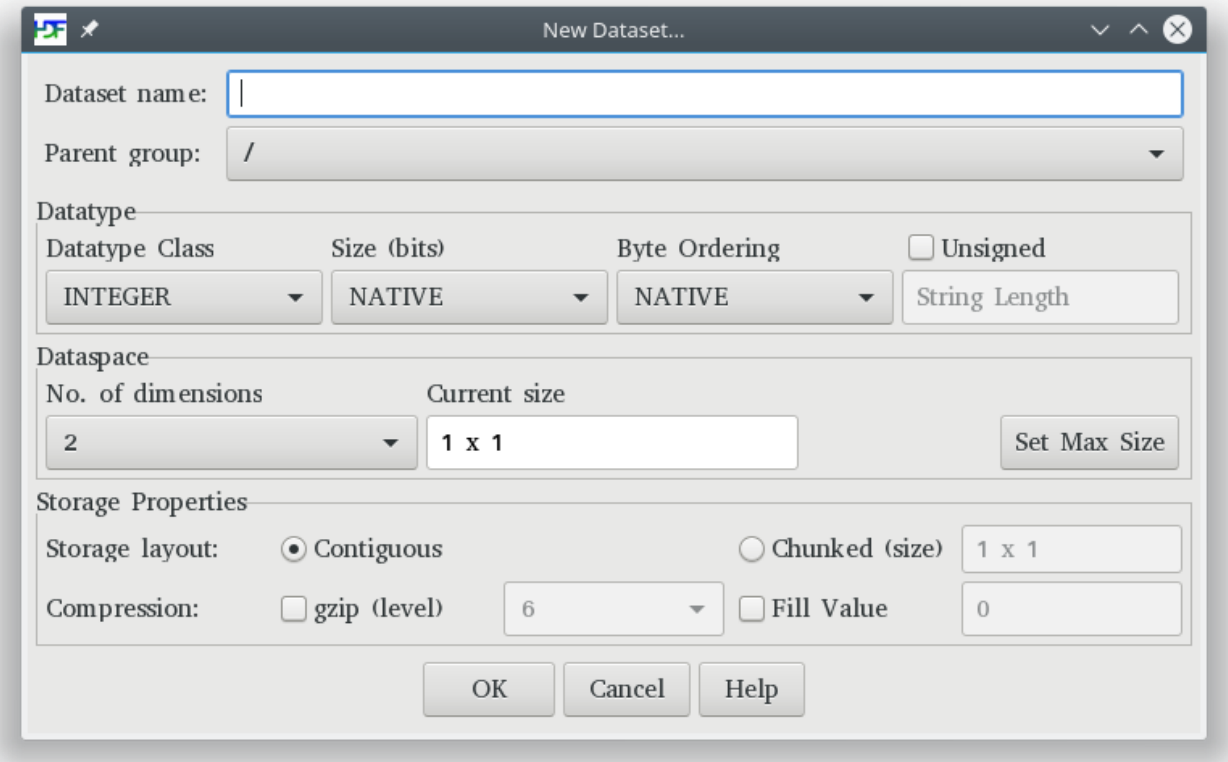
By clicking on the “more” button, the user can set the options, creation order, and link storage when creating groups. Help buttons are provided for more information about the options.




**Create new group options**

#### *4.5.3.2 Create a New Dataset*

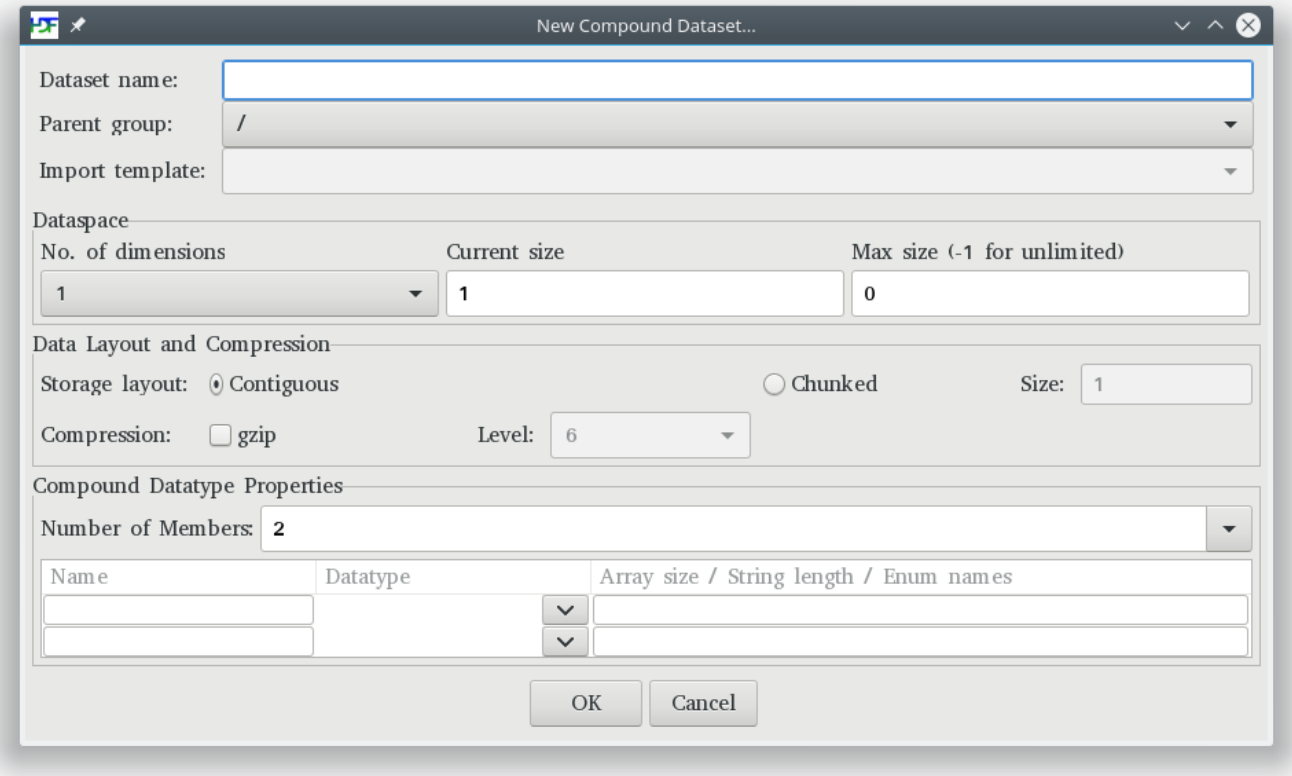
To create a new dataset, select the “New -> Dataset” command from the HDF5 Context menu, or “New -> SDS” from the HDF4 Context menu, then specify the name, path, datatype, dataspace, and storage layout and compression of the new dataset. The current version only supports creating simple datatypes. This version cannot create an HDF4 VData; however, these types of objects can be viewed. Supported datatypes include integers (byte, short, int, long), float, double, and character. For HDF5, it also includes String and Object references.




**The New Dataset dialog box**

### *4.5.3.3 Create a New Compound Dataset*

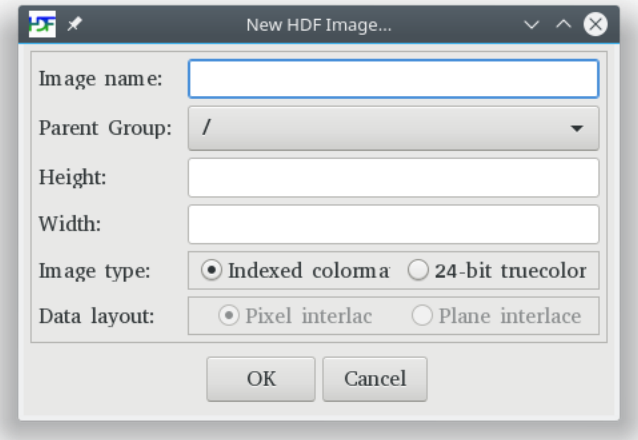
To create a new compound dataset (HDF5 only), select the "New -> Compound Dataset" command from the Context menu. Then, specify the name, path and template to use (if applicable), as well as the dataspace, storage layout and compression and properties of the new dataset. Supported datatypes include integers (byte, short, int, long), float, double, String and Object reference.




**The New Compound Dataset dialog box**

### *4.5.3.4 Create a New Image*

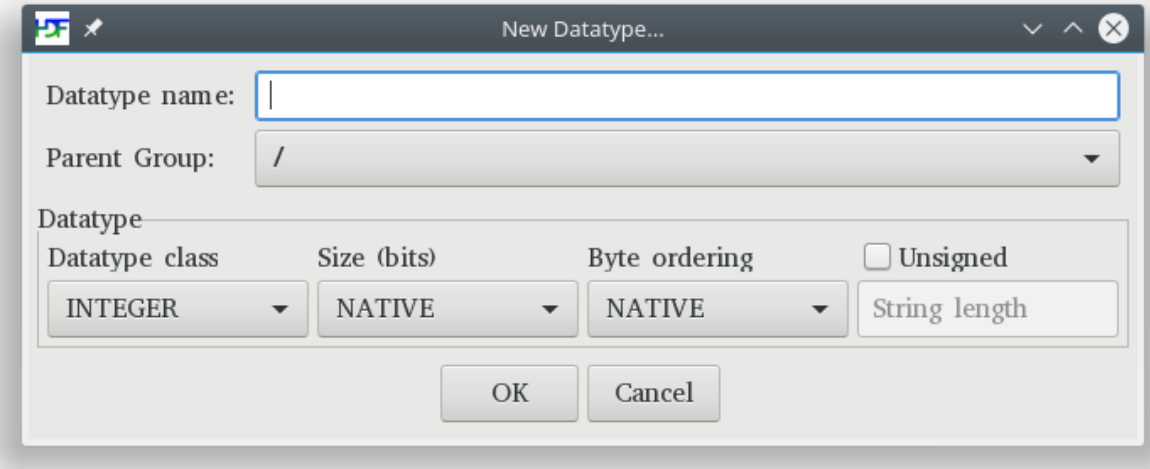
The “New -> Image” command from the Context menu allows you to create an empty image with default image attributes. The new image does not have any palette attached to it. You can create two kinds of images: an indexed image with 256 colors and a 24-bit true color image.




**The New HDF Image dialog box**

### *4.5.3.5 Create a New Committed Datatype*

The "New -> Datatype" command from the Context menu allows you to create a new committed datatype (HDF5 only). Committed datatypes are able to be shared among datasets within an HDF5 file.

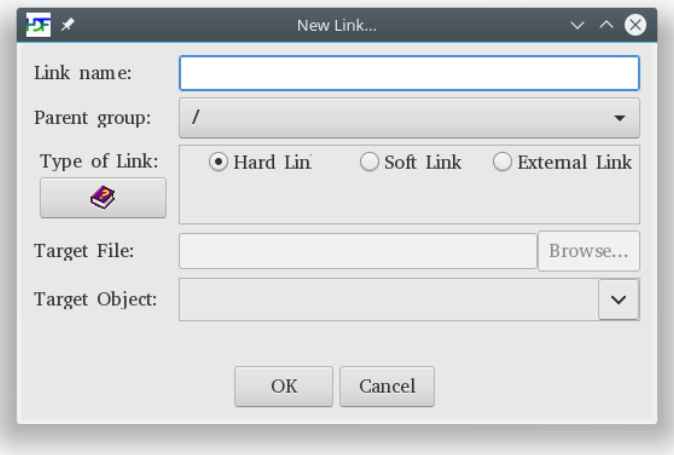



**The New Datatype dialog box**

### *4.5.3.6 Create a New Link*

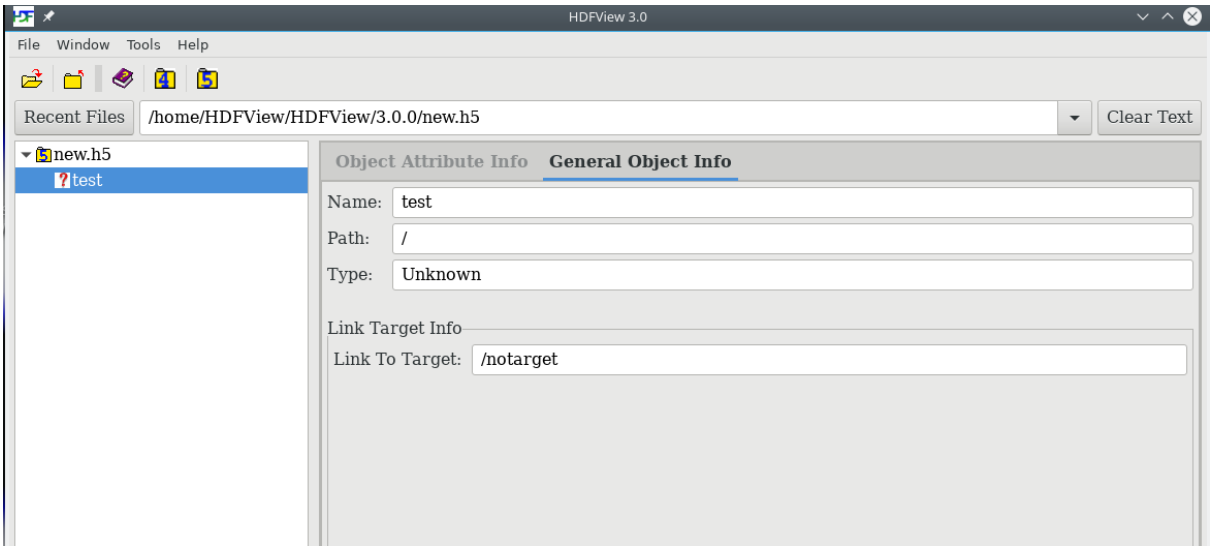
The "New -> Link" command from the Context menu allows you to create links to objects (HDF5 only). Three types of links can be created: hard, soft, or external. Hard links and soft

links are links to objects in the same HDF5 file. External links are links to objects in a different HDF5 file. Soft links and external links are symbolic links and are allowed to dangle, meaning that the target object need not exist at the time the link is created. A Help button is provided for more information about links.




**The New Link dialog box**

The type for a soft link or external link object that links to a non-existing target object is unknown, so it will have a different icon: ‘ ’ The user can change the target object being linked to by selecting the link object from the tree view and select the “General Object Info” tab in the Metadata panel. In the Properties dialog box, edit the “Link to Target” field to change the target object.




Change the target object for a dangling link

#### *4.5.3.7 Delete an Object*

To delete objects from the tree, you first select the objects, then choose the "Delete" command from the Context menu. A confirmation dialog appears to verify that you want to delete the selected objects. **Deleting an object from an HDF4 file is not supported**.

Deleting an HDF5 object calls the H5Gunlink() function of the HDF5 library. The unlink function removes the link, and when an object has no remaining links, it can be deleted. However, the HDF5 library does not currently reclaim the space in the file. This means that the size of the file does not shrink even though objects are removed. (Users are cautioned that repeatedly creating and deleting objects with HDFView can cause the HDF5 file to grow.) To get rid of the inaccessible objects, you have to save the current file into a new file and rename the new file to the current file.

### 4.5.4 Copy and Paste Object

HDFView allows you to copy an object and paste it in the same file or a different file of the same type. To copy and paste an HDF5 object from the tree, select the object and then select the "Copy" command from the Context menu. You then select a group by right clicking on the group to which the object is to be pasted. After you select the "Paste" command, the object is copied into the selected group. HDFView does not allow you to copy/paste an object of different format. The "Paste" command will create a new object in the selected group instead of making a hard link.

You can also select multiple objects in the tree and apply the copy/paste action to the selected objects. Selection can be contiguous and discontiguous. To make a contiguous selection, click on the first object, then click on the second object while holding down the Shift key. All the objects between the two objects will be selected inclusively. To make a discontiguous selection, hold down the Ctrl key while clicking the objects with the left mouse button. The objects being clicked will be selected. The rest of the operation of copying/pasting multiple objects is the same as that of copying/pasting a single object.

Copying a group will also copy all of the descendents of the group. Copying a root group is not allowed.

### 4.5.5 Move Object

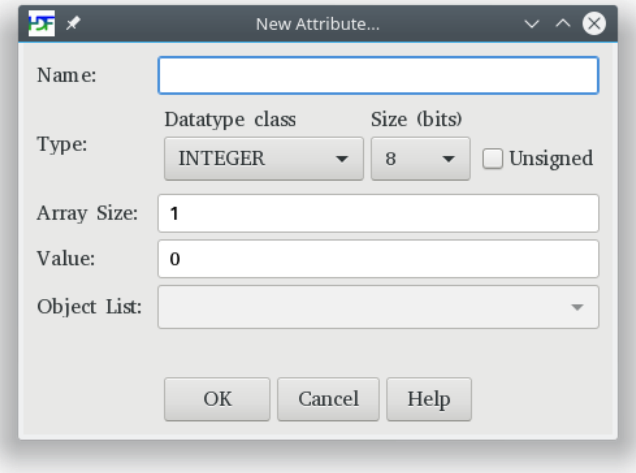
HDFView allows you to move an object within the same HDF5 file from one location to another. Moving objects from one file to another file is not supported by HDFView. To move an HDF5 object from the tree, right click the object and then select the "Cut" command from the Context menu. You then select a group by a right click on the group where the object is to be moved. Select the "Paste" command, the object is then moved into the selected group.

You can also select multiple objects in the tree and apply the move action to the selected objects. Selection can be contiguous and discontiguous.

Moving a group will also move all the descendents of the group. Moving a root group is not allowed. **Moving objects is not supported by HDF4 Files**.

### 4.5.6 Add, Delete and Modify Attribute

To add an attribute, click the "Add Attribute" button on the "Object Attribute Info" tab in the object's metadata panel. You can create an attribute of a string, scalar, and 1D array of simple datatypes. Attributes of multiple dimensions or a compound datatype are not supported.




**The New Attribute dialog box**

To change the value of an attribute, type the new value into the cell. Only string and scalar attributes can be modified. The new value must be interpretable as a value of the correct type for the attribute. For a string value, the value typed will be truncated to the size of the attribute if it is longer than the declared size of the attribute. A valid data value must follow the rules specified by the section 5.4 Change Data Value.

To change the name of an attribute, right-click the attribute in the "Object Attribute Info" tab in the object's metadata panel. Note: once the attribute name is changed, it cannot be changed again until the file is closed and reopened.

To delete an attribute, select the attribute and click the "Delete" button in the "Object Attribute Info" tab in the object's metadata panel. **Deleting an attribute of an HDF4 object is not supported**.

---

*[Table of Contents]*

# Chapter 5: Table Viewer

HDFView displays datasets in a two-dimensional table, the TableView. The TableView allows you to view and change the values of an image's dataset. You can select rows and columns and plot the row/column data in a line plot. The current version of HDFView does not allow you to change data values of an HDF4 Vdata.

If a dataset has three or more dimensions you can only view two dimensions at a time. Using the Dataset Selection Dialog box, you may select any two dimensions of the dataset to display and a third dimension to flip the two-dimension table along that dimension.

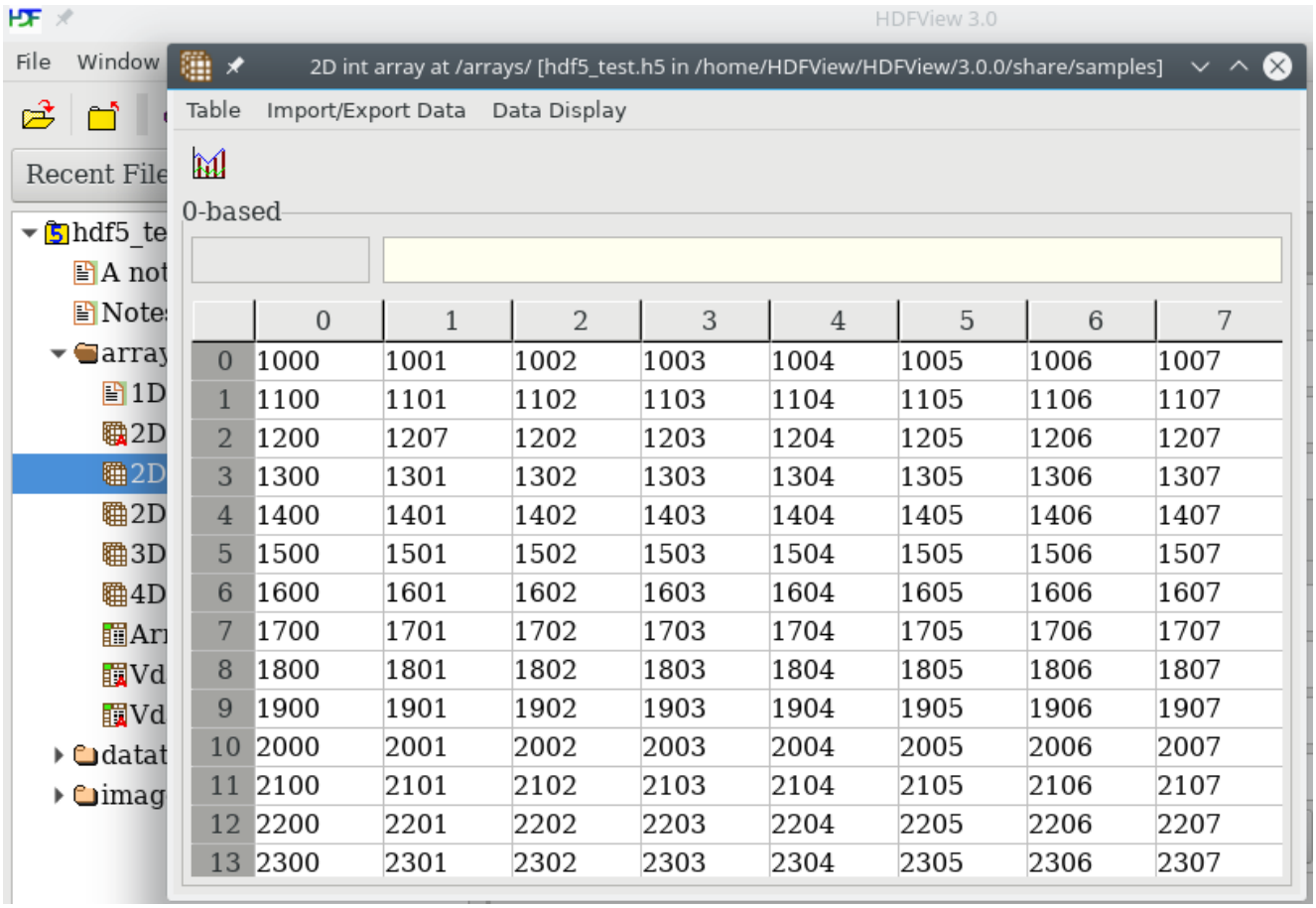


---

## 5.1 Open Dataset

To open the entire contents of a dataset, double-click on the dataset or select the dataset, then choose the “Open” command from the Context menu. A new spreadsheet window is created.

