



Parallel IO Libraries

Steve Heistand

NAS Division

NASA Ames Research Center

2012 Summer Short Course for Earth System
Modeling and Supercomputing

What is parallel IO and why should I use it



- The most basic definition of parallel IO is simply any and all file IO done by an application that is running multiple processes at once.
 - could be 1 process acting on 1 to M # of files
 - ... N processes acting on a single file
 - ... N processes acting on M files.
- Note that a serial application operating on a file in a parallel filesystem (eg Lustre) isn't quite the same thing.
- Why: Speed

What is parallel IO and why should I use it



- Options for parallel IO:
 - MPI-IO
 - HDF5
 - (parallel) NetCDF
- Which library and how you use it in most cases is decided by factors unrelated to the code itself.
 - historical reasons
 - due to what post processing tool is used
 - which method is fastest on the machine at hand
- Please think carefully before making up a new IO library

MPI-IO



- MPI-IO provides a low-level interface to carrying out parallel I/O
- With MPI-IO the user defines how to access the data in a file
 - There is no metadata stored about the file
 - There are no tools to analyse what kind of data is stored in the file
- The MPI-IO API has a large number of routines
 - The MPI 2.2 standard has 64 pages for the I/O section



MPI-IO (continued)

- The MPI-IO standard includes the concept of multiple ranks all operating on the same file in a safe and consistent manner.
- It's very similar to normal posix file operations but with the prefix `MPI_` in the routine name.
 - `MPI_FILE_OPEN()`
 - `MPI_SHARED_FILE_WRITE()`
 - `MPI_FILE_CLOSE()`
- The user must take care of any inherent structure the file needs to have in terms of data layout on their own.



MPI-IO (continued)

- MPI_File_open is a collective call to open a file
 - The collective is defined on the communicator
 - Typically you might use MPI_COMM_WORLD as a communicator
 - To open a file just on one process use MPI_COMM_SELF as the communicator
 - All processes must call MPI_File_open with the same filename and access mode parameters.

- There are many MPI-IO routines that are collective calls, a common mistake is to call a routine either by not everyone in the communicator or with different arguments on different processes.



- **MPI-IO CODE**

HDF5



- HDF5 is a completely new Hierarchical Data Format product consisting of a data format specification and a supporting library implementation.
- HDF5 is designed to address some of the limitations of the older HDF product and to address current and anticipated requirements of modern systems and applications.
- HDF5 files are organized in a hierarchical structure, with two primary structures: *groups* and *datasets*.



HDF5 (continued)

- *HDF5 group*
 - a grouping structure containing instances of zero or more groups or datasets, together with supporting metadata.

- *HDF5 dataset*
 - A multidimensional array of data elements, together with supporting metadata.

- Working with group members is similar in many ways to working with file structures in UNIX. As with UNIX directories and files, objects in an HDF5 file are often described by giving their full path names.

