# How should dimension scales be handled in HDF5?

What should the model be?
How should it be implemented?

HDF Seminar

Mike Folk

September 26, 2004

# Earlier work

- HDF4 & netCDF
- Earlier proposals and discussions for HDF5
  1. “Dimension Scales in HDF5: Preliminary Ideas.” McGrath, May 2001. http://hdf.ncsa.uiuc.edu/RFC/ARCHIVE/DimScales/H5dimscales.htm.
  2. “Coordinate Systems in HDF5.” Slides. Koziol, March 2004.
  3. “Needed: A convenience API to Support Dimensions in HDF5.” McGrath, July 2001. http://hdf.ncsa.uiuc.edu/RFC/ARCHIVE/DimScales/H5dims.htm.
  4. “Should Dimension Scales be basic HDF5 constructs or higher level constructs?” Folk, May 2004. http://hdf.ncsa.uiuc.edu/RFC/ARCHIVE/DimScales/How_H5dimscales.htm.
- This proposal is an attempt to combine [1] and [2].
- Also draws from [3] and [4]

# Different approaches

**Proposal [1]**

- Stresses need for compatibility with HDF4 and netCDF
- Recommends no changes to HDF5 storage model or library
- Recommends design be similar to HDF4-to-HDF5 mapping
- Conceptual model
  - unspecific about role of dim scales and their relationships to one another
  - leaves these decisions up to the application, or TBD as the model is fleshed out.

**Proposal [2]**

- Covers wider range of apps, including coordinate systems.
- Recommends significant change to the HDF5 storage model
- Does not address library changes.
- Conceptual model
  - fairly specific about how dim scales are to be related to one another
  - proposes that coordinate system information be included.

# New proposal

- Tries to compromise between the various views
- Acknowledges group view that basic mechanisms be in format & library
- But also our view that the meaning of dim scales be left to apps, and that relationships between dim scales and datasets loosely specified

# At what level should dim scales be implemented?

- The most difficult issue

  Define and implement as high level objects

  versus

  Define and implement as part of the basic HDF5 data model, format, library

# Dim scales both app-specific and app-independent

- *specific* -- meaning derives from the scientific space to which they apply – for example, they help locate information in a geographic space.
- *independent* -- provide relevant information about a dataset's dataspace – for example, a dataspace includes information about each dimension

# Sharability also

- Apps might have different ideas about what that sharing means.
  - For instance, one dimension might map to the entire scale, and another to a subset of the scale.
- In HDF5, sharability means two or more HDF5 objects share another HDF5 object
  - E.g. a datatype or a dataspace, and the relationships are strictly defined in the HDF5 model.

# We can resolve the issue in four ways:

- Do nothing.
- Hybrid approach
  - implement some features within the HDF5 data model and library
  - *and* implement some features as a high level model and library.
- High level approach
  - implement completely as high level model and library
- Basic model approach
  - implement completely within the basic HDF5 data model and library

# *1. Do nothing.*

- We have ruled this out already.

# *2. Hybrid approach*

- Seemed to reflect the consensus of the group
- But I believe that it will be very hard to accommodate both views in an initial implementation
  - Could be very confusing to users
  - If implement a dim scale as a dataset, *and* scale info into the base library and format*, we are mixing high and low level information and functions*.
- We do this in HDF4 with Vdatas
  - On balance, not a great solution.
- Also as yet no other examples in which basic HDF5 library deals with high level objects.
- *Therefore, recommend that we choose one or the other*
- But there may be some exceptions to this

# Basic or high level: How to choose?

**Are dim scales as just another use of datasets?**

- If so, a high level library seems best,
- Advantages:
  - no changes to the format.
  - can treat the first implementation as a prototype, change later with no harm to the format or base library.
- Disadvantages
  - If we later change our mind, we're be stuck with legacy apps that depend on old approach, new applications that depend on the new approach
  - Leads to confusion and maintenance headaches (But could be mitigated by labeling the approach as a prototype.)
  - Requires resources to do a full implementation in the near future.