HDFView displays numerical datasets in a “spreadsheet”, which shows the data values in a grid. A one-dimensional dataset is displayed as a single column and a number of rows of dimension size. A two-dimensional dataset is displayed as a number of columns of the first dimension size and a number of rows of the second dimension size, i.e. dim[0]=height and dim[1]=width by default. You can change the order of the dimensions using the “Open As” command.



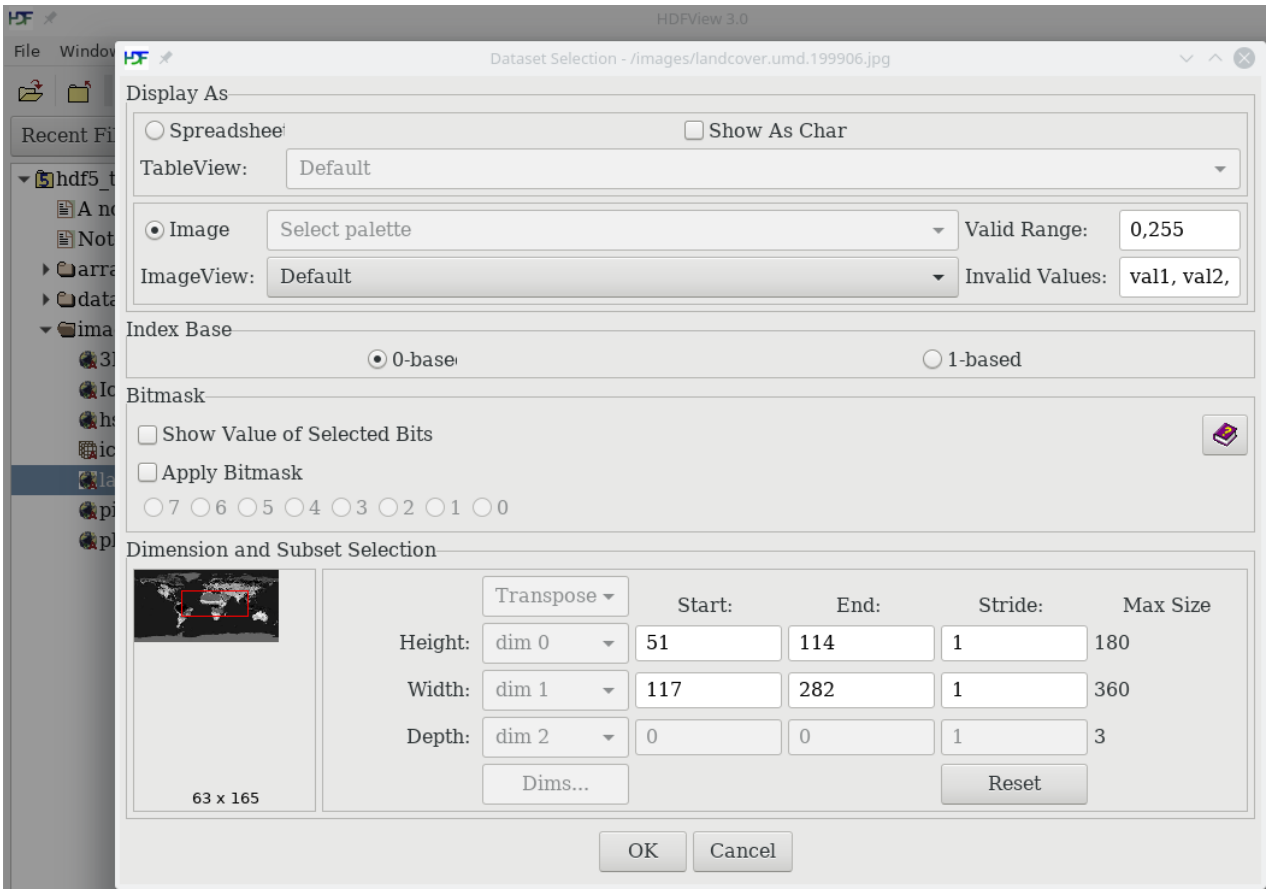

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 1000 | 1001 | 1002 | 1003 | 1004 | 1005 | 1006 | 1007 |
| 1 | 1100 | 1101 | 1102 | 1103 | 1104 | 1105 | 1106 | 1107 |
| 2 | 1200 | 1207 | 1202 | 1203 | 1204 | 1205 | 1206 | 1207 |
| 3 | 1300 | 1301 | 1302 | 1303 | 1304 | 1305 | 1306 | 1307 |
| 4 | 1400 | 1401 | 1402 | 1403 | 1404 | 1405 | 1406 | 1407 |
| 5 | 1500 | 1501 | 1502 | 1503 | 1504 | 1505 | 1506 | 1507 |
| 6 | 1600 | 1601 | 1602 | 1603 | 1604 | 1605 | 1606 | 1607 |
| 7 | 1700 | 1701 | 1702 | 1703 | 1704 | 1705 | 1706 | 1707 |
| 8 | 1800 | 1801 | 1802 | 1803 | 1804 | 1805 | 1806 | 1807 |
| 9 | 1900 | 1901 | 1902 | 1903 | 1904 | 1905 | 1906 | 1907 |
| 10 | 2000 | 2001 | 2002 | 2003 | 2004 | 2005 | 2006 | 2007 |
| 11 | 2100 | 2101 | 2102 | 2103 | 2104 | 2105 | 2106 | 2107 |
| 12 | 2200 | 2201 | 2202 | 2203 | 2204 | 2205 | 2206 | 2207 |
| 13 | 2300 | 2301 | 2302 | 2303 | 2304 | 2305 | 2306 | 2307 |

**Spreadsheet with a 2-D dataset**

# 5.2 Subset and Dimension Selection

Opening an entire large dataset may cause an ‘Out Of Memory Error’ because the Java Virtual Machine cannot create the required objects. HDFView provides options to select a subset of a dataset for display. You can also select dimensions and the order of dimensions to display, e.g., to switch the columns and rows.

To make a selection, select a dataset from the tree and choose the “Open As” command from the Context menu. The selection dialog box appears. You can make a selection by dragging the mouse on the preview image or entering the values of start, end, and stride. The figure below shows that a subset of size 66 x 159 is selected from a true color image with the size of 179 x 359.

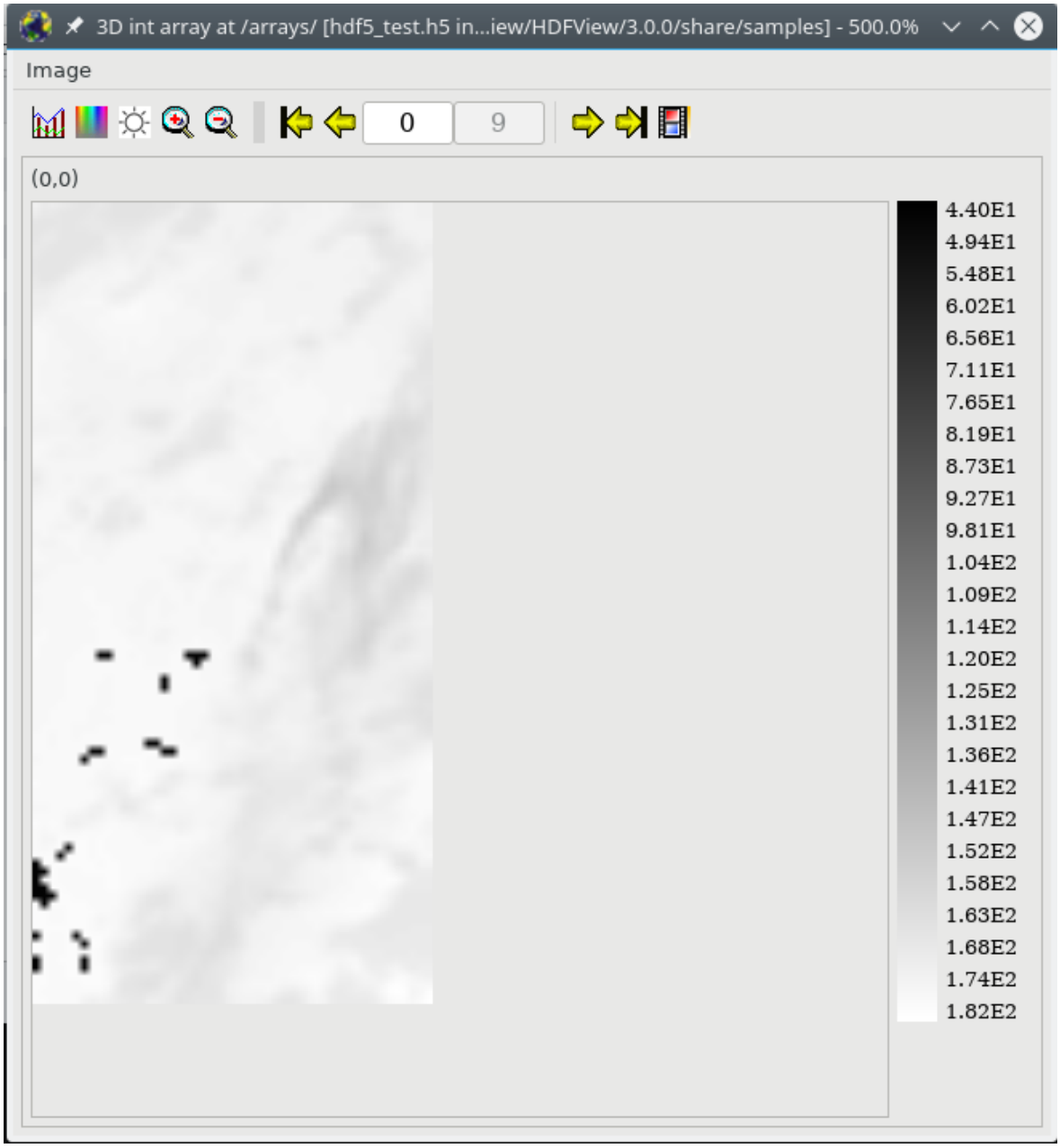



**The Dataset Selection dialog box**

By default, a scalar dataset (e.g., a dataset or SDS of numbers) is displayed in a spreadsheet. You can also display a dataset as an image. To display a dataset as an image, click the Image radio button in the Dataset Selection dialog box and select a predefined color table for the dataset. This operation takes the data values of the dataset as values of an indexed array, i.e., as indices into a palette. The default palette will be used to create the image from the dataset if it does not have an attached palette. If the data values are not integers or have a range outside 0 to 255: they are binned into 256 equally spaced intervals.

### 5.2.1 Setting Valid Values

The "Open As" option allows you to set a range of valid data values, as well as setting specific values to be considered invalid, when displaying a dataset as an image (see dataset selection dialog figure above). By setting these, any pixels (data points) whose values are outside the valid range will not be shown and pixels whose value has been set as an invalid value will be mapped to 0. As an example, the figure below shows a dataset with the value '55' set to be invalid:




### 5.2.2 Dimension Size

A subset is determined by the start and end locations, and the stride. The “Start” array determines the starting coordinates of the subset to select. The “End” array determines the ending coordinates of the subset to select. The “Stride” array chooses array locations from the dataspace with each value in the stride array determining how many elements to move in each dimension. Setting a value in the stride array to 1 moves to each element in that dimension of the dataspace; setting a value of 2 in a location in the stride array moves to every other element in that dimension of the dataspace. In other words, the stride determines the number of elements to move from the start location in each dimension.

HDFView uses a common-sense indexing scheme for selecting rows and columns. If the user wants a subset that begins at i,j the START coordinates will start at j, j instead of i-1, j-1.

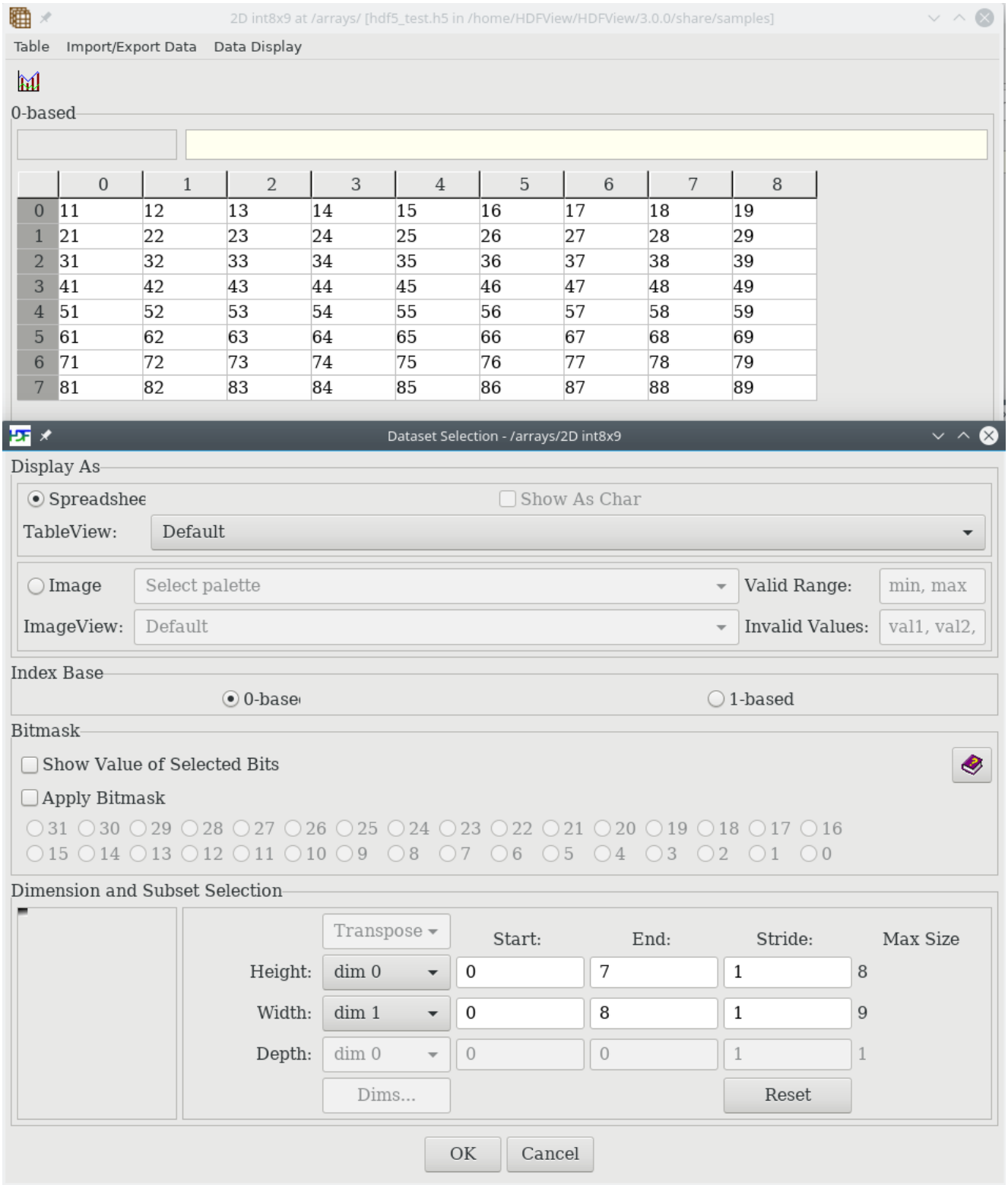
An example of a 2-D integer dataset of size 8 X 9 is shown below.

**2-D int array of size 8X9**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 |
| 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 |
| 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 |
| 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 |
| 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 |

The following are a few examples of subsets of the 2-D int array.

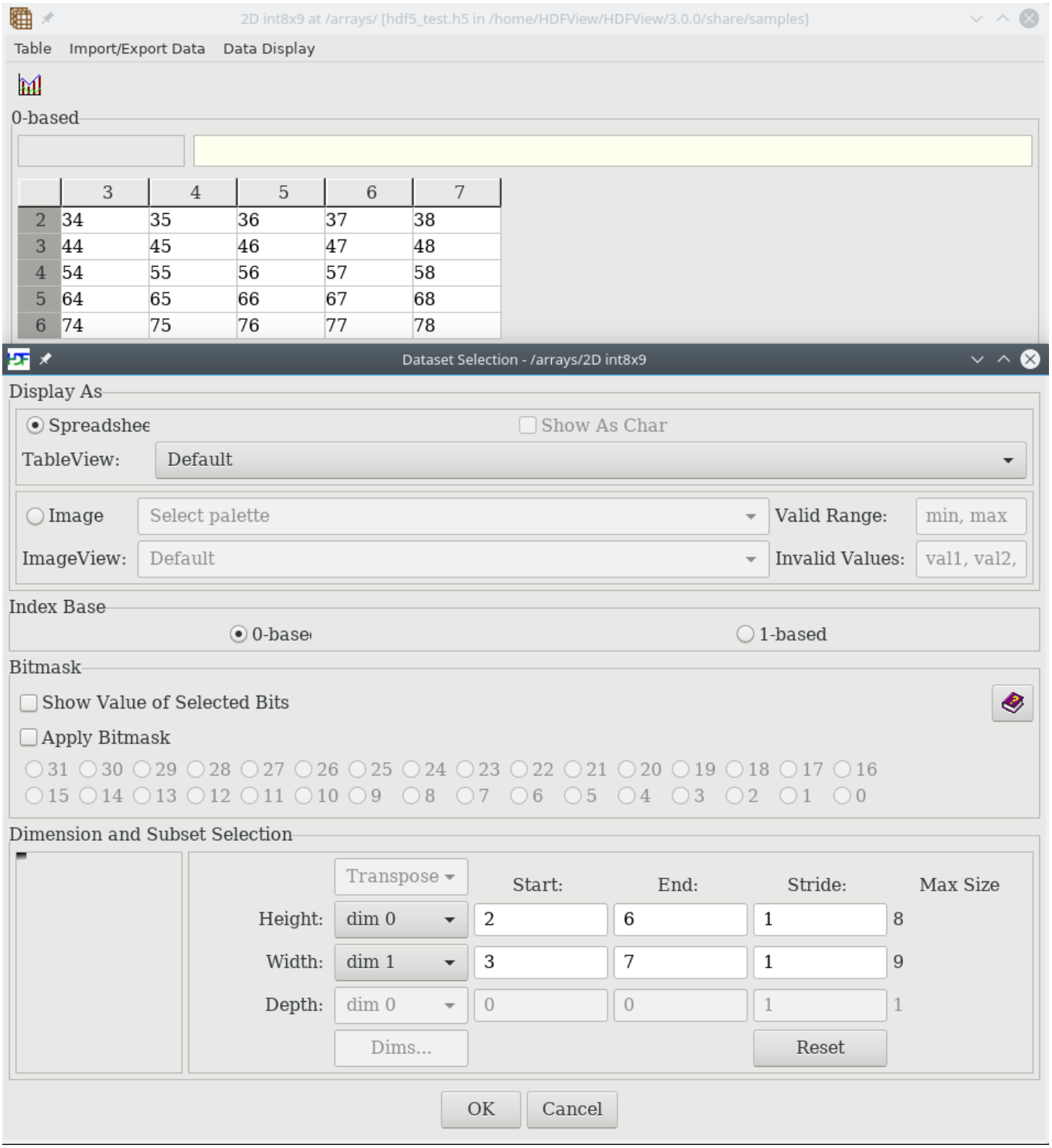
### *5.2.2.1 Example of reading an entire dataset -- start=(0, 0), end=(7, 8) and stride=(1, 1)*




| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 1 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 2 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| 3 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 |
| 4 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 |
| 5 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 |
| 6 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 |
| 7 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 |

