How is HDF5 different than HDF4

What are the Most Noticeable Differences between HDF4 and HDF5?

The HDF5 Format is completely different from HDF4. For most users it will never be necessary to know about the file format for either format.

Most users will work with the Data Model and APIs. The HDF4 and HDF5 Data Models and APIs are considerably different.

How is the HDF5 Data Model Different from HDF4?

The HDF4 Data Model has eight basic objects:

- A scientific dataset (SD), a multidimensional array with dimension scales
- An 8-bit raster image (DFR8), a 2-dimensional array of 8-bit pixels
- A 24-bit raster image (DF24), a 2-dimensional array of 24-bit pixels
- A general raster image (GR), a 2-dimensional array of multi-component pixels
- An 8-bit color lookup table or palette (DFP), a 256 by 3 array of 8 bit integers
- A table (Vdata), a sequence of records
- An annotation (AN), a stream of text that can be attached to any object
- A group (Vgroup), a structure for grouping objects

HDF5 includes two primary objects:

- A dataset, a multidimensional array of records
- A group, a structure for grouping objects

The HDF4 objects can all be conceptually mapped to an HDF5 Group or Dataset.

In addition to the fundamental data objects, HDF5 manages data spaces, data types, and attributes as objects in their own right. HDF4 uses these concepts, but has no object model that represents them.

The HDF5 data model is "simpler" in the sense that it has fewer objects and has a consistent object model throughout. In comparison, HDF4 has many more objects, and lacks a clear object model.

On the other hand, the HDF5 data model exposes many details. This makes the object model very powerful, but usually requires the user program to handle many routine details. In contrast, the HDF4 objects are simpler, requiring less programming to accomplish simple tasks.

How is the HDF5 API Different?

HDF4 has interfaces for the six types of objects supported, which allow a user program to perform common operations in a few calls. Equivalent operations may require many more calls to HDF5, even for very simple tasks.

The HDF5 high-level APIs provide simpler interfaces for common tasks.

What features does HDF4 have that HDF5 does not (yet) provide?

Aside from the general differences, HDF5 does not currently support these features of HDF4:

- Can I read/write netCDF with HDF5? No, but see the Converting HDF5 Files to NetCDF page for more information.
- Can I use JPEG, etc. Compression with HDF5?

  HDF5 currently supports only GZIP and SZIP compression. However, additional compression can be added easily.

What features does HDF5 have that HDF4 does not?

- **An HDF5 file has a true hierarchical file structure and a naming scheme for all the objects in the file (path names).**

  HDF5 uses a true hierarchical file structure, similar to the Unix file system. There is a "root group" and all objects belong to at least one group. In HDF5, every link has a name and objects are accessible by path names.

HDF4 uses a "pseudo-flat" file structure using Vgroups. There is no "root group" and objects do not have to belong to a group (Vgroup). Not all HDF4 objects have names.

- HDF5 has "Dataspace" objects, which are not in HDF4.

  In HDF5, a dataspace describes the positions of elements in a dataset. However, it is not the space allocated for a dataset.

  In HDF4, the software handled this, so the idea of a dataspace did not exist.

  To give more flexibility in defining objects in HDF5, the responsibility of creating a dataspace was given to the user. Therefore, whenever an object is created, a dataspace for the object has to be created first.

  Currently in HDF5, only simple dataspaces are supported. A simple dataspace is a regular N-dimensional array of data points. A *scalar* d ataspace is a special case of the simple dataspace, which is defined to be a 0-dimensional single data point in size. It is used for creating strings in HDF5.
- HDF5 has "Datatype" objects, which are different than HDF4 datatypes.

  In HDF 4, a datatype had a specific value always. In HDF5, a datatype is a datatype identifier created when the object was opened by HDF5. It can have a different value from one HDF5 session or platform to the next. To obtain information about a datatype a user can make H5T calls, passing in the datatype identifier as a parameter. New datatypes can be derived from existing types.

  In HDF5 you have "standard" pre-defined types. For example:

  ```
  H5T_IEEE_F64LE     Eight-byte, little endian, IEEE floating point
     H5T_STD_UI6BE      Two-byte, big endian, unsigned integer
  ```

  These datatypes have the same meaning on different platforms. You also have "native" pre-defined types. For example:

  ```
  H5T_NATIVE_INT   (int)
     H5T_NATIVE_FLOAT (float)
  ```

  These datatypes can be different on different platforms. For example, H5T_NATIVE_FLOAT might be 16-bit on one platform and 32-bit on another.
- HDF5 supports multiple storage models and it supports MPI-IO.

  HDF5 has an open interface to access raw storage. This enables HDF5 files to be written to a variety of media, including sequential files, families of files, memory, Unix sockets (i.e., a network).

  New "Virtual File" drivers can be added to support new storage access mechanisms.

  HDF5 also supports MPI-IO with Parallel HDF5. When building HDF5, parallel support is included by configuring with the --enable-parallel option. A tutorial for Parallel HDF5 is included with the HDF5 Tutorial.

How do I....?

...convert software that uses HDF4 to start using HDF5?

In general, it will be necessary to rewrite software that uses HDF4.

...use HDF5 data with the HDF4 library and vice-versa?

HDF5 data cannot be read or written by HDF4. HDF4 data cannot be read or written by HDF5.