**Or fundamental components of datasets?**

- If so, seems best to change basic model and format
- Then the format will have to change,
  - E.g. include more dimension information in dataspaces
  - Possibly also in the headers of datasets that are dim scales
- Disadvantages
  - If we need to change later, the same disadvantages as do those of the higher level approach
  - Perhaps even harder to alter later on.

# My conclusion:

- I believe that our users are best served if we choose the first option – to think of dim scales as a high level use of datasets that have no special meaning in the HDF5 data model or library.

# **Proposal –** *Summary*

- Don’t include coordinate systems
- Store dim scales as datasets
  - Metadata indicating that they are to be treated as dim scales.
  - Each dim scale is to have an optional name.
  - Dim scales can be stored anywhere
  - Dim scale names need not be unique within a file.
- Datasets to be linked to dim scales
  - Each dim can optionally have one or more associated dim scales
  - Also can have local name for the dim scale
  - Dim scale sharable by two or more dimensions

# **Proposal –** *Summary*

- Relationships between dataset dims and their corresponding scales
  - not maintained or enforced by the library
    - e.g. don't automatically delete dim scale when all datasets are deleted that refer to it.
  - Store relationship info as an attribute in the dataset
- Functions proposed for dealing with dim scales
  - E.g. a function to convert a dataset to a dim scale
  - For the most part, these should be high level functions.
- Expand dataset model
  - Allow datasets to be represented by a generating function.
  - Enables dim scales to be represented as functions (a frequently requested feature)

# Detailed summary

# dim scales should have these properties

- Stored as HDF5 datasets
- No restriction on the size, shape, or datatype
- Read/write behavior same as for normal datasets
- Dim scales are public:
  - as visible within a file as other datasets
- Specifying a dataset is to be interpreted as a dim scale
  - Use HDF5 attribute CLASS
  - Similar to the way it is used with HDF5 images
- Can have at most one primary name
  - Store as an HDF5 attribute for the dim scale dataset
- Dim scale names not required to be unique within file

# New properties for datasets

- Each dimension may have any number of corresponding dim scales.
  - Identify multiple scales by index values
- Each dimension may have any number of names
  - Even if there is no corresponding dim scale
- How to associate dimension names and scales
  - “dimension attribute records”
  - One or more for each dimension
  - Contents of dimension attribute records:
    - Index value
    - Dimension name (optional)
    - Object reference to dim scale (optional)
- We recommend storing this information as attributes, but we can see benefits in storing it in the dataspace header.

# Relationships between dims and dim scales.

- Two or more dimensions can share the same scale
  - I.e. a dim scale may be associated with more than one dimension in more than one dataset.
- Relationships between dim scales and dataset dimensions not maintained or enforced by library.
  - When a dataset dimension extended, the library is *not* required to automatically extend any corresponding dim scales
  - The library does not maintain information about dimensions that are shared by more than one dataset.
  - Dim scales not automatically deleted when datasets deleted
  - Library does not enforce, nor require, that a dimension name in dimension attribute record be the same as name in the scale

# New functions for dealing with dim scales

- Convert dataset to scale (D, name)
  - convert dataset D to a dim scale.
- Attach scale (D, S, i)
  - attach dim scale S to the ith dim of D.
- Detach scale (D, i, j)
  - detach the jth scale from the ith dim of D, and decrement the dim scale count for D.
- Get number of scales (D, i)
  - get the number of scales associated with the ith dim of D.
- Get ID of scale (D, i, j)
  - get the ID for the jth scale associated with the ith dim dataset D.
- Get scale info (S)
  - get info about dim scale S (existence, size, name, etc.)

# Expanding raw data options

- Extend the dataset model to allow a new storage option for datasets whereby datasets can be represented by a function, or formula.
- Study the possibility of allowing formulas to be used for attributes.

# Formula datasets

- Formula dataset: dataset represented by function of the indices.
- *Recommendation: extend the dataset model to allow a new storage option for datasets whereby datasets can be represented by a function, or formula.*
- Further study would seem to be advisable before proceeding with this option.
- Formula *attributes?*
  - Further study