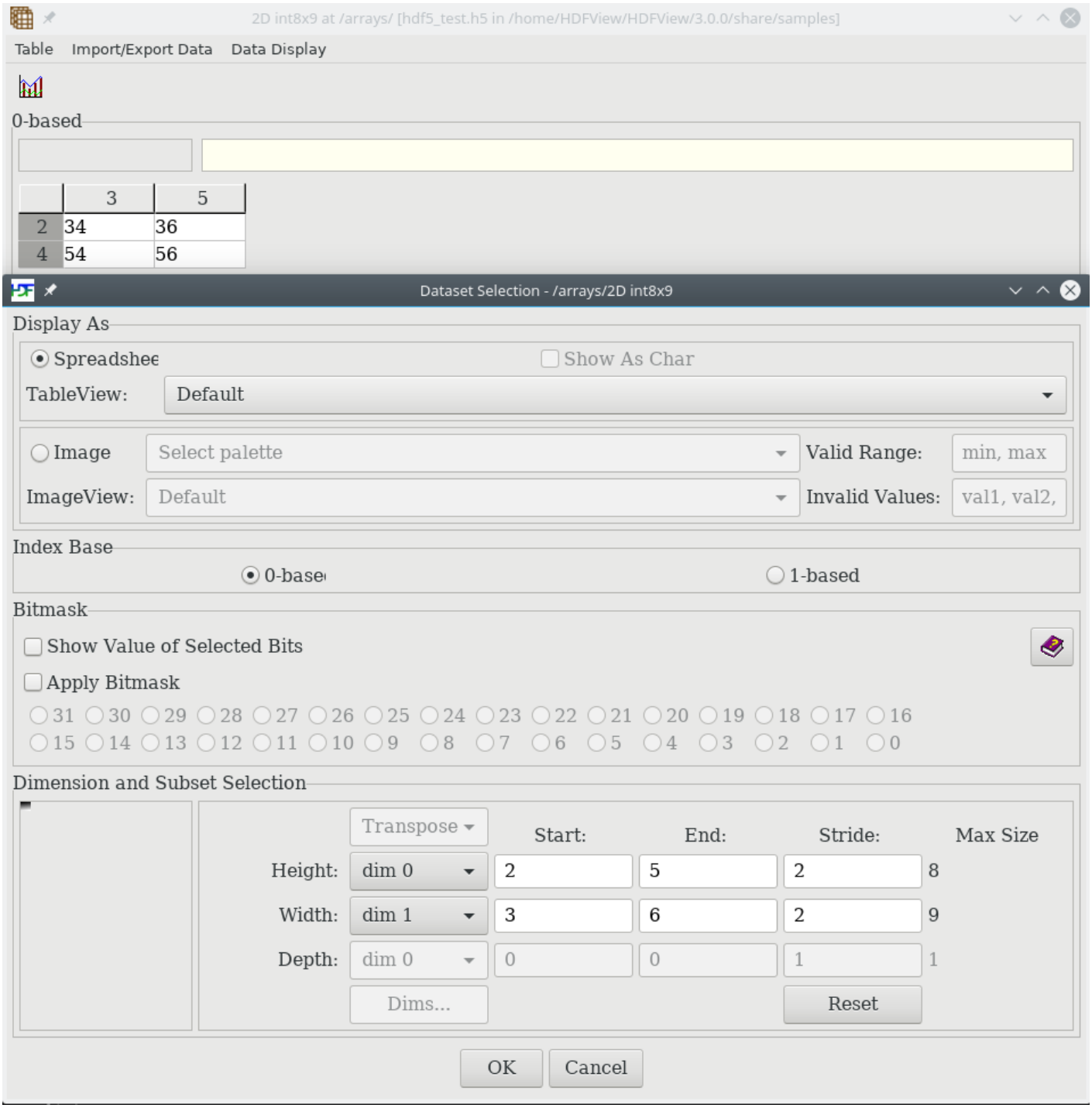
**The whole dataset -- start=(0, 0), end=(7, 8) and stride=(1, 1)**

### *5.2.2.2 Example of Reading Subset -- start=(2, 3), end=(6, 7) and stride=(1, 1)*




**Subset -- start=(2, 3), end=(6, 7) and stride=(1, 1)**

### *5.2.2.3 Example of Reading Subset with stride -- start=(2, 3), end=(5, 6) and stride=(2, 2)*



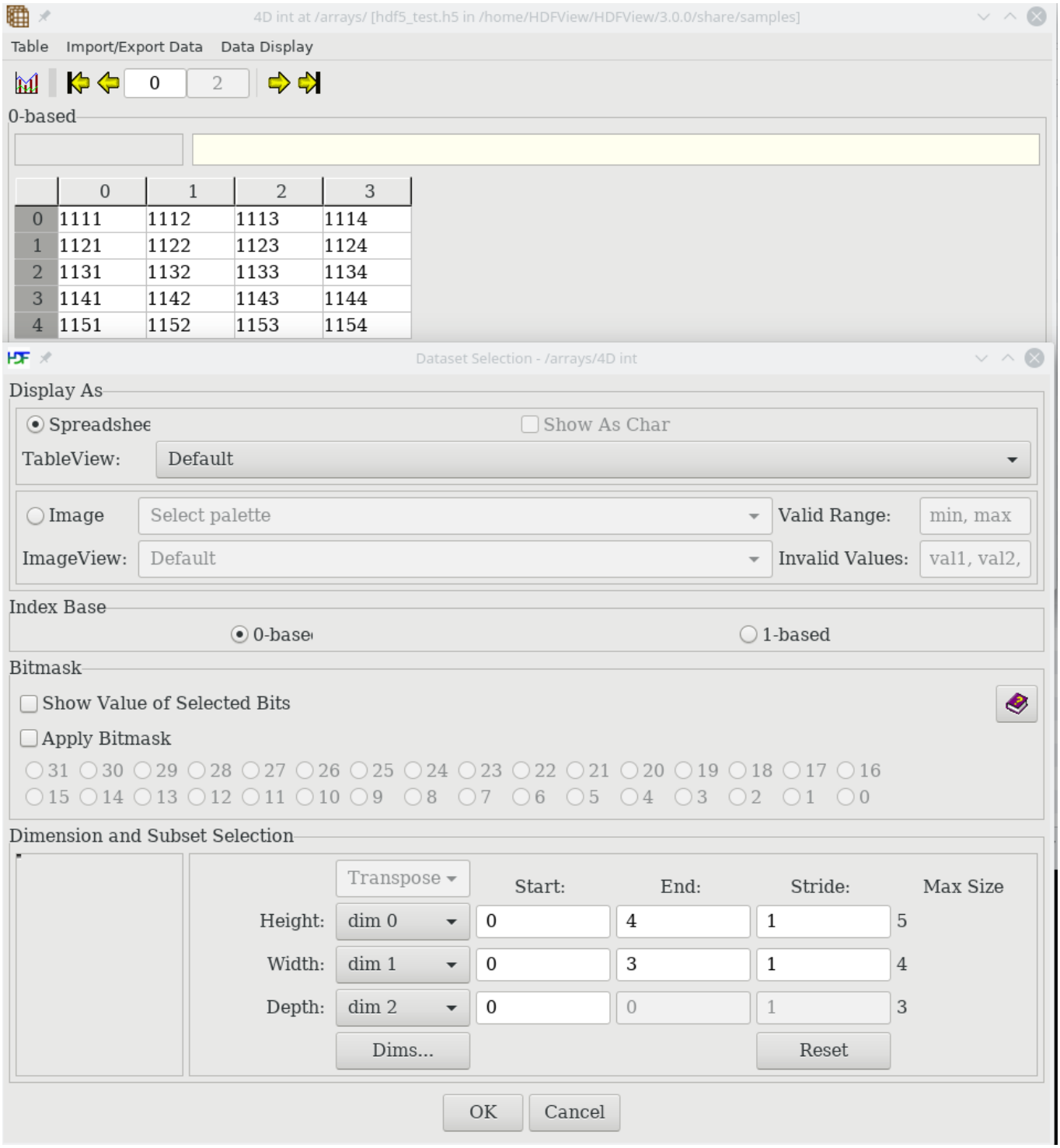

**Subset with stride -- start=(2, 3), end=(5, 6) and stride=(2, 2)**

### 5.2.3 Three or More Dimensions

For a three or more dimensional dataset, the first two dimensions are displayed as a 2-D spreadsheet, and the third dimension is chosen as the page number of the 2-D spreadsheet, i.e. dim[0]=height, dim[1]=width and dim[2]=depth by default. Using the "Open As" option to change the default order of dimensions, the user can flip the page forward or backward to look at the 2D data values at different positions along the third dimension. The current page number is displayed in the status bar of the HDFView.

To flip a data sheet of a 3-D dataset, use the "First", "Previous", "Next" or "Last" command on the tool bar.

The following figure shows how a 4-D integer dataset of size 5 x 4 x 3 x 2 is displayed. The data is displayed in a spreadsheet of 5 X 4 (dim0 by dim1), of page 2 (dim2), cutting locations 2 at dim3.




**Spreadsheet with 4-D dataset**

### 5.2.4 Swap Dimension and Data Transpose

By default, HDFView chooses the first coordinate, dim[0], as the ROW index and the second coordinate, dim[1], as the COLUMN index. For example, a 2-D dataset of 8 X 9 (dim0=8, dim1=9) is displayed as eight rows and nine columns by default.

You can also swap the dimension order. However, swapping the dimension order does not change the data order. To change the data order, use the "Transpose" option. Swapping and transposing only apply to the data in the display and not to the data in the file.

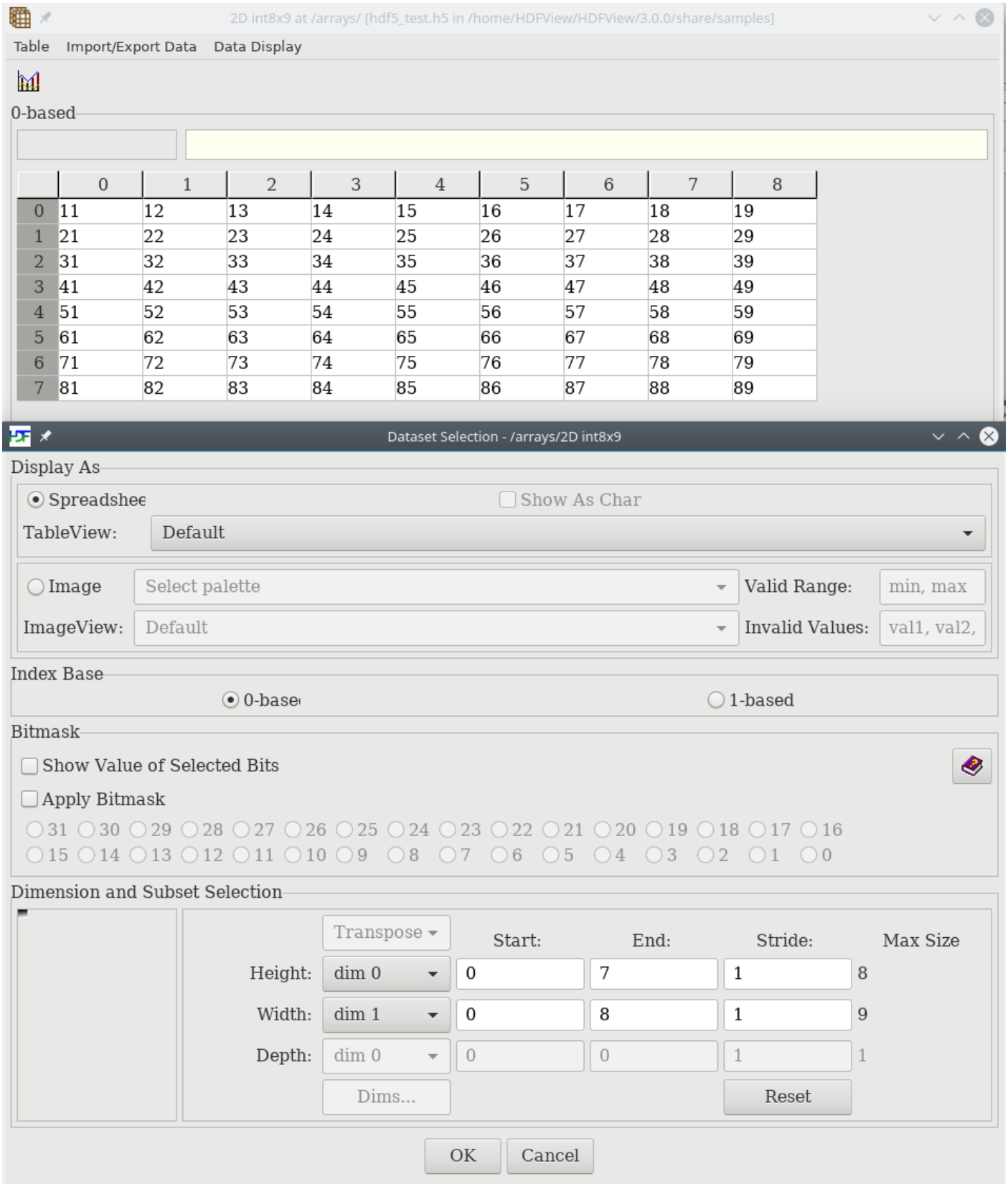
Let us use the previous example to demonstrate this procedure.

**2-D int array of size 8X9**

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 |
| 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 |
| 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 |
| 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 |
| 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 |

#### *5.2.4.1 Example of Default dimension order -- row index = dim[0], column index = dim[1]*

By default, the dataset is displayed as eight rows and nine columns.

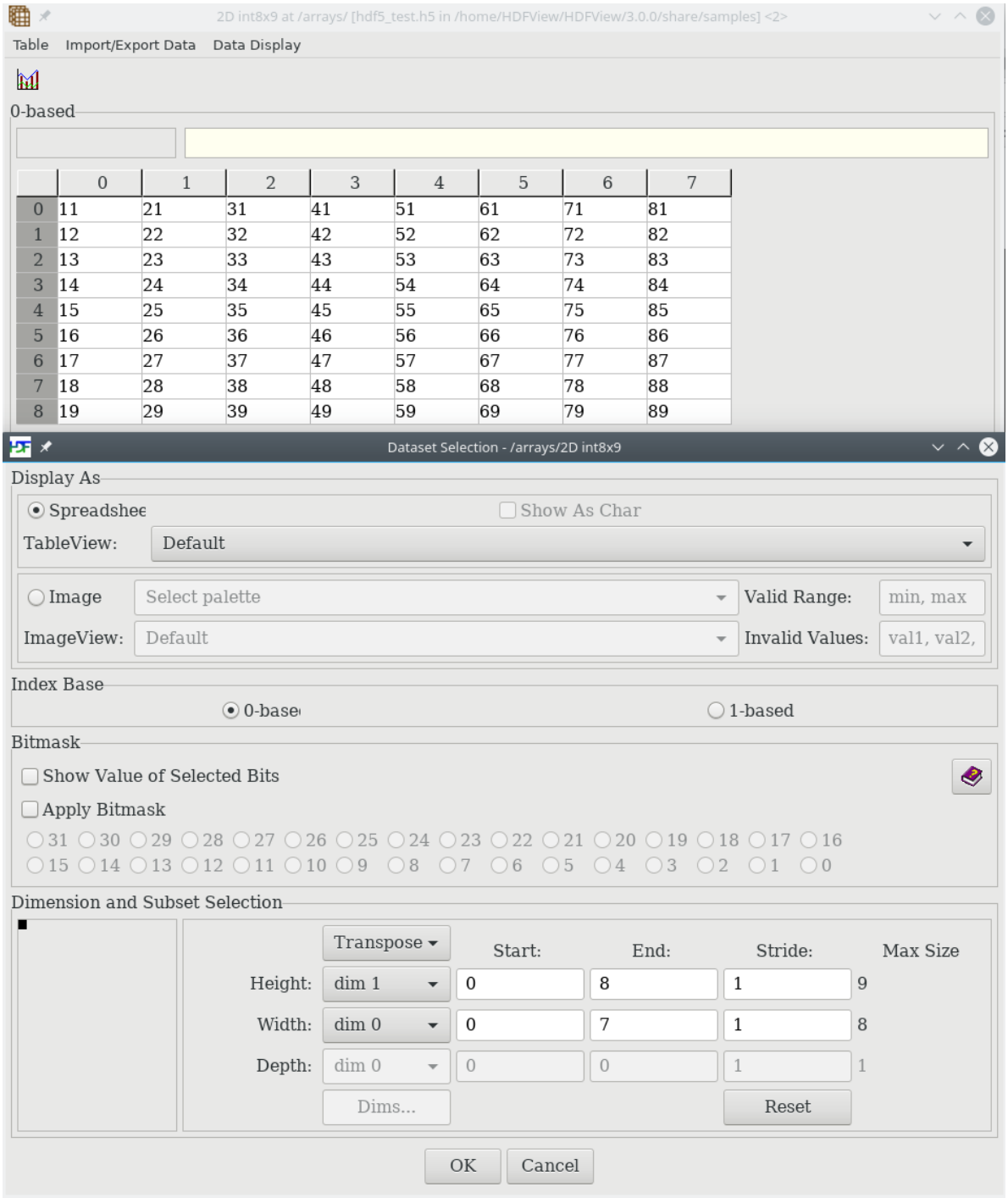



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| 1 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 |
| 2 | 31 | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 |
| 3 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 |
| 4 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 |
| 5 | 61 | 62 | 63 | 64 | 65 | 66 | 67 | 68 | 69 |
| 6 | 71 | 72 | 73 | 74 | 75 | 76 | 77 | 78 | 79 |
| 7 | 81 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 |

**Default dimension order -- row index = dim[0], column index = dim[1]**

### *5.2.4.2 Example of Swap row/column dimensions -- row index = dim[1], column index = dim[0]*

Swap row and column dimensions to display the dataset as nine rows and eight columns and transpose the data (changing the data order) when we swap the row and column dimensions. To transpose the data, select the “Transpose” option from the drop-down menu. For example, transpose the data to display the dataset as nine rows and eight columns.



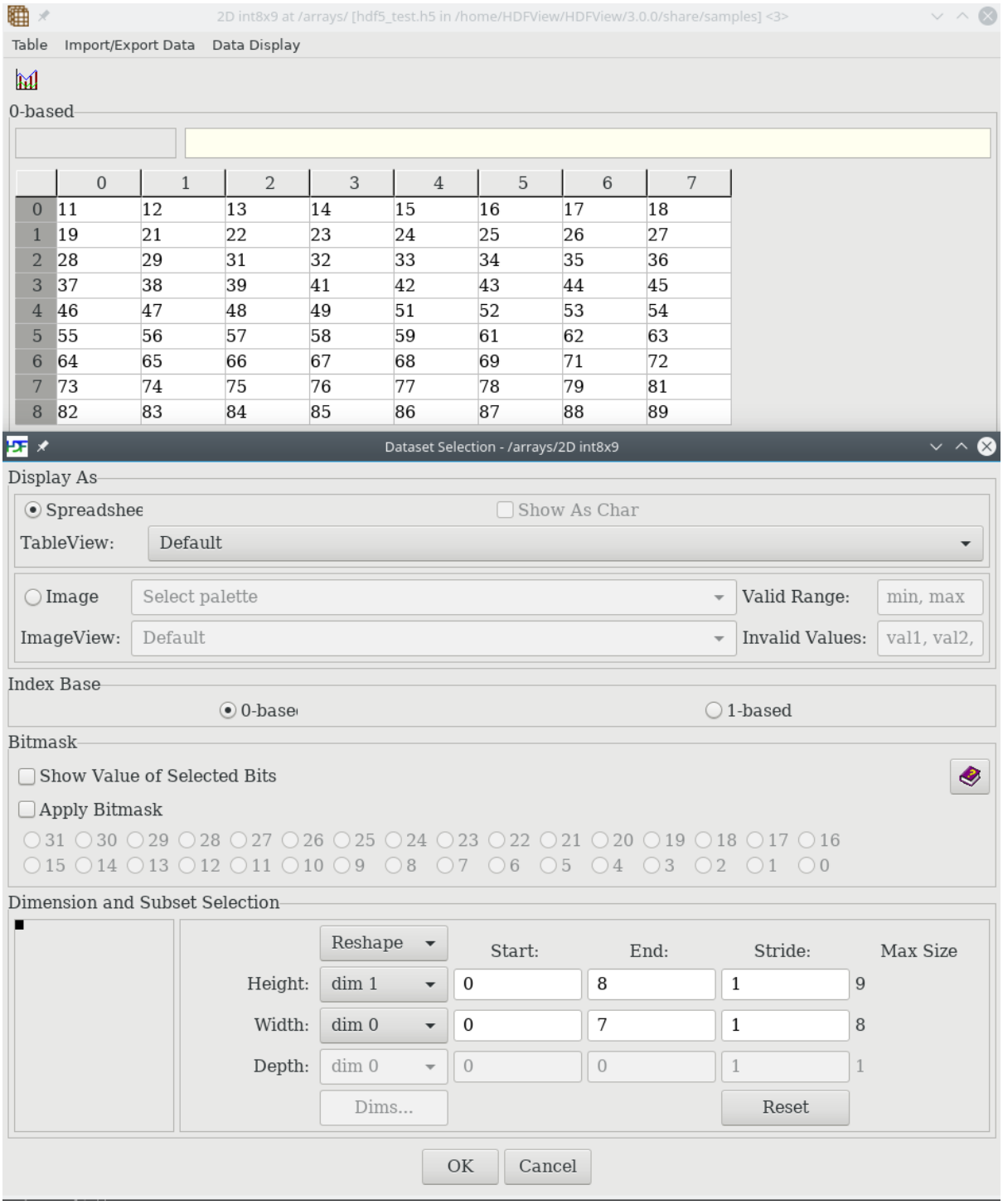

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 11 | 21 | 31 | 41 | 51 | 61 | 71 | 81 |
| 1 | 12 | 22 | 32 | 42 | 52 | 62 | 72 | 82 |
| 2 | 13 | 23 | 33 | 43 | 53 | 63 | 73 | 83 |
| 3 | 14 | 24 | 34 | 44 | 54 | 64 | 74 | 84 |
| 4 | 15 | 25 | 35 | 45 | 55 | 65 | 75 | 85 |
| 5 | 16 | 26 | 36 | 46 | 56 | 66 | 76 | 86 |
| 6 | 17 | 27 | 37 | 47 | 57 | 67 | 77 | 87 |
| 7 | 18 | 28 | 38 | 48 | 58 | 68 | 78 | 88 |
| 8 | 19 | 29 | 39 | 49 | 59 | 69 | 79 | 89 |

| | Start: | End: | Stride: | Max Size |
|---|---|---|---|---|
| Height: dim 1 | 0 | 8 | 1 | 9 |
| Width: dim 0 | 0 | 7 | 1 | 8 |
| Depth: dim 0 | 0 | 0 | 1 | 1 |

**Swap row/column dimensions -- row index = dim[1], column index = dim[0]**

#### *5.2.4.3 Example of Transpose data -- row index = dim[1], column index = dim[0]*

In some cases, we also want to reshape the data (not changing the data order) when we swap the row and column dimensions. To reshape the data, select the “Reshape” option from the drop-down menu. For example, reshape the data to display the dataset as nine rows and eight columns. The order of data stays the same (counting from the location [0, 0], [0, 1], ... [2, 0], [2, 1], ...).