...convert data from HDF5 to HDF4?

Some objects in an HDF5 file can be conceptually mapped to equivalent HDF4 objects. See the H4 to H5 Mapping Specification (pdf). However, HDF5 has many more possibilities than HDF4, so many HDF5 objects simply cannot be translated to HDF4. Also, some objects might be converted to more than one possible HDF4 object, depending on the intention of the creator.

The h5toh4 utility can convert some of the objects of an HDF5 file to a default HDF4 file.

...convert data from HDF4 to HDF5?

Almost all HDF4 objects can be conceptually mapped to equivalent HDF5 objects. See the H4 to H5 Mapping Specification (pdf). The h4toh5 utility converts all the objects in an HDF4 file to a default HDF5 file. The H4 to H5 library provides a conversion for individual objects.

It is important to realize, though, that a default conversion preserves the design of the HDF4 file, which may not be an optimal design for the resulting HDF5 file. The H4 to H5 library can help users convert HDF4 objects to create the HDF5 file they want.

...find out what is in an HDF5 file?

The h5dump utility lists the contents of an HDF5 file

...create HDF4 objects in HDF5?

You may want to review the H4 to H5 Mapping Specification (pdf) for detailed information about how to create HDF4 objects in HDF5. The following is generally how HDF4 objects get created in HDF5.

- **SDS**:

  A simple HDF5 dataset is comparable to an SDS.
- **Vgroups**:

  HDF5 Group objects are similar to Vgroups, and are used in exactly the same way.
- **Vdatas**:

  A compound datatype is typically used in instances where you want to obtain a "record" of information with differing datatypes, similar to a Vdata. The HDF5 Tutorial has an example of using compound datatypes. We also have an HDF5 High Level Interface for working with tables. See: H5TB: HDF5 Table Interface
- **Images (GR, DFR8, DF24)**:

  With HDF5 we have developed a "template" or specification for creating images. In general with HDF5, if a dataset attribute called "CLASS" exists with a value of "IMAGE", then the data will be interpeted as an image. There are many other attributes specified for clearly defining an image, as well. For details, refer to the Image and Palette Specification in the HDF5 Application Developer's Guide.
- **Palettes:**

  If a dataset attribute called "PALETTE" exists that is defined according to the Image and Palette Specification then the dataset will be interpreted as a palette.
- **Annotations:**

  In HDF5, attributes are used instead of annotations.

...create and use "String" data?

String attributes/datasets are created differently in HDF4 than they are in HDF5.

In HDF4, you typically used the DFNT_CHAR8 datatype to create the attribute/dataset and then it showed up as a string when viewed with a tool for reading HDF files. In HDF5, there is a similar datatype, H5T_NATIVE_CHAR, but if you use this then your data will NOT show up as a string with HDF5 tools. It will show up as individual characters. If you wish to use strings in HDF5, then you should use the H5T_C_S1 datatype, as follows (in C):

```
...
    hid_t       file_id, dataset_id, space_id, stype, attr_id, grp_id;
    herr_t      status;
    size_t      size;

    space_id = H5Screate (H5S_SCALAR);
    stype = H5Tcopy (H5T_C_S1);
    size = 21;                          /* Size of string for this example */
    status = H5Tset_size (stype, size);  /* Set the length of the string */

    /* Then use "stype" when calling H5Acreate or H5Dcreate.
       For a dataset it might look as follows:
         dataset_id = H5Dcreate (file_id, "Sdata", stype, space_id, H5P_DEFAULT, H5P_DEFAULT, H5P_DEFAULT);


       For an attribute it might look as follows:
         attr_id = H5Acreate (grp_id, "Adata", stype, space_id, H5P_DEFAULT);
    */
    ...
```

The HDF5 Examples by API on the HDF5 Examples page, contains examples of creating strings in HDF5. As an example, see the C example, h5ex_t_string.c, which creates a string dataset.

...create a Chunked/Compressed Dataset?

The storage properties, including chunking and compression, are controlled with an HDF5 Property List. See the HDF5 tutorial topic Creating an Extendible Dataset.

...read/write a sub-set or sub-sample of a Dataset

HDF5 supports a very flexible and general set of data selection features, which control both the source and destination. HDF5 supports:

- Hyperslabs, including repeated blocks as well as strides
- Unions of hyperslabs
- Sets of points

See the HDF5 Tutorial topic Reading From or Writing to a subset of a dataset.

...find the tag and ref of an HDF5 object?

HDF5 does not use tags and refs.

...locate and access specific objects in an HDF5 file?

You have to traverse all the objects in the file using H5Giterate. For any object for which the link count is > 1, record it in a table, using "objno" as a key (See H5Gget_objinfo). The first time you see it, there will be no entry in the table, and subsequent visits will find an entry.

If you need a table of contents for the whole file, then collect all the information you need, and use "objno" as a key to avoid loops.

In a future release of HDF5 we will be providing a function to return a table of contents, as well as functions to determine the number of members in a group and to return the name and type of the member of the group for a specific index value.

...write or access global attributes in HDF5?

You can write attributes to the root group, which would be "global" to all objects in the HDF5 file. To do this you must open the root group with *H5Gopen* and attach the attribute to this group. You cannot attach an attribute to the file, using the file identifier.