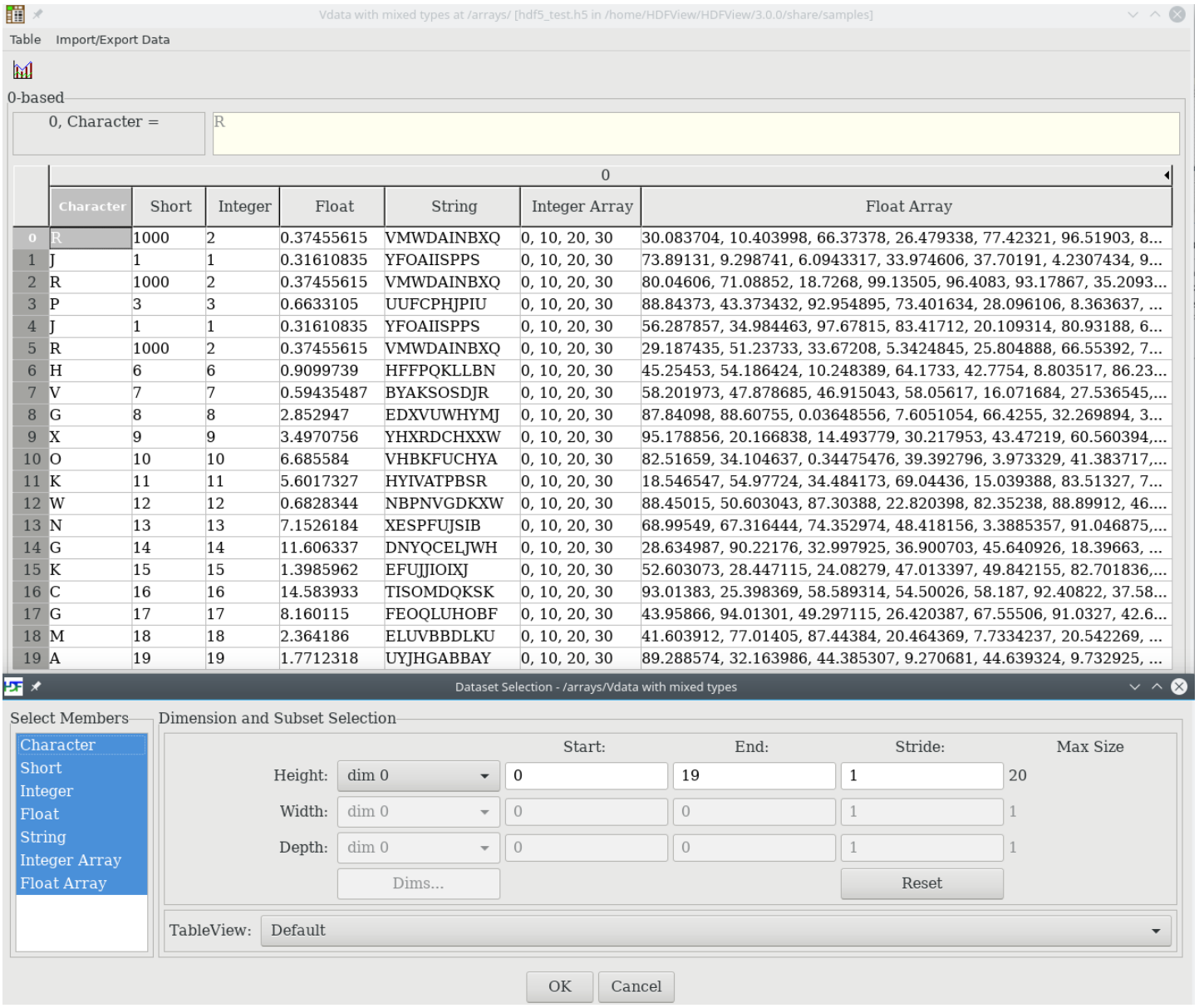



| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 |
| 1 | 19 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| 2 | 28 | 29 | 31 | 32 | 33 | 34 | 35 | 36 |
| 3 | 37 | 38 | 39 | 41 | 42 | 43 | 44 | 45 |
| 4 | 46 | 47 | 48 | 49 | 51 | 52 | 53 | 54 |
| 5 | 55 | 56 | 57 | 58 | 59 | 61 | 62 | 63 |
| 6 | 64 | 65 | 66 | 67 | 68 | 69 | 71 | 72 |
| 7 | 73 | 74 | 75 | 76 | 77 | 78 | 79 | 81 |
| 8 | 82 | 83 | 84 | 85 | 86 | 87 | 88 | 89 |

**Transpose data -- row index = dim[1], column index = dim[0]**

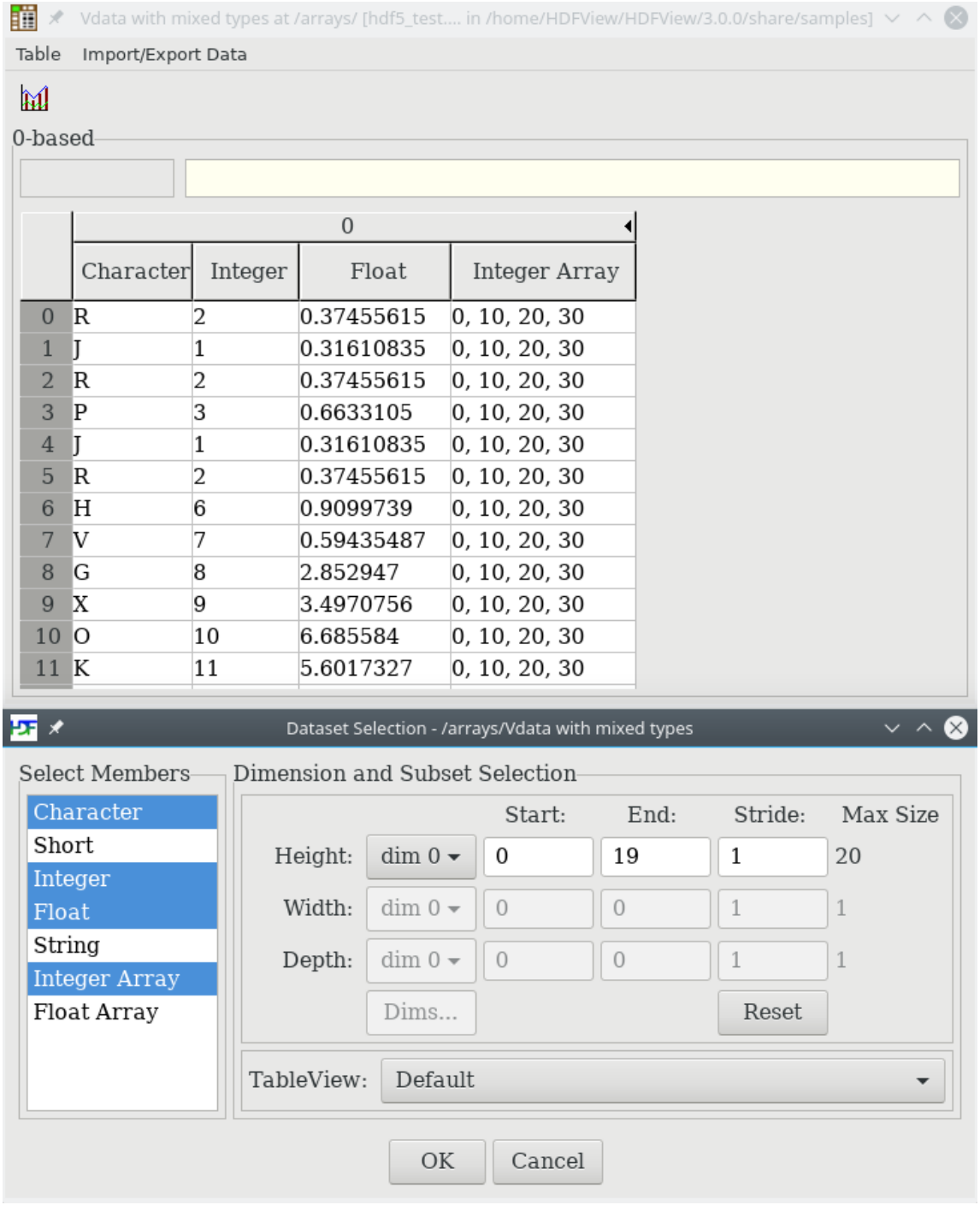
### 5.2.5 Compound Dataset Options

HDFView displays HDF4 Vdata and a simple, one-dimension HDF5 compound dataset (without a nested compound) as a 2-D table with rows as records and columns as fields/members.




**Compound dataset**

You can also select fields/members to display. For a contiguous selection, hold down the “Shift” key while clicking the first and last fields/members of your selection. For a discontiguous selection, hold down the “Ctrl” key while clicking the fields/members that you want to select.

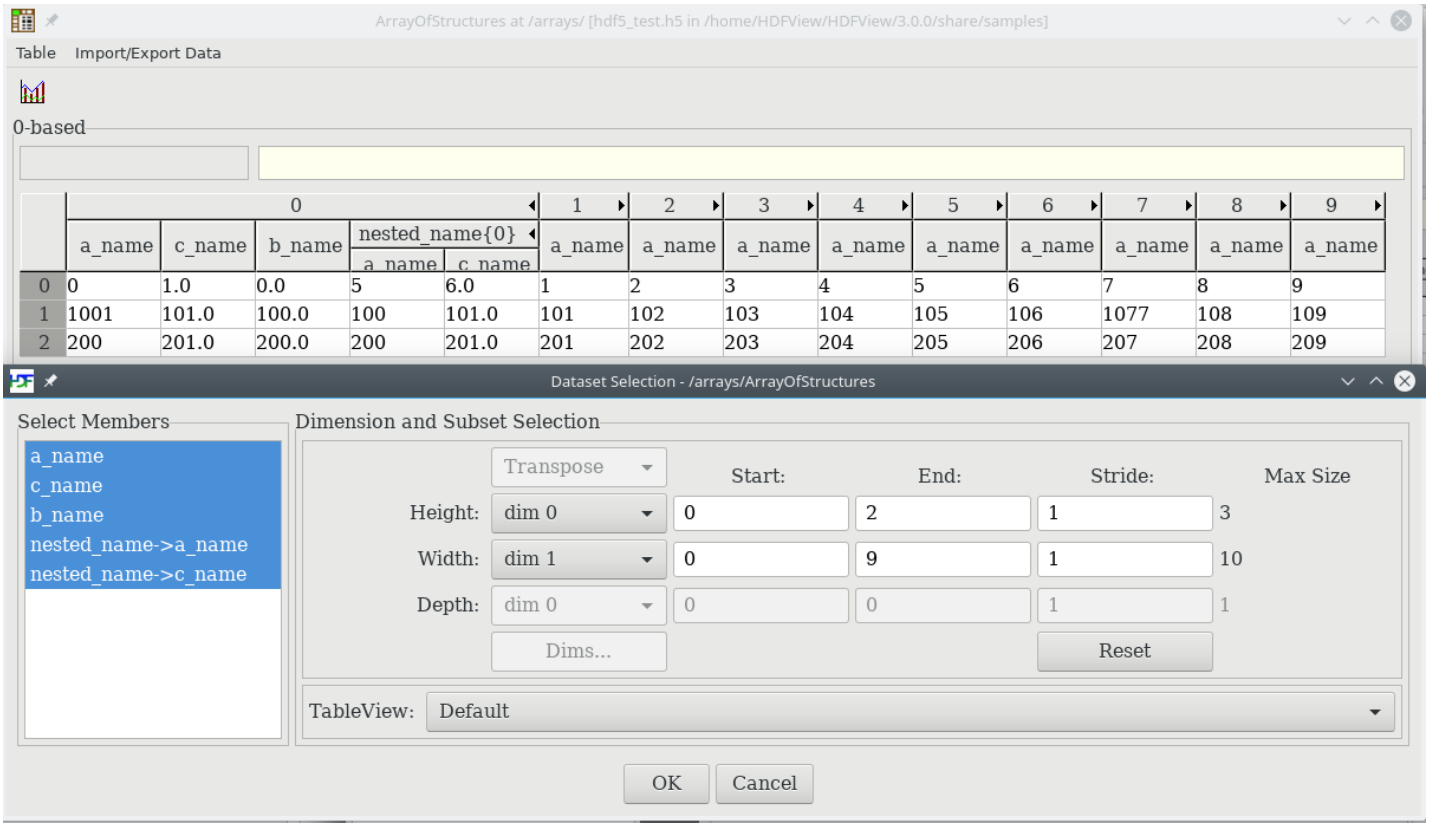



**Field/Member selection of a compound dataset**

HDFView displays a nested HDF5 compound dataset as a nested list of members, only for two levels. The nested names are separated by "->" for more deeply nested names. For example, if a compound dataset "A" has the following nested structure:

```
A --> a_name
A --> b_name
A --> c_name
```

```
A --> nested_name --> a_name
A --> nested_name --> c_name
i.e.
A = {a_name, b_name, c_name, nested_name{a_name, c_name}}
```



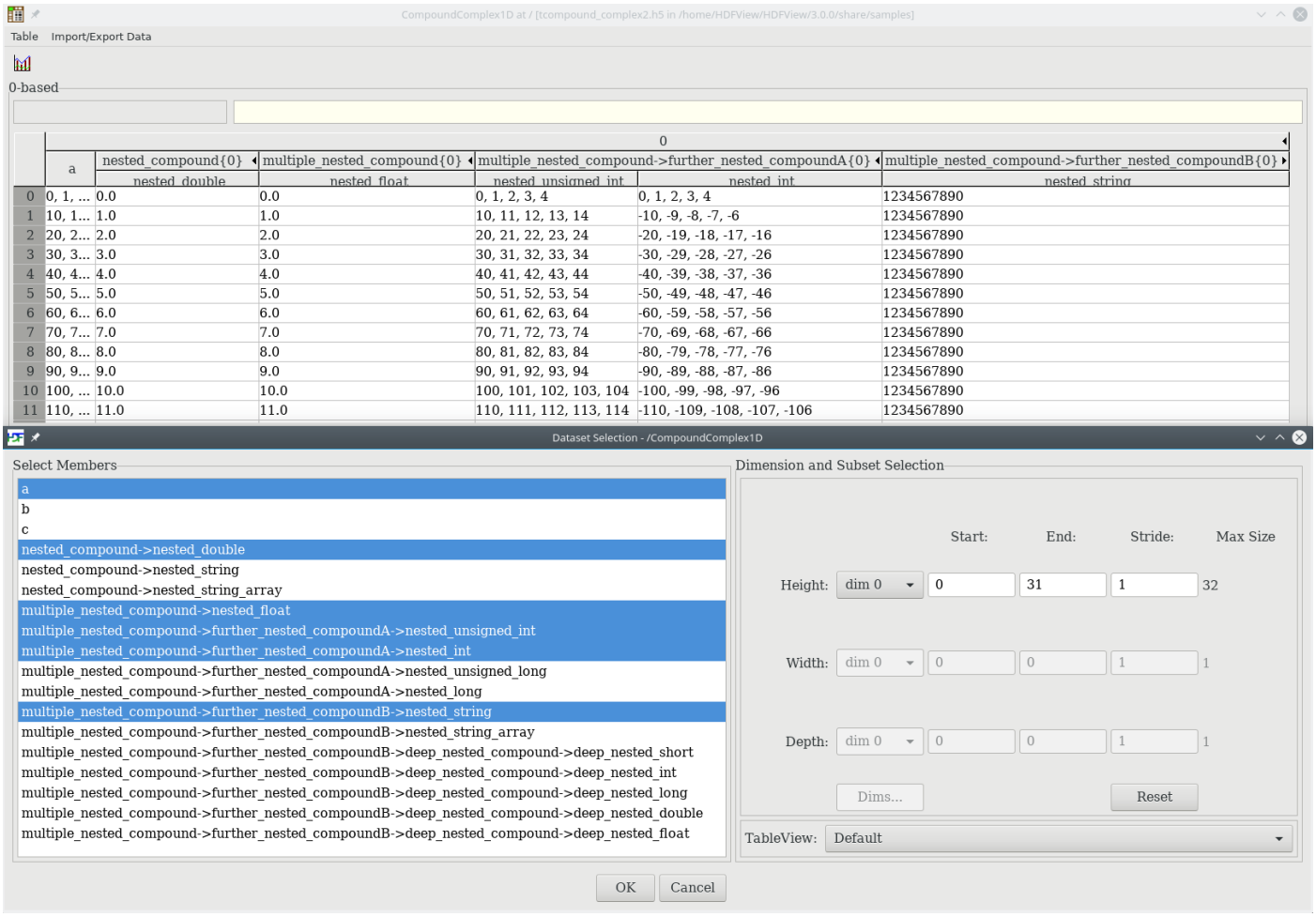

| | 0 | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | a_name | c_name | b_name | nested_name{0} | a_name | a_name | a_name | a_name | a_name | a_name | a_name | a_name | a_name |
| | | | | a_name | c_name | | | | | | | | |
| 0 | 0 | 1.0 | 0.0 | 5 | 6.0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | 1001 | 101.0 | 100.0 | 100 | 101.0 | 101 | 102 | 103 | 104 | 105 | 106 | 1077 | 108 | 109 |
| 2 | 200 | 201.0 | 200.0 | 200 | 201.0 | 201 | 202 | 203 | 204 | 205 | 206 | 207 | 208 | 209 |

**Nested compound dataset**

If the nested_name->a_name member of the compound dataset “A” had members:

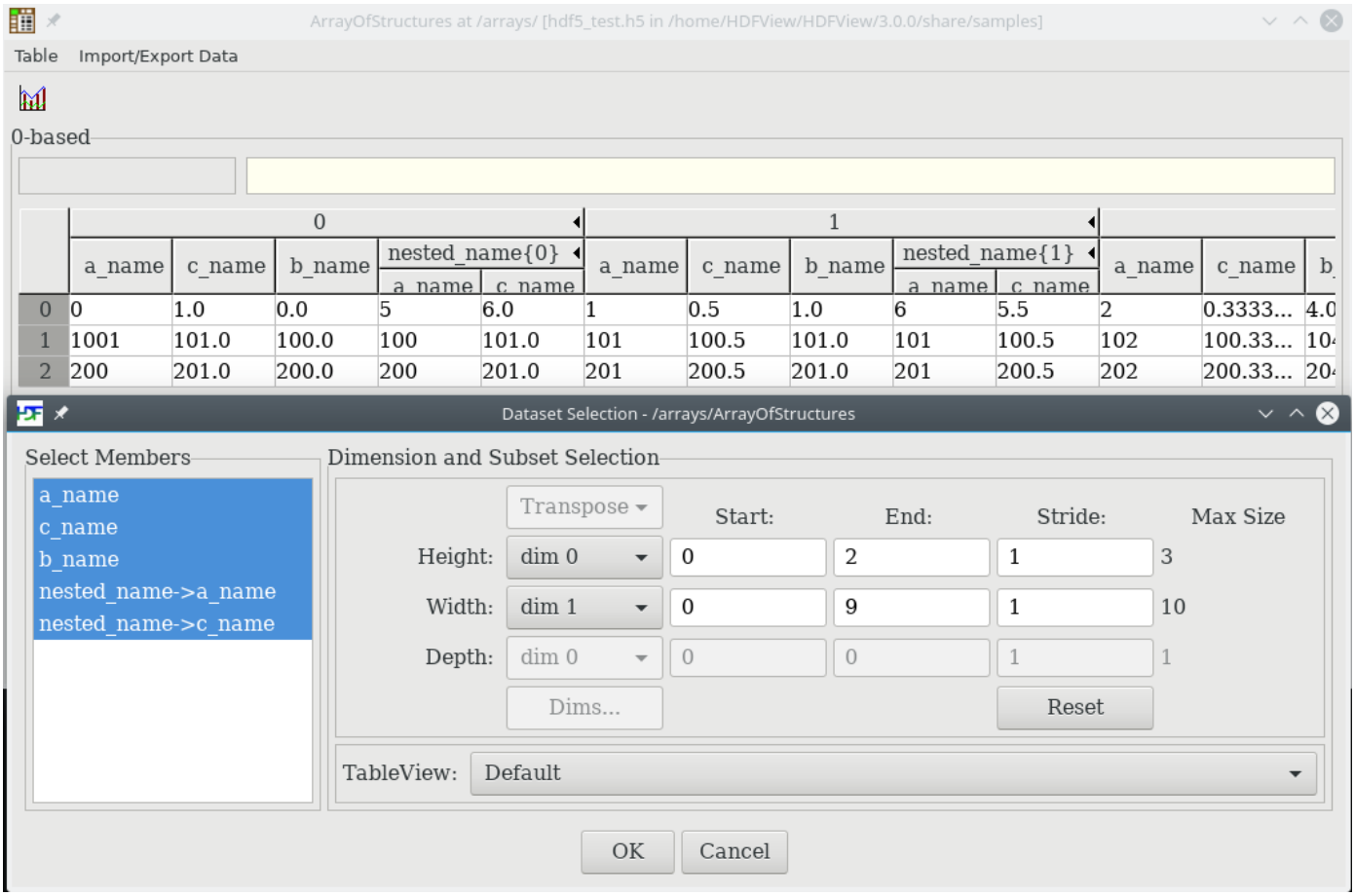
```
A --> a_name
A --> b_name
A --> c_name
A --> nested_name --> a_name --> d_name
A --> nested_name --> c_name --> e_name
i.e.
A = {a_name, b_name, c_name, nested_name{a_name{d_name, e_name}, c_name}}
```

Then the more deeply nested header names include the full parent name with “->”.




**Nested compound dataset**

HDFView displays multi-dimension compound datasets as a 2-D table with nested sub-columns. The members are shown in the sub-columns with nested header names appended with {#} where # is the dimension index.

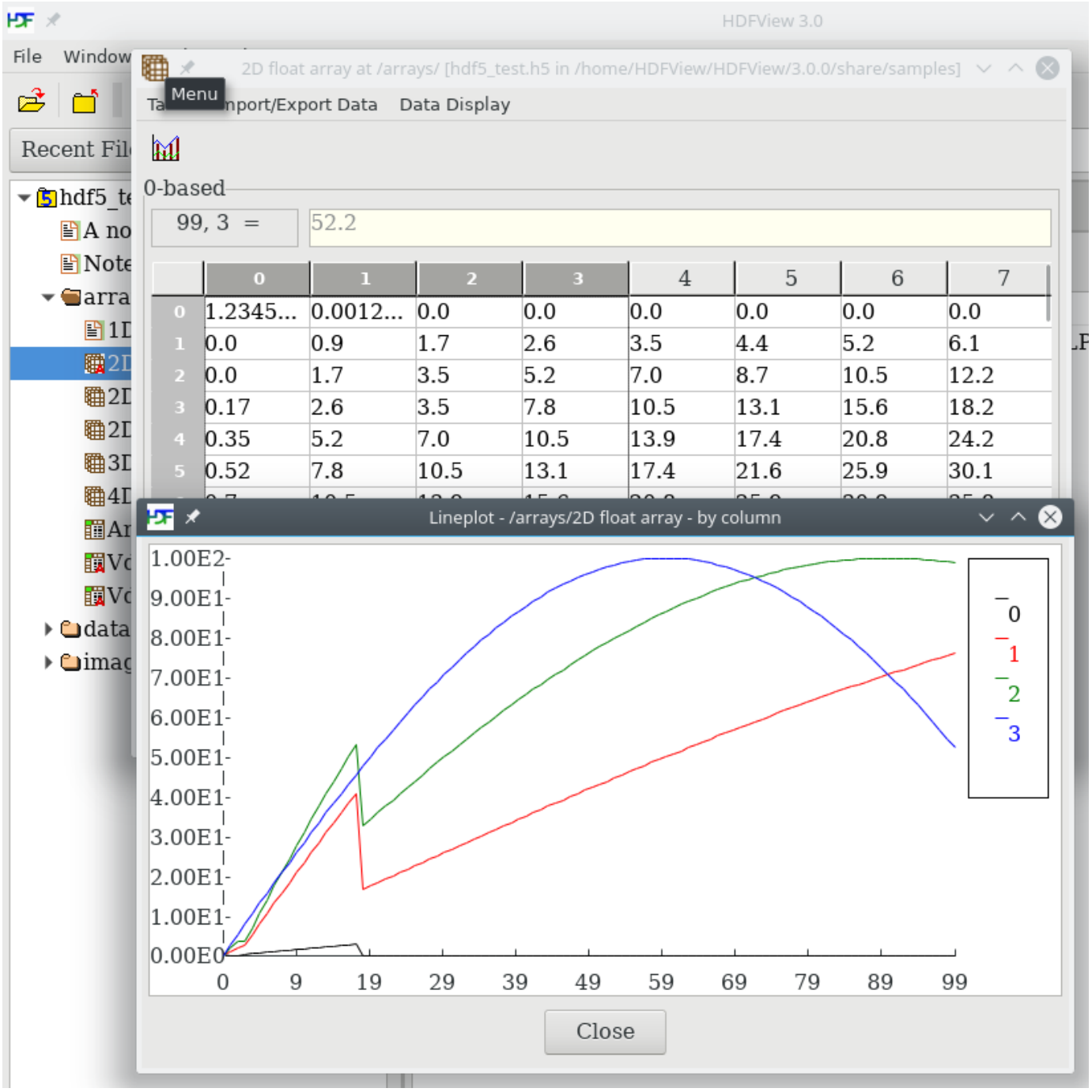



**2-D compound dataset**

# 5.3 Display a Column/Row Line Plot

Column or row data of numerical values can be displayed in a simple line plot. Select rows or columns by dragging the mouse on the rows or columns that you want to plot; then click the chart icon:  . The row or column data is plotted against a column or row index, respectively.

The following figure shows that data of five columns are displayed in five lines of different colors. The horizontal labels are the row index of the 100 data points. The vertical labels are the ten points of equal data range with the maximum and minimum of the column data. The line legend is drawn at the right of the line plot with column names and line colors.




| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 1.2345... | 0.0012... | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| 1 | 0.0 | 0.9 | 1.7 | 2.6 | 3.5 | 4.4 | 5.2 | 6.1 |
| 2 | 0.0 | 1.7 | 3.5 | 5.2 | 7.0 | 8.7 | 10.5 | 12.2 |
| 3 | 0.17 | 2.6 | 3.5 | 7.8 | 10.5 | 13.1 | 15.6 | 18.2 |
| 4 | 0.35 | 5.2 | 7.0 | 10.5 | 13.9 | 17.4 | 20.8 | 24.2 |
| 5 | 0.52 | 7.8 | 10.5 | 13.1 | 17.4 | 21.6 | 25.9 | 30.1 |

**Line plot**

# 5.4 Change Data Value

You can change the values of a dataset in two ways: type data into the table cell or paste data from the system clipboard. HDFView rejects invalid data values. For example, it does not accept a floating point number into an integer dataset. The table below lists the rules for entering data. ***Any changes of data values only exist in memory. They are not saved to file until you choose***

***"Save" from the File menu, or, when you dismiss the table, you will be asked if you want to save the changes to the file.***

| Data type | Acceptable formats |
|---|---|
| byte | -127 to 128 |
| short | -32768 to 32767 |
| int | -2147483648 to 2147483647 |
| float, double | Numbers of the form "99.9" or "8", and "-9.9" or "-9", and ".9" or "-.9" Numeric overflow or underflow will be detected. |
| string | A string longer than the stored value will be silently truncated when written to the file. |
| unsigned byte | 0 to 255 |
| unsigned short | 0 to 65535 |
| unsigned int | 0 to 4294967295 |

**Data formats for entering data**

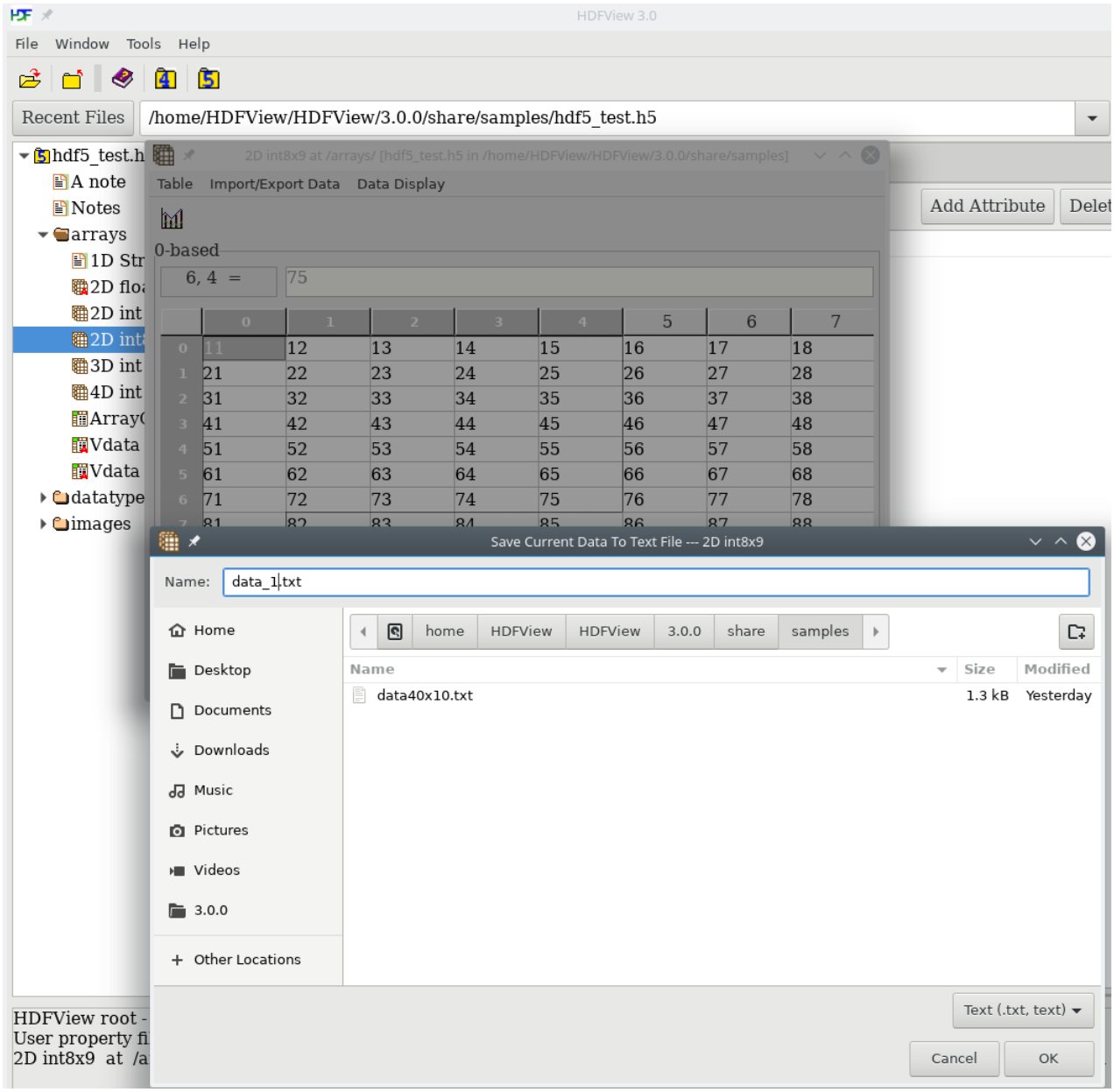
Data values can be copied with "Copy" and "Paste". Data can be copied within a spreadsheet or between two spreadsheets. To copy data, select the data cells to copy, then choose the "Copy" command from the Table menu. Then select the cells to paste into and select the "Paste" command from the Table menu.

You can also copy and paste between HDFView and other applications. To copy external data from other applications such as a text editor or Microsoft Excel, select and copy data from the application, then paste the data into the HDFView TableView, and vice versa.

You can also change values by using predefined math functions. To change table values, select the data area and choose the "Math conversion" command from the Table menu. A list of predefined mathematic functions are provided. Select a function and enter the function parameters. The values of the selected data cells will be changed based on the mathematic function.

# 5.5 Save Data Values to a Text File

Writing table data into an ASCII file is nearly transparent. Select “Import/Export Data” from the menu bar, then select “Export data to Text File” from the dropdown menu, and the Save Current Data to Text File dialog box pops up for you to enter the name of the file. The data values of the current table will be written to the file. The data values are separated by the data delimiter specified in “User Options”. ***The text file does not contain any datatype and dataspace information.***.




**Save current data to text file**

# 5.6 Import Data from a Text File

You can fill the table cells directly from a text file. Select “Import/Export Data” from the menu bar, then select the “Import Data From Text File” command from the dropdown menu and select the text file to import. The data values must be separated by a space or the delimiter specified in “User Options”. The cells of the table are filled row by row starting with the selected cell. The line breaks in the text file are not important. For example,

```
10 11 12 13 14 15
16 17 18 19 20 21
```

and

```
10 11 12 13
14 15 16 17
18 19
20 21
```

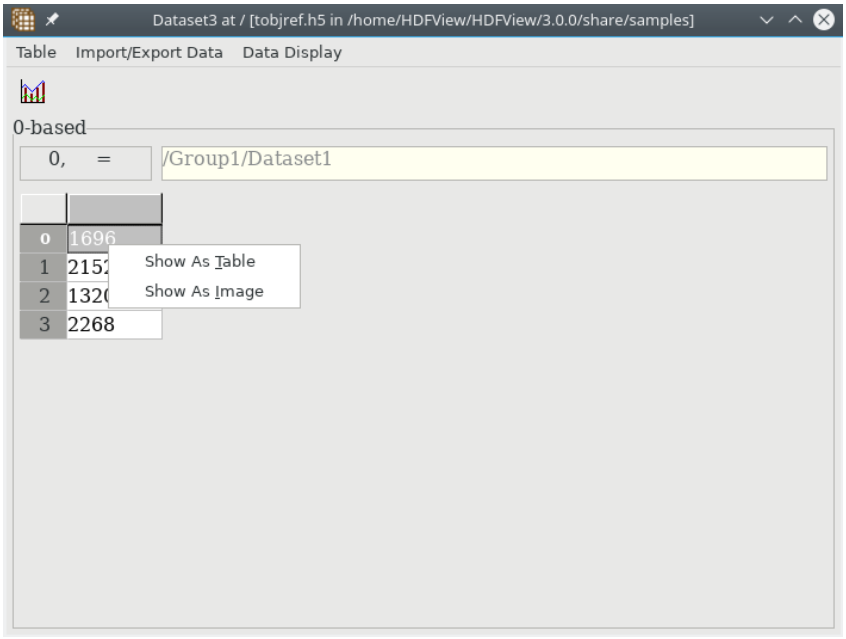
are the same.

# 5.7 Dataset storing references

A dataset can store reference values of other datasets in the same file.

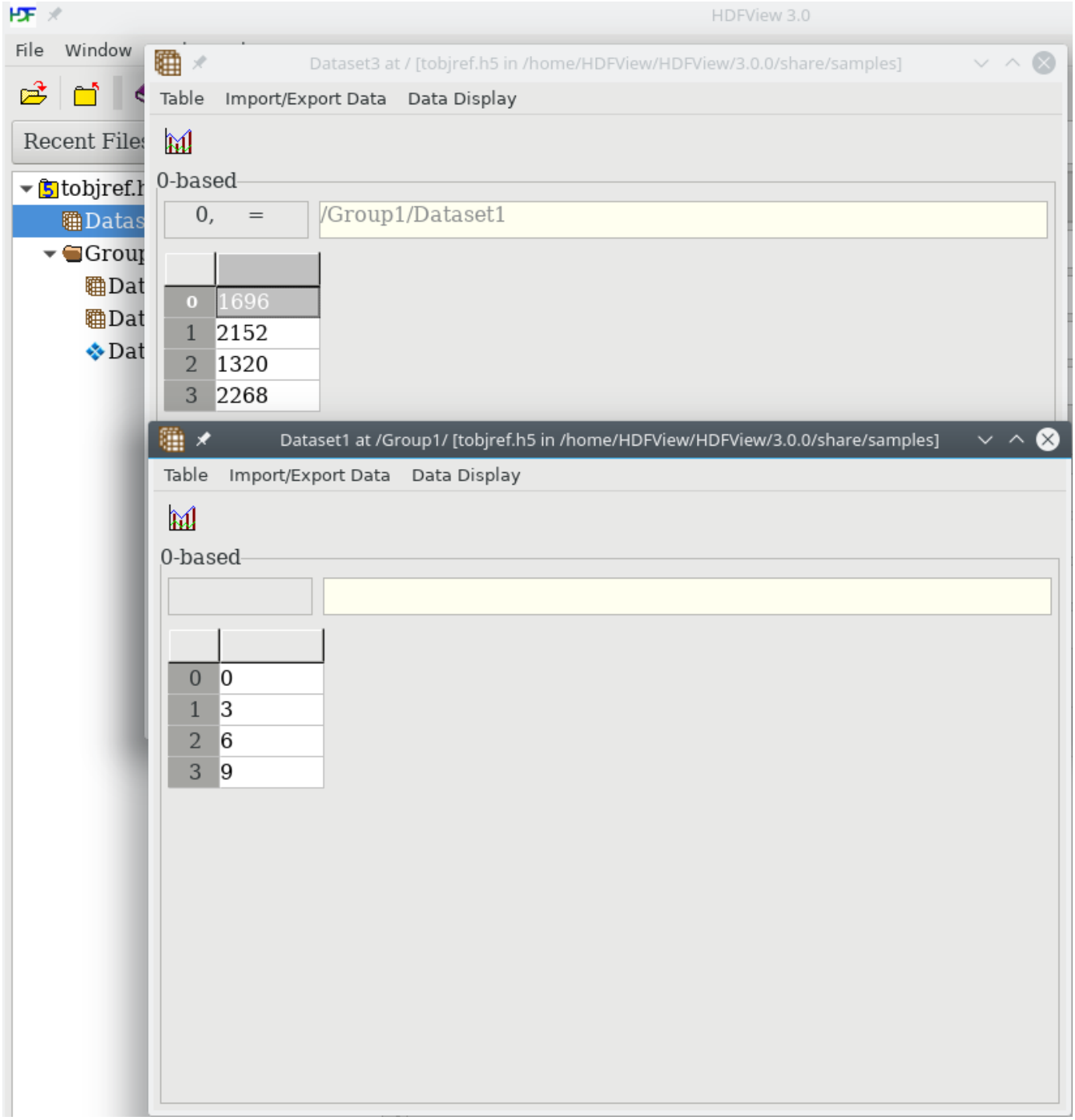
### 5.7.1 Dataset Storing Object References

A dataset can store object references of other datasets. You can open a dataset containing object references. You can then select and right click on any value in the dataset and it will give two options, to either open as a table or an image.




**Dataset storing object references**

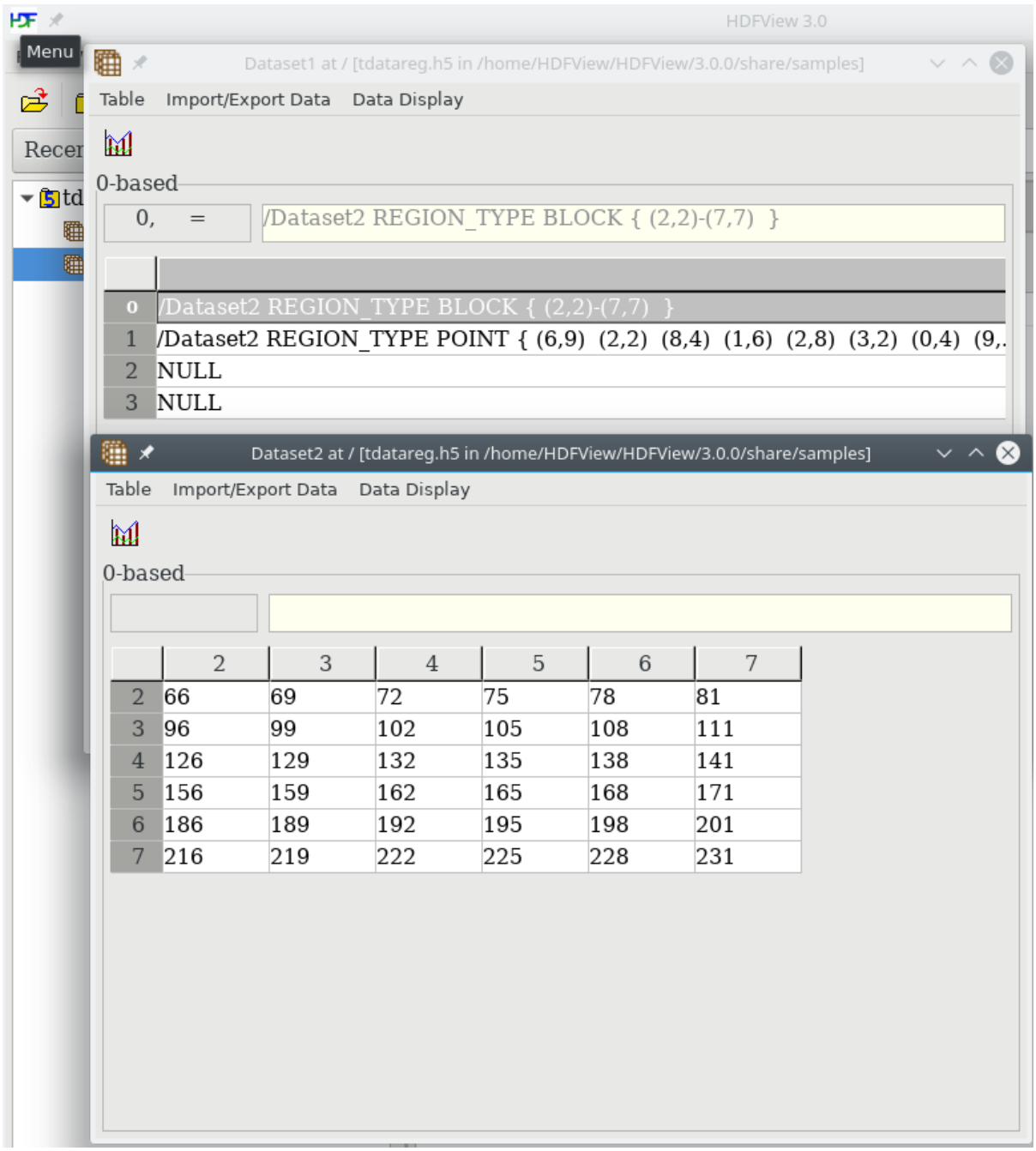
If opened as a table, then the dataset of reference value “1696” is opened as shown below.



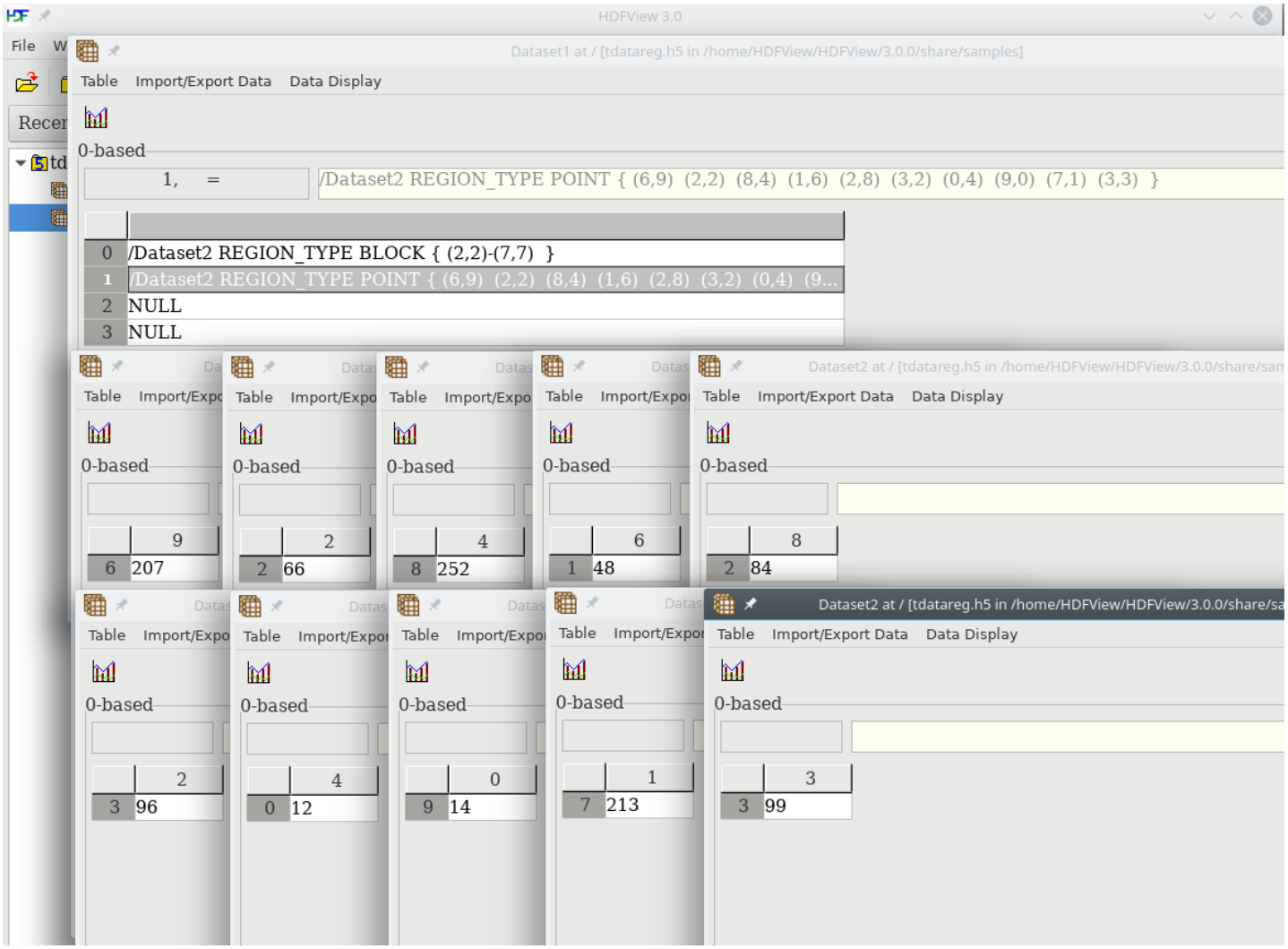

**Dataset pointed by object reference**

### 5.7.2 Dataset Storing Dataset Region References

A dataset can store dataset region reference values. You can right click on any value to either show the dataset as a table or an image.



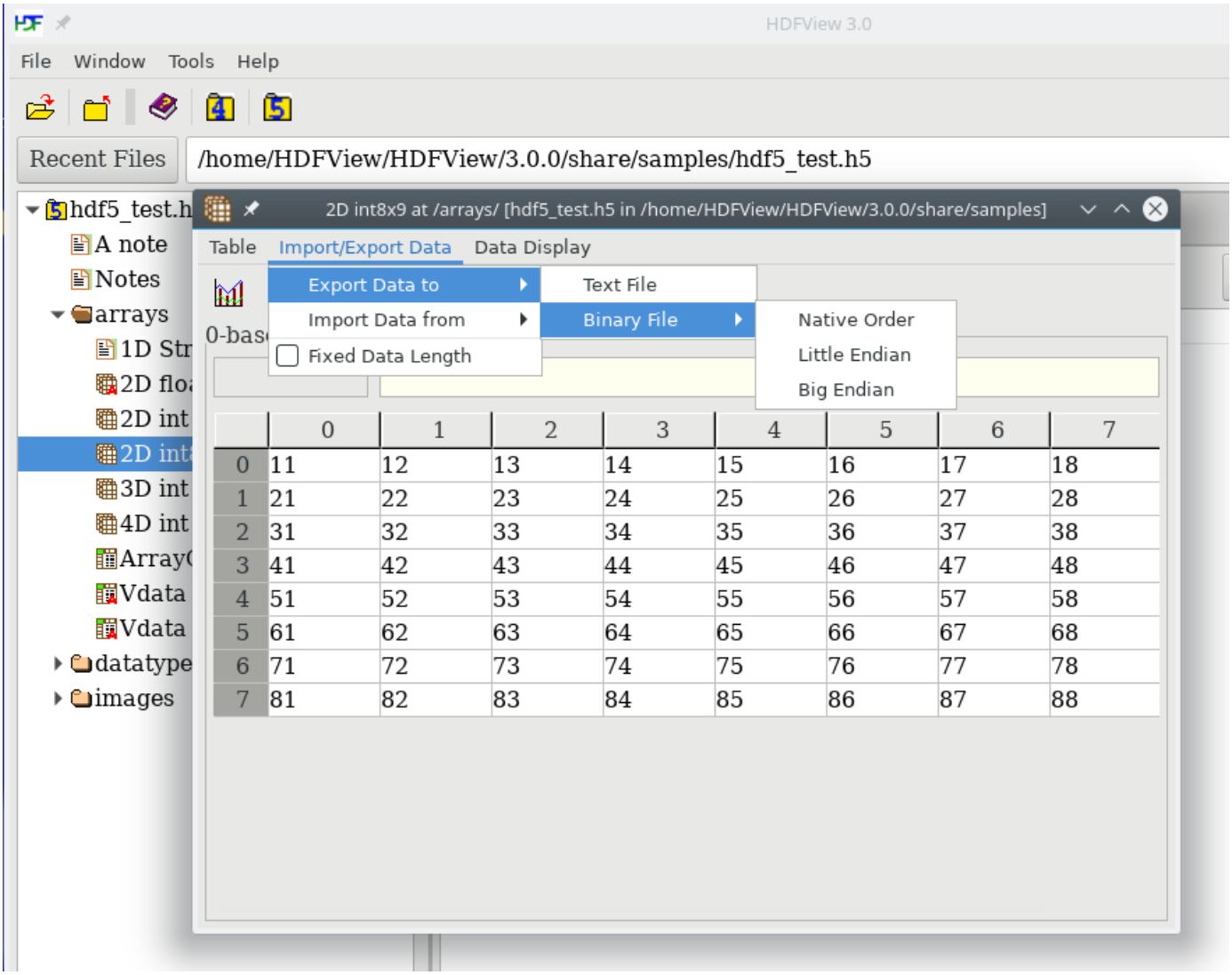

**Dataset pointed by region reference block reference**




**Dataset pointed by region reference point reference**

# 5.8 Save Data Values to a Binary File

The table data can be written to a binary file. Select “Import/Export Data” from the menu bar, then select the “Export Data to Binary File” from the dropdown menu. Select the order in which you want the bytes to be. The Save Current Data to Binary File dialog box pops up for you to enter the name of the file. The data values of the current table will be written to the file. ***The binary file does not contain any datatype and dataspace information. Currently, only the entire contents of the table are written to a binary file.***.




**Save current data to binary file**

# 5.9 Import Data from a Binary File

You can fill the table directly from a binary file. Select “Import/Export Data” from the menu bar, then select the “Import Data From Binary File” command from the dropdown menu and select the byte order. It is assumed that the user should know the data type in the binary file and the byte order. Select the binary file to import. The cells of the table are filled with the corresponding values.

---

*[Table of Contents]*

# Chapter 6: Image Viewer

Image Viewer is a graphical window that displays HDF images. HDFView is a simple image viewer for HDF4/5 and has a limited function for processing an image.

An HDF4 image is a raster image of 8-bit pixels with an indexed RGB color table, or a 24-bit true color image. The HDF4 library provides image APIs to access image data and color tables.

An HDF5 image is a dataset that conforms to the HDF5 Image Specification. HDFView supports two types of images: indexed and true color. Both indexed image and true color image have predefined attributes and data layout according to the HDF5 image specification. For more details about HDF5 image, see the HDF5 Image Specification.



---

## 6.1 Display a 2-D or 3-D Image

HDFView displays HDF4 raster images or HDF5 datasets that follow the HDF5 Image and Palette Specification for indexed images with an 8-bit standard RGB color model palette or three-dimensional true color images. Other image formats supported by the Image and Palette Specification are not supported by this tool.

If an image is larger than the visible area of the image viewer, users can grab and move the image by mouse-drag. Users can also use the side scroller bar to move the image to a desirable view area. Use Shift+Mouse-drag to select a subset of an image in the image viewer.

Fill values are displayed as the color black. Fill values are also excluded from calculating the minimum, maximum and other statistics.

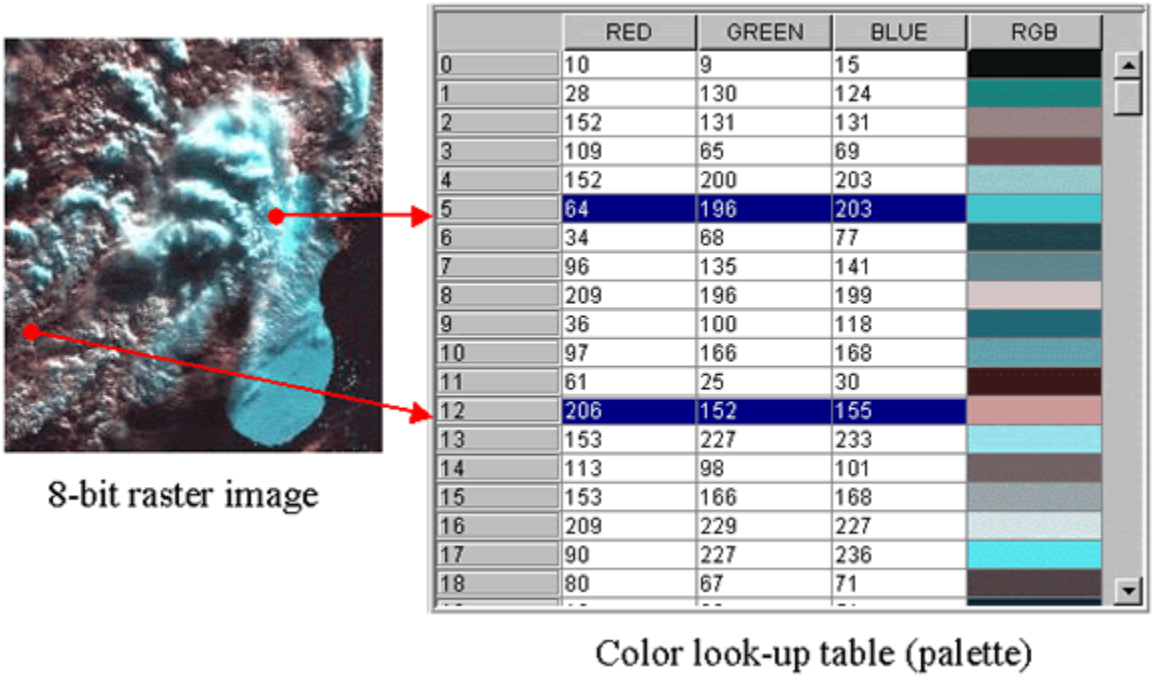
### 6.1.1 Indexed Image (8-Bit)

An indexed image is one of the following:

- An HDF4 RI8 image
- An HDF5 dataset that conforms to the HDF5 Image specification, and is an "IMAGE_SUBCLASS=IMAGE_INDEXED"
- An SDS or HDF5 dataset with data that can be interpreted as an image

The dataset is displayed as a Java image using IndexColorModel. The dataset is converted to a raster image using the first palette specified by the PALETTE attribute, or the default palette for HDF4. Multiple user-defined palettes (i.e., the PALETTE attribute may be a list) are not supported in version 1.0.

The dataset of an indexed image holds the values of indices of the color look-up table (palette). The dataset is converted into image pixels by looking up the index in the color table. The following figure is an example of mapping dataset values into pixels.




| | RED | GREEN | BLUE | RGB |
|---|---|---|---|---|
| 0 | 10 | 9 | 15 | |
| 1 | 28 | 130 | 124 | |
| 2 | 152 | 131 | 131 | |
| 3 | 109 | 65 | 69 | |
| 4 | 152 | 200 | 203 | |
| 5 | 64 | 196 | 203 | |
| 6 | 34 | 68 | 77 | |
| 7 | 96 | 135 | 141 | |
| 8 | 209 | 196 | 199 | |
| 9 | 36 | 100 | 118 | |
| 10 | 97 | 166 | 168 | |
| 11 | 61 | 25 | 30 | |
| 12 | 206 | 152 | 155 | |
| 13 | 153 | 227 | 233 | |
| 14 | 113 | 98 | 101 | |
| 15 | 153 | 166 | 168 | |
| 16 | 209 | 229 | 227 | |
| 17 | 90 | 227 | 236 | |
| 18 | 80 | 67 | 71 | |

**Mapping of dataset values to image pixels**

For a two-dimensional indexed image, HDFView assumes that the width of the image is the size of the second dimension and the height of the image is the size of the first dimension, i.e. dim[0]=height and dim[1]=width.

Although HDFView displays the entire image by the order of (dim[0], dim[1], dim[2])=(depth, height, width) by default, you can always change the order and select a subset for the display as discussed in Chapter 5.

HDFView also displays a three-dimensional array as an array of 2-D images arranged along the third dimension, i.e. dim[0]=depth, dim[1]=height and dim[2]=width. You can flip back and forth to look at images at a different position of the depth dimension. For instance, if the dataset is 20 x 400 x 600 (dim[0]=20, dim[1]=400, and dim[2]=600), HDFView will display it as 20

images each with the size of 600 x 400 (width is 600, height is 400). However, a three-dimension image of [1][height][width] or [height][width][1] is treated as a two-dimension indexed image of [height][width].

A 2-D or 3-D SDS or HDF5 dataset with integer or float data can be displayed as an indexed image using the "Open As" selection from the Context menu. Since the dataset does not have a palette, a default palette is used. The palette is chosen from the "Select Palette" menu in the "Dataset Selection" window. The predefined palettes include:

- gray
- rainbow
- nature
- wave

If no palette is selected, a gray scale will be used.

## Converting non-byte data to byte data

Non-byte data will be converted to image byte data in one of the two algorithms: simple linear conversion or auto gain conversion.

Simple linear conversion

y = [(x - min)/(max-min)]*255.
Where,
x is the original value;
y is the byte value.

Auto gain conversion

y = (x + bias) * gain.
Where,
x is the original value;
y is the byte value;
gain = MAX_VALUE / (max-min);
bias = -min;
MAX_VALUE is the max value of original data type, e.g. USHRT_MAX for unsigned short integers.

Auto gain algorithm will be used only if Auto gain is selected from "Tools" ==> "User Options", and the original dataset does not have any attached palette and you did not select any palette from the "Open As" window. Simple linear conversion will be applied to all other cases. The image values will be different and the image will look different if a different algorithm is used.

### 6.1.2 True Color Image

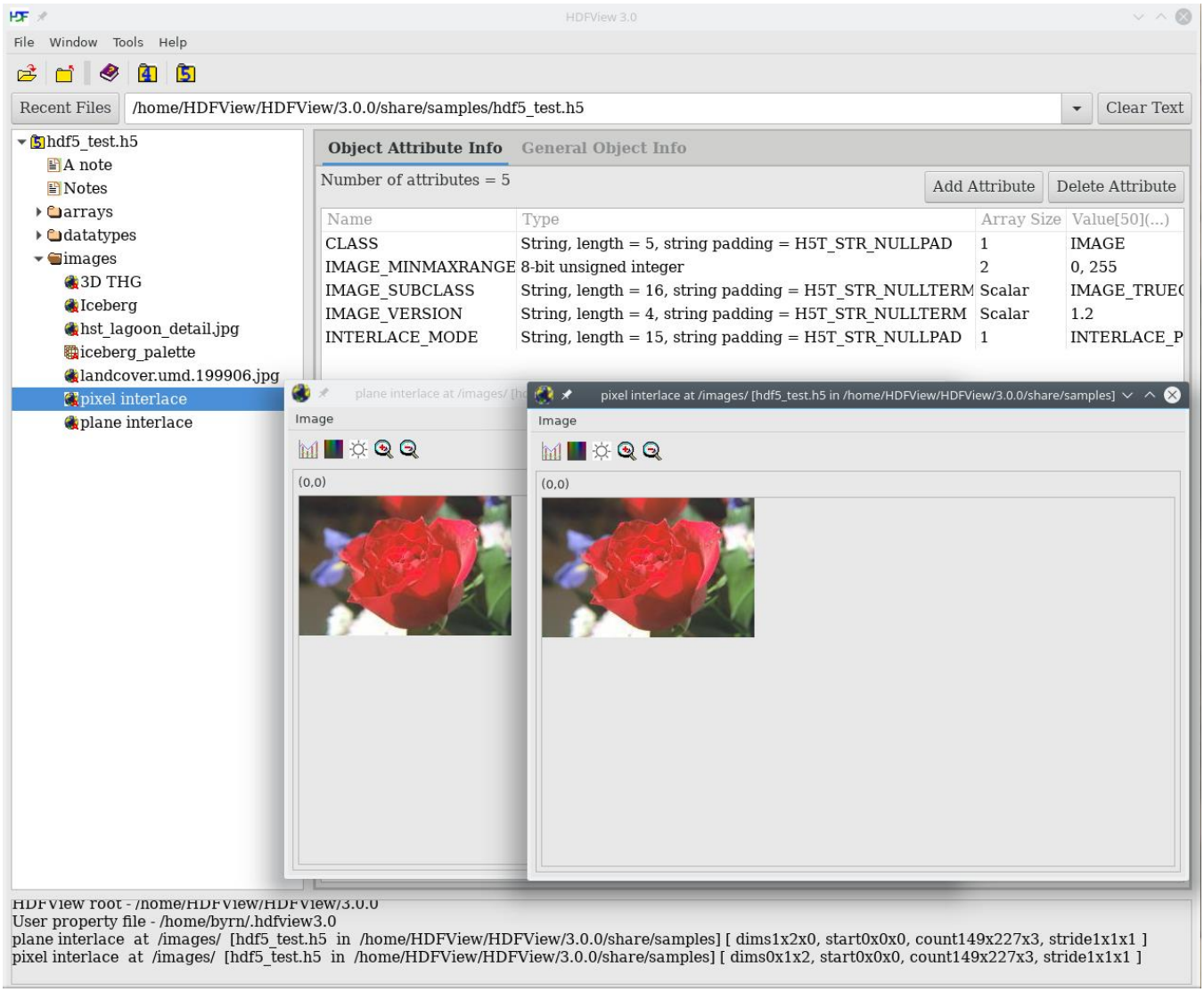
In the case of an image with more than one component per pixel (e.g., red, green, and blue), the data may be arranged in one of two ways. HDFView only supports three color components: red, green and blue.

Following HDF4 terminology, the data may be interlaced by *pixel* or by *plane*. For an HDF5 Image dataset the interlace should be indicated by the INTERLACE_MODE attribute. In both cases, the dataset will have a dataspace with three dimensions, *height*, *width*, and *components*. For *pixel interlace*, the data is arranged in the following order: [height][width][pixel components]. For *plane interlace*, the data is arranged in the following order: [pixel components][height][width].

The translation from pixel values to color components for display or processing purposes is a one-to-one correspondence of data values to components. Data of RGB color components is converted into byte data, which is packed into a single *int* pixel. The Java Image is created with a DirectColorModel, with masks to define packed samples. This color model is similar to an X11 TrueColor visual. The default RGB ColorModel is specified with the following parameters:

```
Number of bits:         32
Red mask:               0x00ff0000
Green mask:             0x0000ff00
Blue mask:              0x000000ff
Alpha mask:             0xff000000
Color space:            sRGB
isAlphaPremultiplied:   False
Transparency:           Transparency.TRANSLUCENT
transferType:           DataBuffer.TYPE_INT
```

The following figure shows examples of true color images. The image on the right is pixel interleaving with dimensions of [149][227][3]. The image on the leftt is plane interleaving with dimensions of [3][149][227].




**True color image displayed in the Image View**

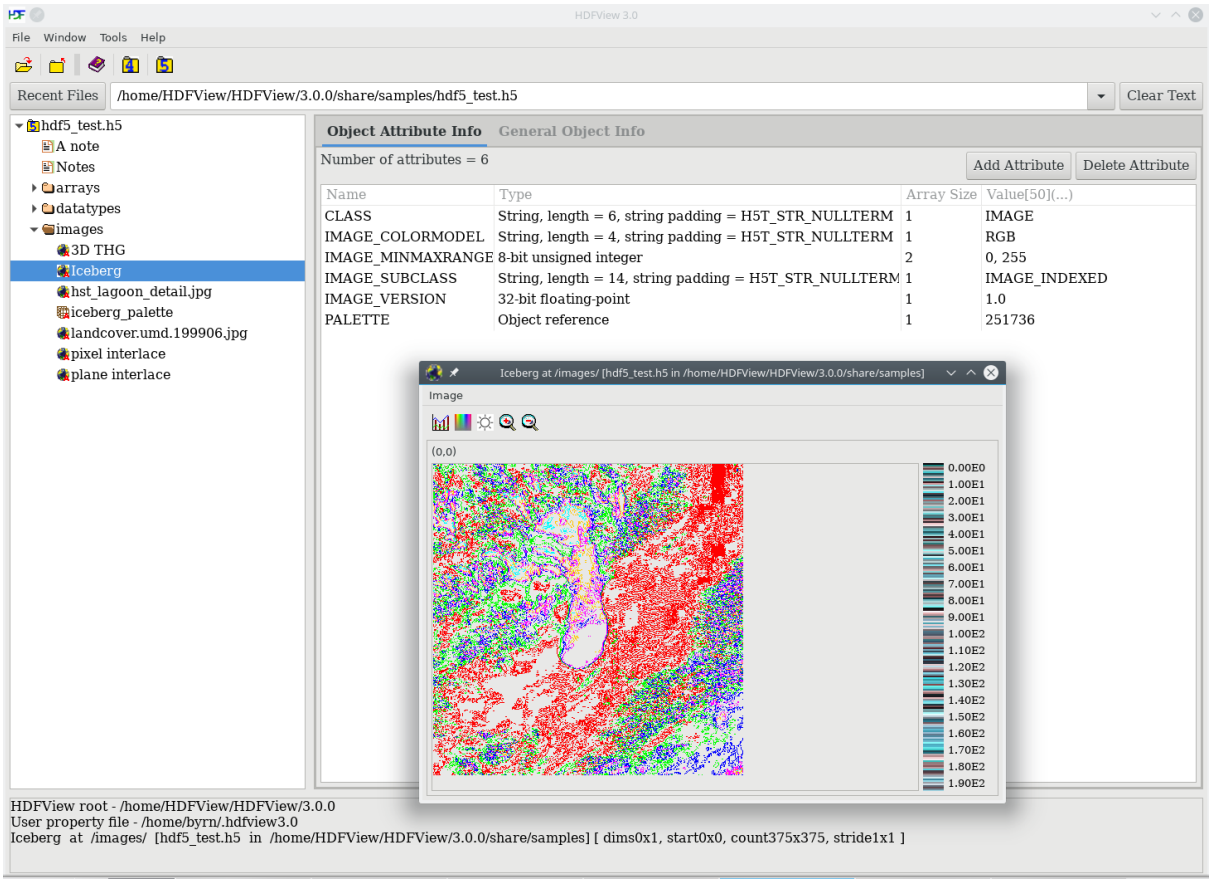
# 6.2 Zoom/Flip/Contour Image

HDFView supports only limited image manipulation such as zooming, flipping, and contour. You can zoom in and out of an image. The minimum zoom factor is 1/8 (reduced to 1/8 the size) and the maximum is 8 (magnified to 8 times the size). Reduction (zoom out) is done by sampling pixels, for example, a 1/2-size image is created by selecting every second pixel. Magnification (zoom in) is done by replicating pixels.

You can also flip an image horizontally or vertically. Flipping an image will change the coordinates of the image. This technique can be used to adjust images that may have been created with different origins from the defaults.

Manipulating the “contour” creates a contour plot of the pixel values. The contour can have from three to nine contour levels. Level three provides less details of contour and level nine provides more. **Repeated contour operations show a *cumulative* effect of contouring. For example, if you contour an image with level 3 and then contour that resulting image with level 4, the final image shows the cumulative effect of applying a level 4 to the level 3–contoured image**.

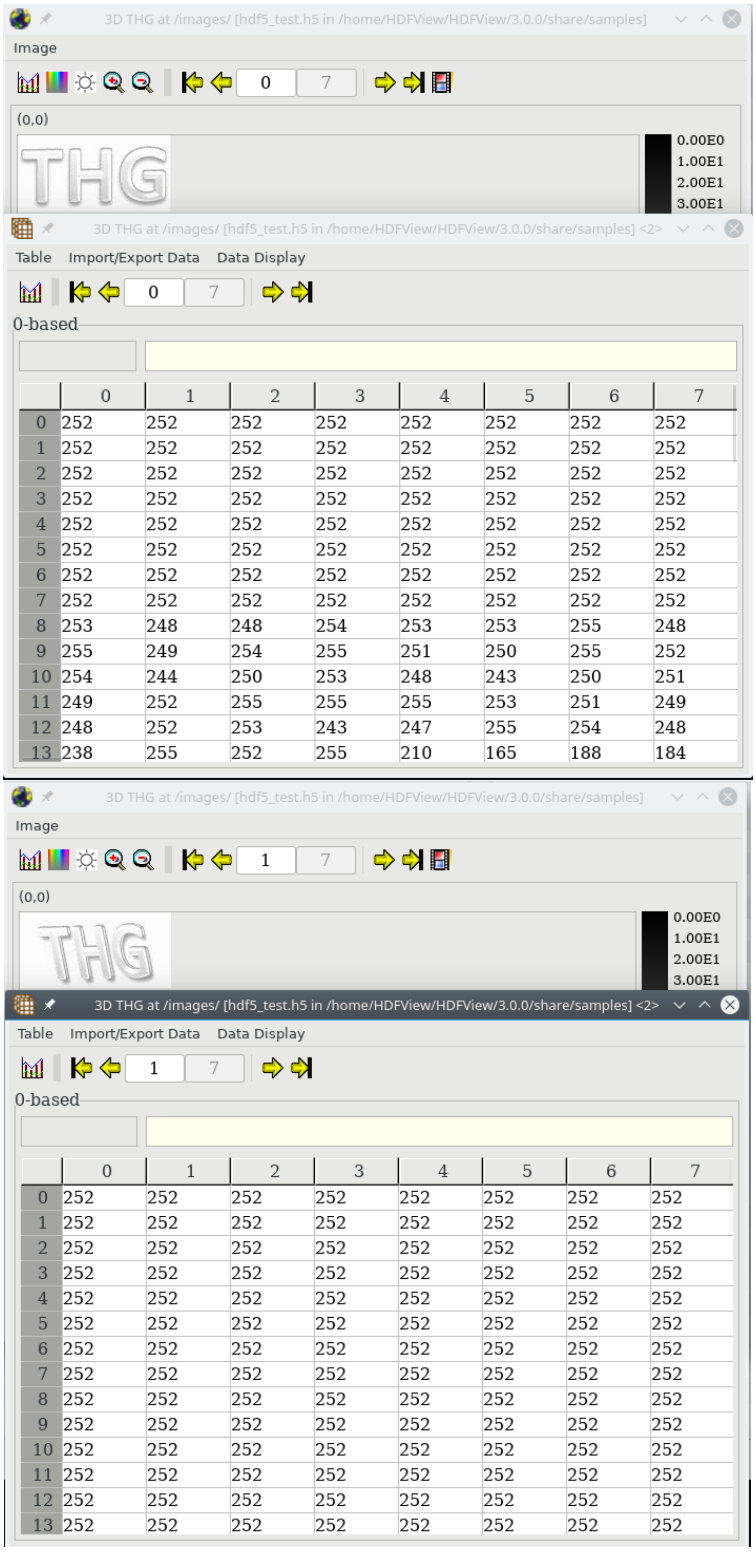
The following figure shows a contour image of level 9.




**Contour image (level 9)**

# 6.3 Animation

24-bit True color images of three dimensions have the option of being displayed as an animation. These "animated" images are represented the same way as a three-dimensional dataset is, with each "frame" of the total animation representing another "flip" along the third dimension of the associated dataset (see figure below). After opening such an image in the ImageViewer, you can step backwards and forwards through the individual "frames" of the image by using the first, previous, next and last buttons located at the top of the ImageViewer panel.




| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 1 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 2 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 3 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 4 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 5 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 6 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 7 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 8 | 253 | 248 | 248 | 254 | 253 | 253 | 255 | 248 |
| 9 | 255 | 249 | 254 | 255 | 251 | 250 | 255 | 252 |
| 10 | 254 | 244 | 250 | 253 | 248 | 243 | 250 | 251 |
| 11 | 249 | 252 | 255 | 255 | 255 | 253 | 251 | 249 |
| 12 | 248 | 252 | 253 | 243 | 247 | 255 | 254 | 248 |
| 13 | 238 | 255 | 252 | 255 | 210 | 165 | 188 | 184 |

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| 0 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 1 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 2 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 3 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 4 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 5 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 6 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 7 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 8 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 9 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 10 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 11 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 12 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |
| 13 | 252 | 252 | 252 | 252 | 252 | 252 | 252 | 252 |

**First page (frame) of image** **Second page (frame) of image**

To view an image as a smooth animation, first select the "Animation (frames/second)" option from the "Image" menu and select the rate at which you wish to view the animation (higher values provide a smoother animation). Then, either select the "Show Animation" option from the "Image" menu or press the animation button located next to the "last" button at the top of the ImageViewer.

## 6.4 View and Modify Image Palette/Values

A palette (or color lookup table) is the means by which color is applied to an image. It is a table in which every row contains the numerical representation of a particular color. In the example of an 8-bit standard RGB color model palette, this numerical representation of a color is presented as a triplet specifying the intensity of the red, green, and blue components that make up each color.

Although the HDF5 palette specification allows for variable color length, different look-up methods, and color models beyond RGB, HDFView only supports the indexed RGB color model of 256 colors. To view the image palette, click the palette icon on the tool bar or select the palette command from the image menu. The red, green, and blue components of the color table are plotted in a line plot.

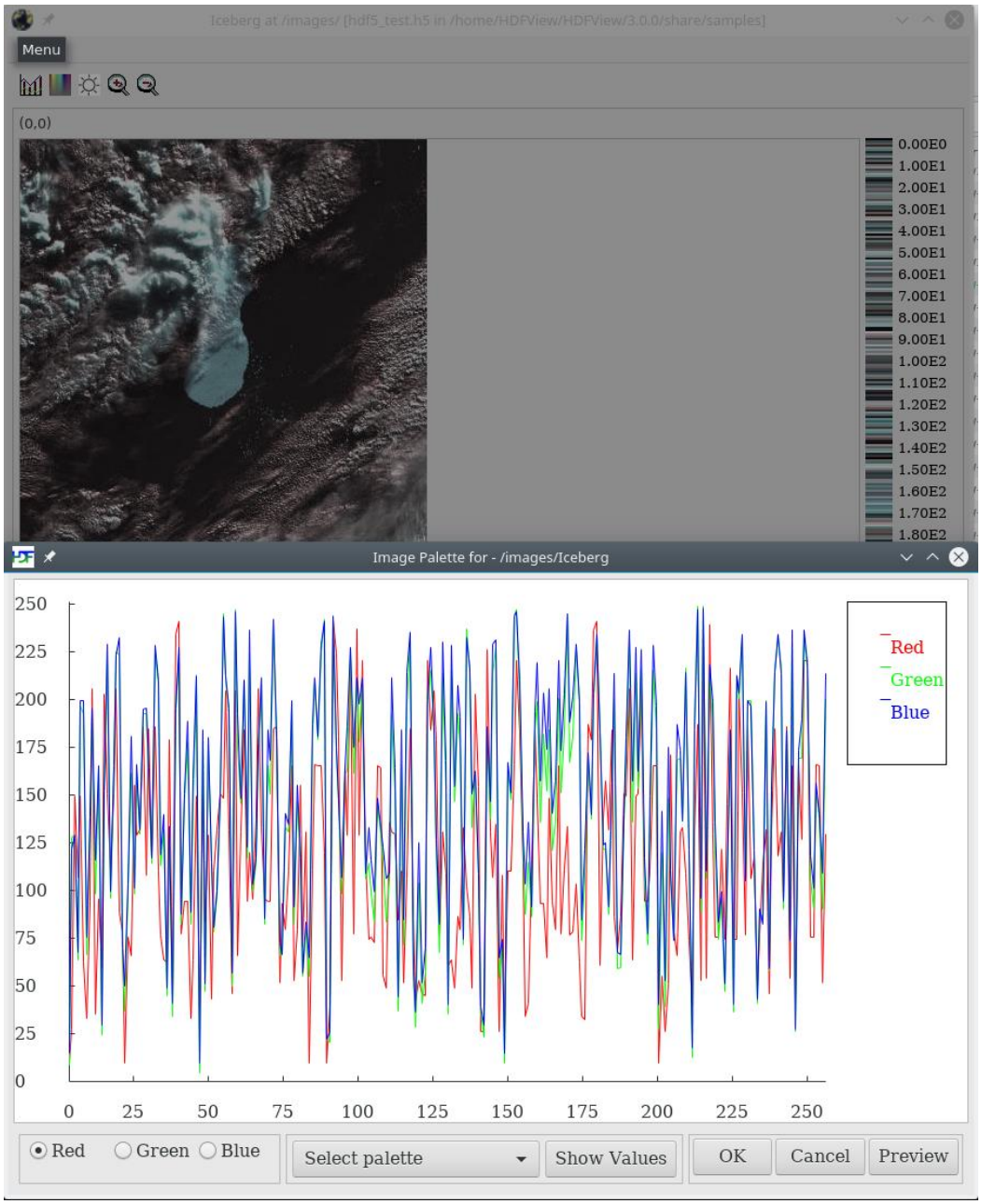
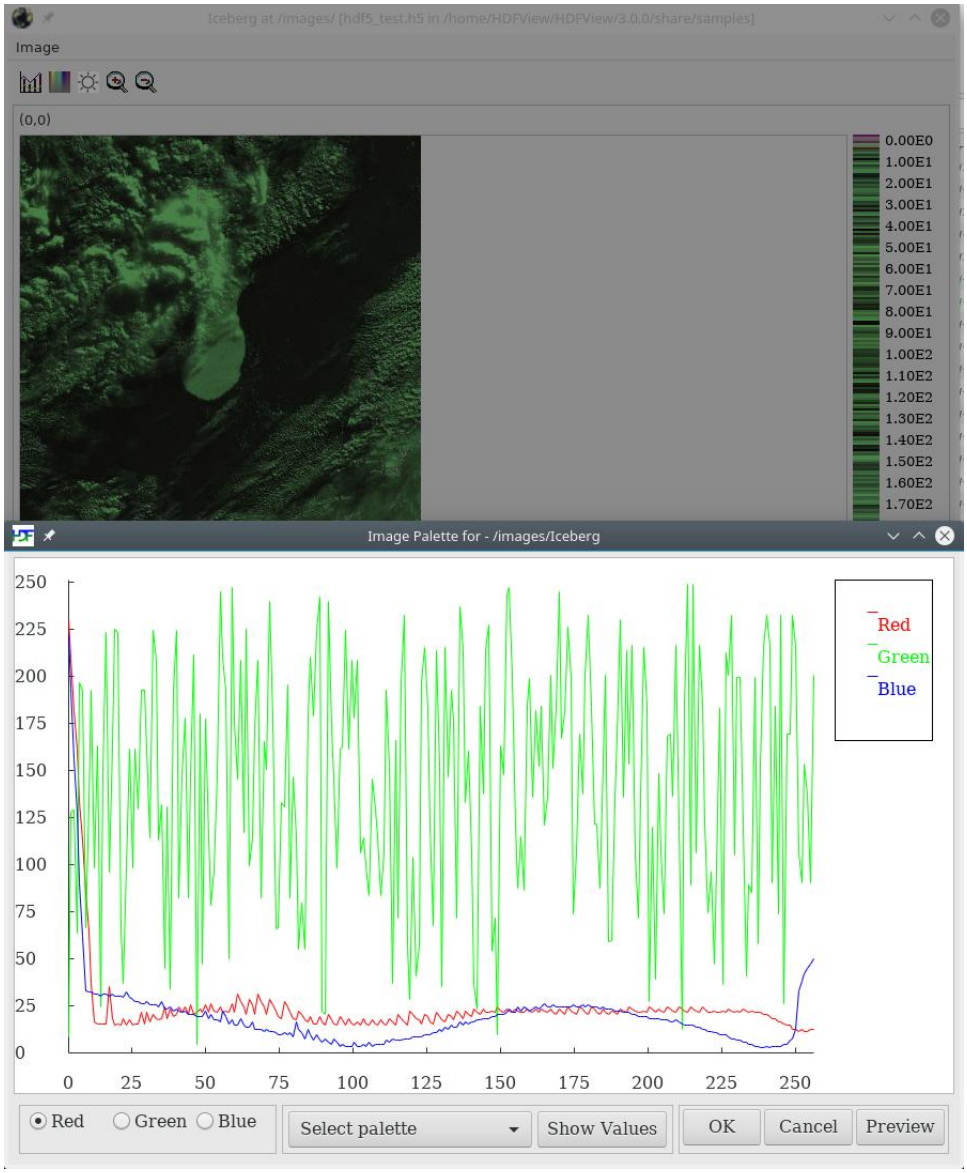



**Image palette (256 colors)**

To view the pixel values of each individual point, check the “Show value” item in the “Image” menu. When you move the mouse over the image, the pixel values of the mouse point are shown at the bottom of the image.

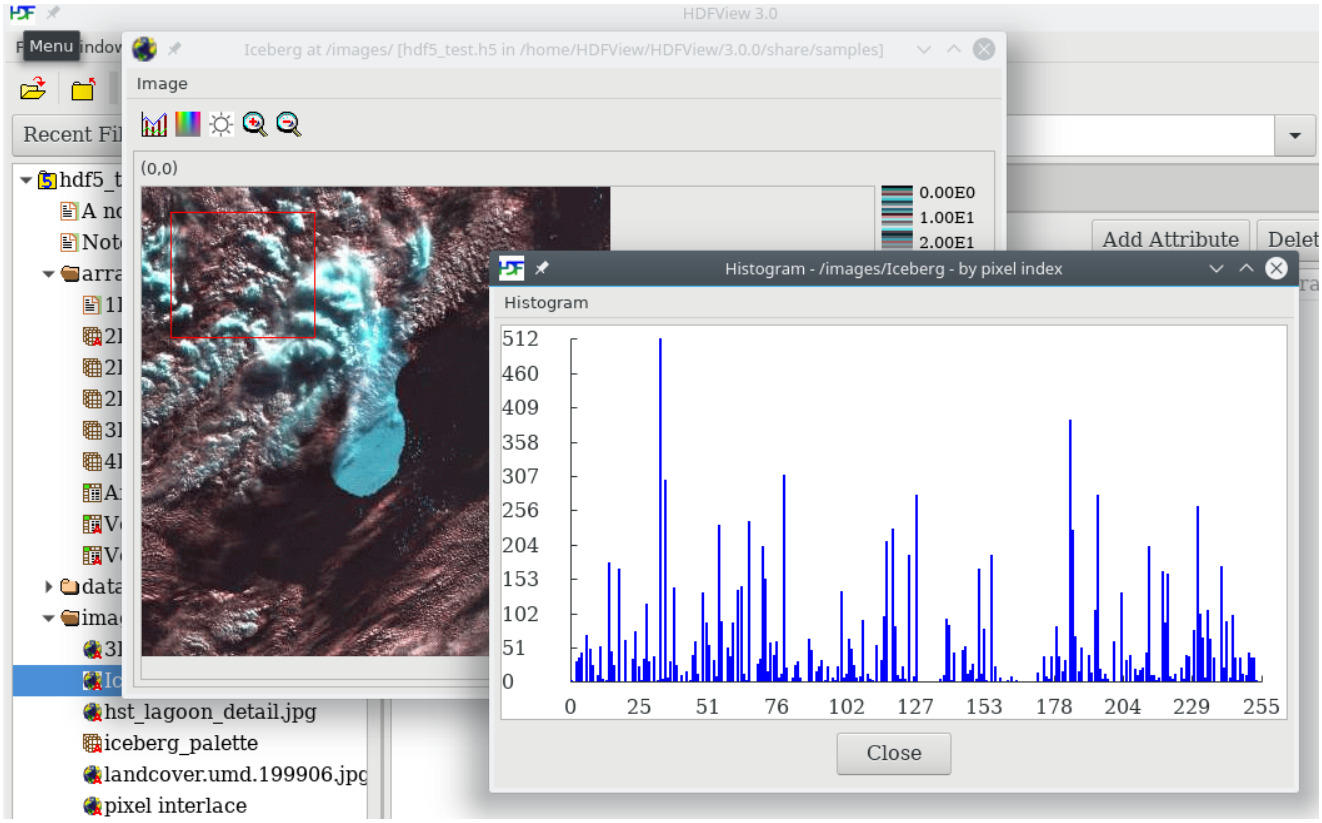
You can modify the values of the color table. Select the color (red, green, or blue) in the palette view and drag the line of the selected color. The value of the selected color changes as you move the color line. In the following figure, is the same image as above but with a modified color table.




**Modified image palette (256 colors)**

# 6.5 Show Histogram of Pixel Values

The frequency of pixel values of a selected area, or the whole image, can be displayed in a histogram chart. The horizontal axis of the histogram chart depicts the 256 pixel values. The vertical axis shows the frequency of the pixel values.




**Histogram of pixel values**

# 6.6 Import JPEG, GIF, PNG, or BMP Image to HDF4/5

Using HDFView, you can convert a JPEG, GIF, PNG, or BMP image into an HDF4 or HDF5 image. Select the "Convert Image To" command in the Tools menu, a popup window prompts you to choose the image file that you want to convert. The image is converted into a 24-bit HDF4 or HDF5 image. The current conversion does not support an image with indexed color model or an image with less than two color components. The image data is saved as an 8-bit unsigned integer regardless of the data type of the original image.

# 6.7 Save HDF Image to JPEG, GIF, PNG, or BMP File

Using the "Save Image As" command in the "Image" menu, you can save the current HDF image as a JPEG, GIF, PNG, or BMP file.

---

*[Table of Contents]*

# Chapter 7: User Options

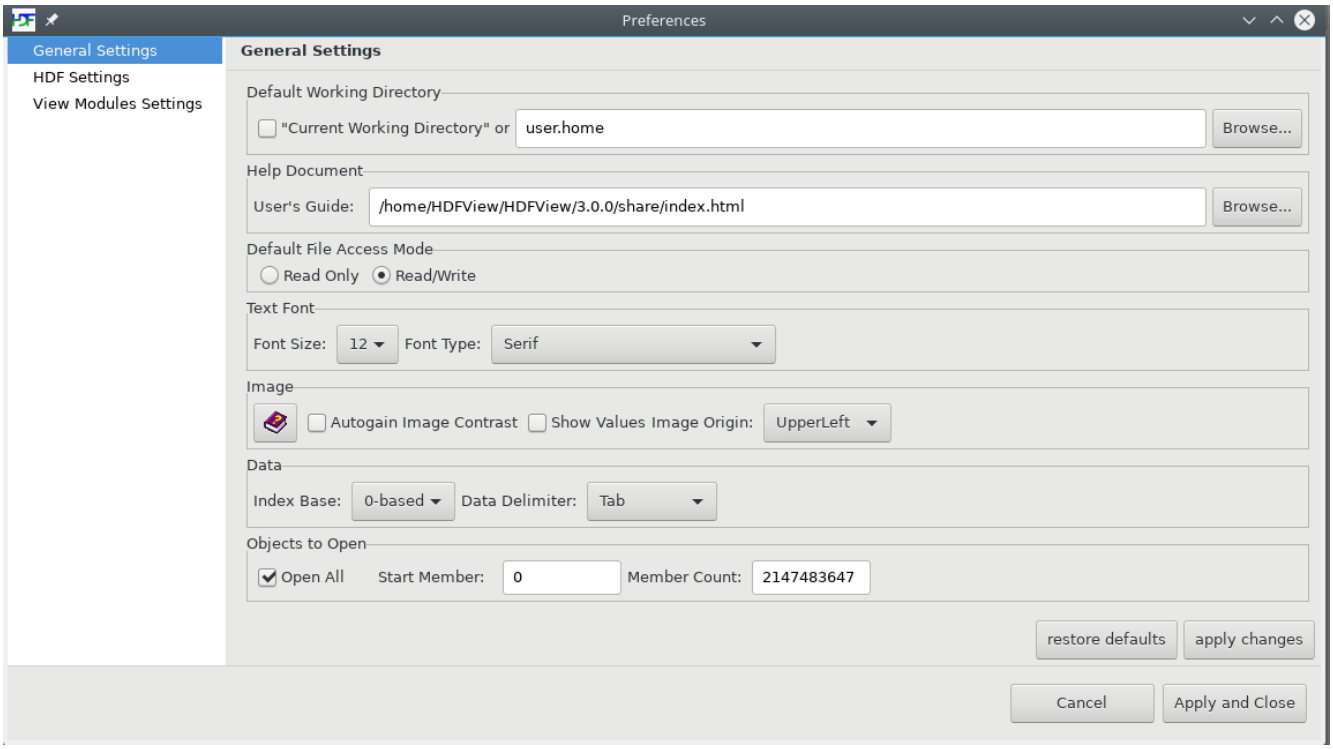
This chapter explains the different options of HDFView accessible from the Tools menu.



---

## 7.1 General Settings

HDFView comes with an array of configurable options that determine the way it operates on or displays data. These can be changed by clicking on "User Options" from the Tools menu. This section describes the general HDFView options.




**User Options General Settings**

Upon exit, HDFView saves all the current settings to a properties file in the user's home directory. The file's name follows the format '.hdfview{version number}' where version number is the current version number of HDFView.

### 7.1.1 Working Directory

By default, HDFView will open with the current directory being set to the path specified in the text field under this section. If "Current Working Directory" is selected instead, HDFView will open with the current directory set to the folder from which HDFView was launched. Setting the current directory affects where HDFView first opens to when opening/saving files, locating files, etc.

### 7.1.2 User's Guide Path

This section contains a single field which holds the path to the HDFView User's Guide. The path can either be entered as an absolute file name, file path, or URL to a remote file, or the browse button can be used to locate the guide from the filesystem. When the user clicks the  icon on the tool bar, the program goes to the path specified here and opens the User's Guide if it is able to locate it.

### 7.1.3 File Access Mode

By default, HDFView opens files in editing mode, allowing both reading and writing to files, as well as the ability to save changes. However, by changing the default mode to read only, the user can force HDFView to open all files as read-only, disabling editing and preventing changes to opened files.

### 7.1.4 Text Font

Under this section, the user can specify what size of font, as well as font type, to use. The newly selected font takes place immediately and affects all text displayed inside HDFView.

### 7.1.5 Image Options

The "Autogain Image Contrast" option here calculates a gain and bias combination that stretches images to match with the graphics system's pixel values. Further information is provided by the  icon in the User Options.

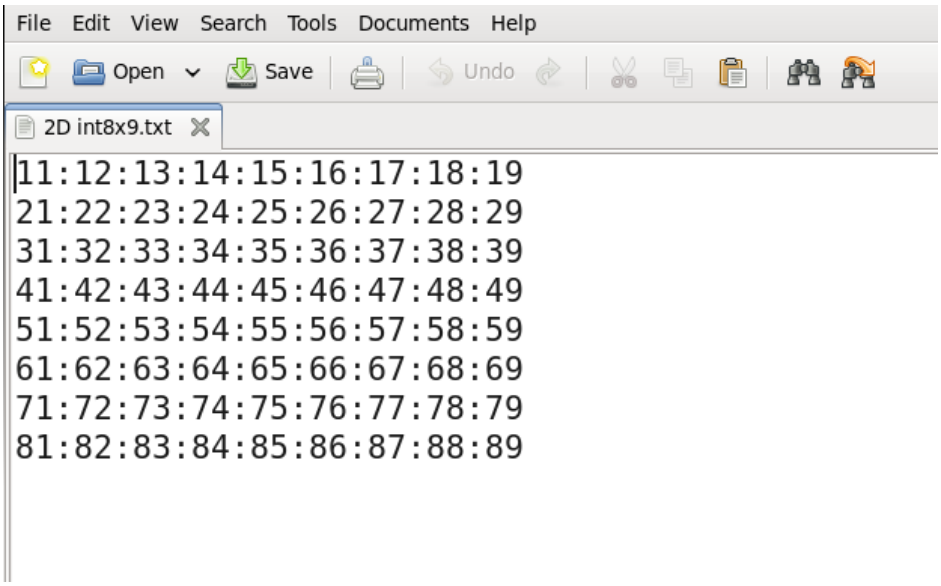
The "Show Values" option enables showing the values of individual pixels in an image. While this option is selected, moving the cursor over a pixel of an image inside the ImageViewer will display that pixel's location (in relative terms from the origin), as well as the value of the pixel.

The Image Origin dropbox in this section allows the user to select which of the four corners, Upper left, Upper right, Lower left or Lower right, to use as the origin for images.

### 7.1.6 Data Options

The Index Base determines whether indexed objects, such as numbered cells or the coordinates of an image's origin, should start with 0 or 1 as the first index.

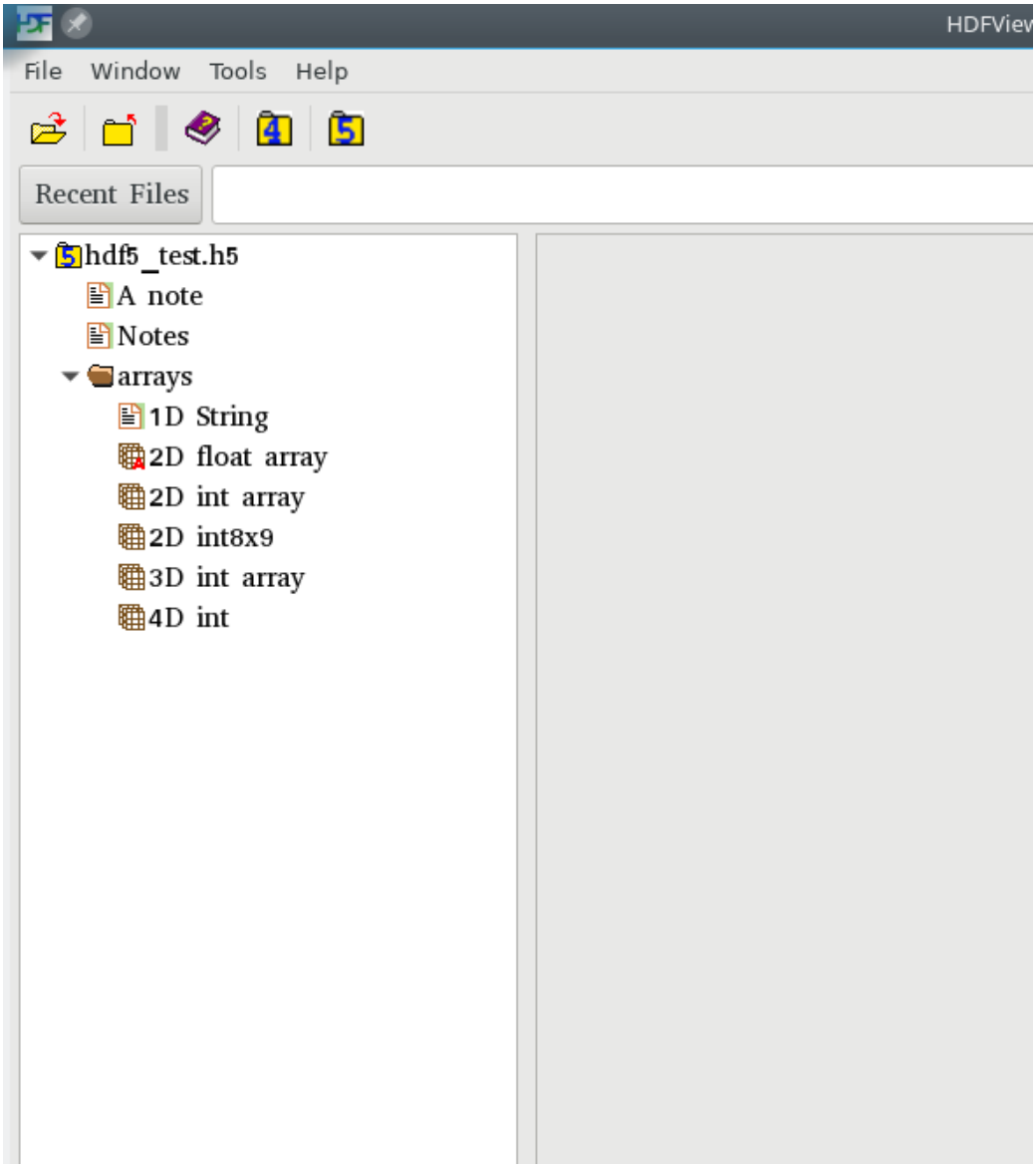
The Data Delimiter section allows the user to select a character to use as a separator for data when performing operations such as saving data to a file, importing data from a file or copying/pasting data between HDFView and other applications. The delimiter selected will be appended to data written to files, as shown below. **Note: When importing data from a file, the user must know the delimiter used for the file, else data may not be able to be imported properly**.




**Dataset saved to file with ':' delimiter**

### 7.1.7 Number of Open Objects

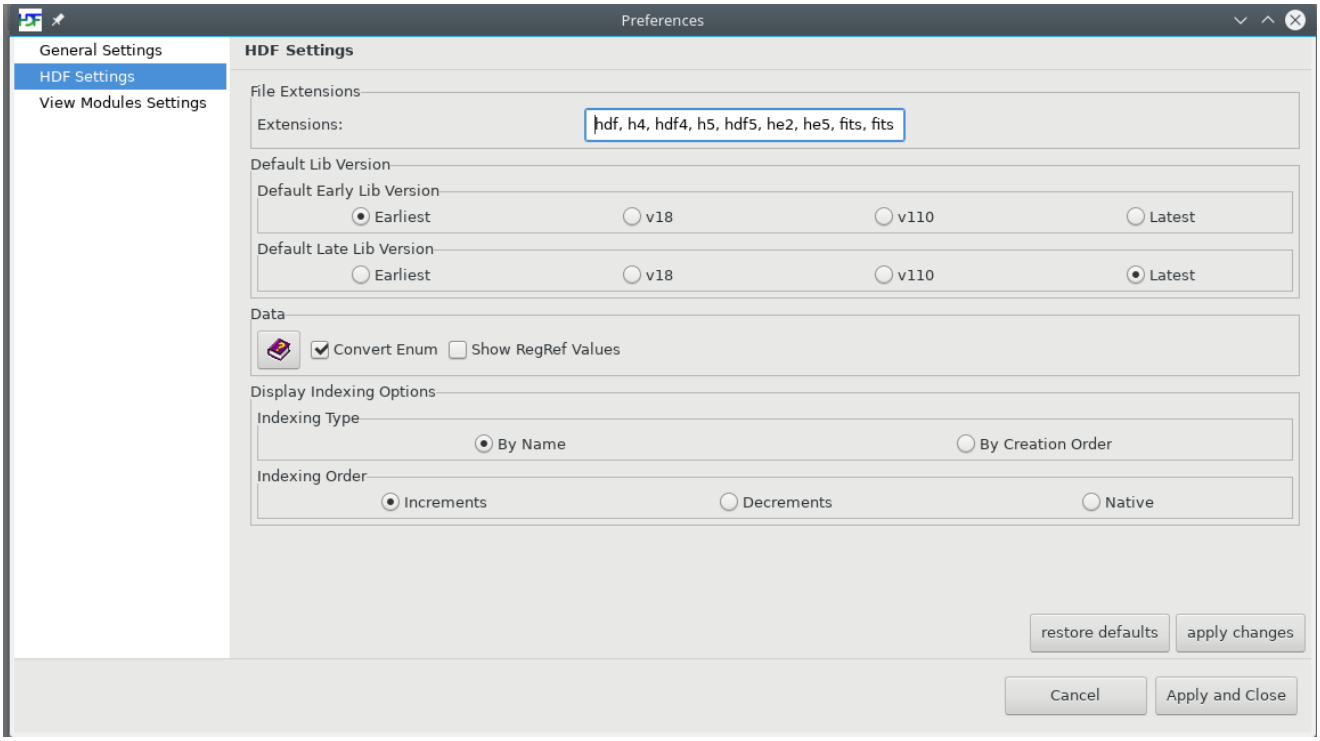
By default, HDFView loads all objects in a file upon loading the file. In some cases, however, it is useful to load a subset of the objects. To do so, first click on "Open All" in the "Objects to Open" section to unselect it, then pick a suitable start member (with member 0 being the root group) and the number of members to load in total. **Note: Groups count towards the member count, as does the root group**.




**Start Member = 0, Member Count = 10**

## 7.2 HDF Settings

HDFView comes with an array of configurable options that determine the way it operates on or displays data. These can be changed by clicking on "User Options" from the Tools menu. This section describes the HDF format specific options.




**User Options HDF Settings**

### 7.2.1 File Extensions

The text field located here displays a list of all file formats currently recognized by HDFView. This is used when choosing a file to open, as HDFView will look for and show files that have an extension matching any from this list. If needed, the user can type into this field and add their own file types for HDFView to look for.

### 7.2.2 Library Versions

Here the user can choose which version, the earliest version or latest version, of the HDF5 library to be used as the default when creating objects.

### 7.2.3 Data Options

The "Convert Enum" option enforces conversion of enum type data into a string equivalent. Further information is provided by the  icon next to the "Convert Enum" option.

"Show RegRef Values" allows the user to see the data pointed to by a Dataset Region Reference inline with the selected reference. If this option is not selected, the user must right click on a region reference and choose to display as a table or image in order to view the associated data.

### 7.2.4 Display Indexing

Display Indexing options change the order in which objects are displayed in the HDFView TreeView panel.

The Indexing Type determines the way in which the objects get sorted. Objects can either be sorted by name or by the order in which they were created.

Indexing Order further sorts the objects by ascending or descending order. If "Increments" is chosen, objects that would come first after sorting will be displayed at the top of the TreeView, with all following objects arranged below. If "Decrements" is chosen, these objects would instead be displayed at the bottom of the TreeView, with all following objects arranged above.

## 7.3 View Modules Settings

HDFView comes with configurable module settings that determine how the modules are displayed. These can be changed by clicking on "User Options" from the Tools menu, and selecting “View Modules Settings”.

### 7.3.1 Module Extensions

The fields located here allow other data display methods to be used for the standard views. A separate document explains the requirements for these modules.

---

*[Table of Contents]*