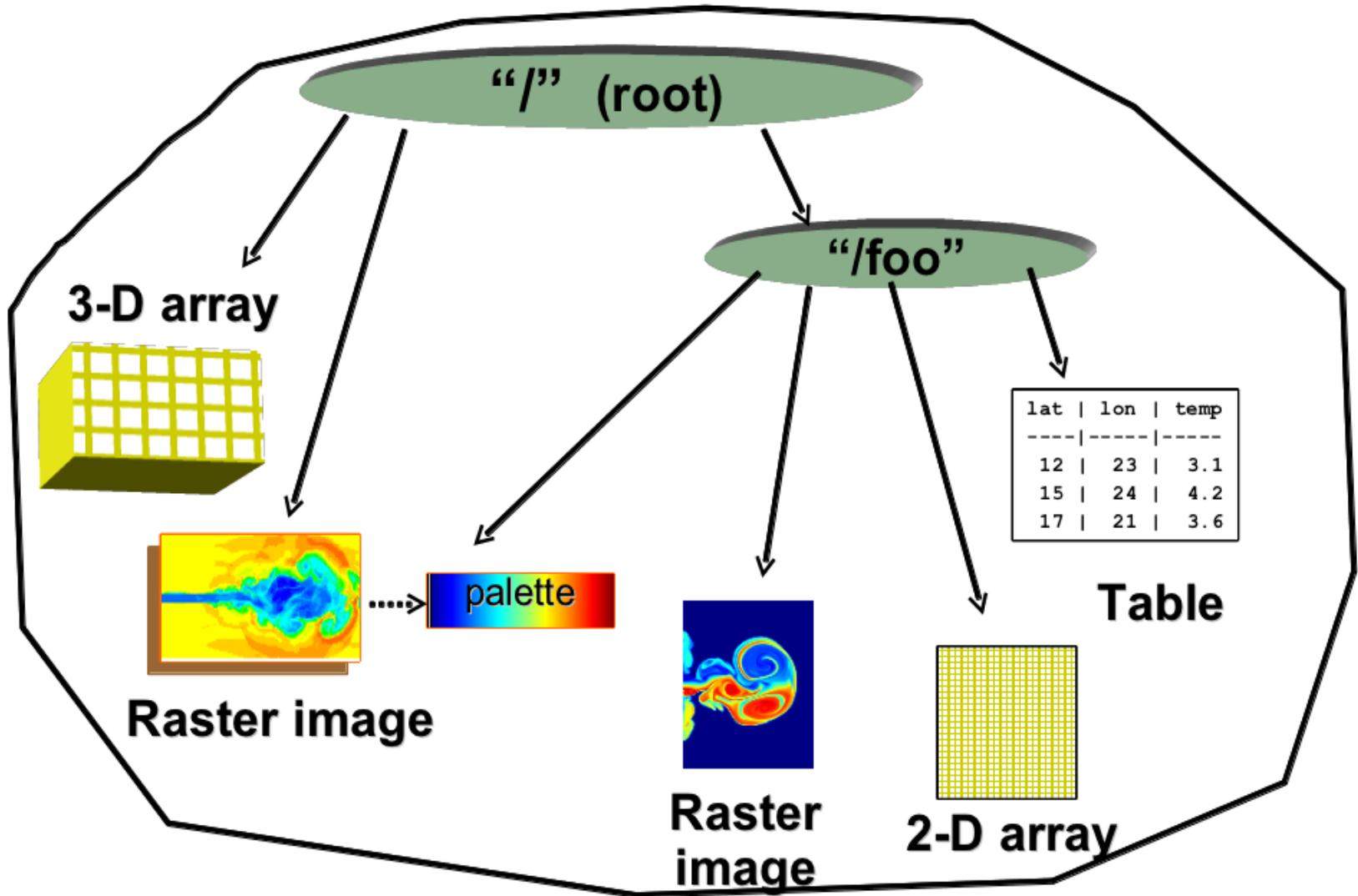


HDF5 (continued)

- / signifies the root group.
- /foo signifies a member of the root group called foo.
- /foo/zoo signifies a member of the group foo, which in turn is a member of the root group.

- Any HDF5 group or dataset may have an associated *attribute list*. An HDF5 *attribute* is a user-defined HDF5 structure that provides extra information about an HDF5 object.

HDF5 (continued)





- **HDF5 CODE**

NetCDF



- NetCDF is a set of data formats, programming interfaces, and software libraries that help read and write scientific data files. NetCDF was developed and is maintained at Unidata, part of the University Corporation for Atmospheric Research (UCAR) Office of Programs (UOP). Unidata is funded primarily by the National Science Foundation.
- The classic netCDF data model consists of variables, dimensions, and attributes. This way of thinking about data was introduced with the very first netCDF release, and is still the core of all netCDF files.

NetCDF (continued)



- Variables
 - N-dimensional arrays of data. Variables in netCDF files can be of type char or string, 8/16/32/64 bit signed or unsigned integers and 32/64 bit floating point.
- Dimensions describe the axes of the data arrays.
 - A dimension has a name and a length.
 - An unlimited dimension has a length that can be expanded at any time, as more data are written to it.
- NetCDF files can contain at most one unlimited dimension.

NetCDF (continued)



- **Attributes**

- annotate variables or files with small notes or supplementary metadata.
- Attributes are always scalar values or 1D arrays, which can be associated with either a variable or the file as a whole.
- Although there is no enforced limit, the user is expected to keep attributes small.



Meteorological Example

- NetCDF can be used to store many kinds of data, but it was originally developed for the Earth Sciences community.
- NetCDF views the world of scientific data in the same way that an atmospheric scientist might: as sets of related arrays. There are various physical quantities (such as pressure and temperature) located at points at a particular latitude, longitude, vertical level, and time.

NetCDF (continued)



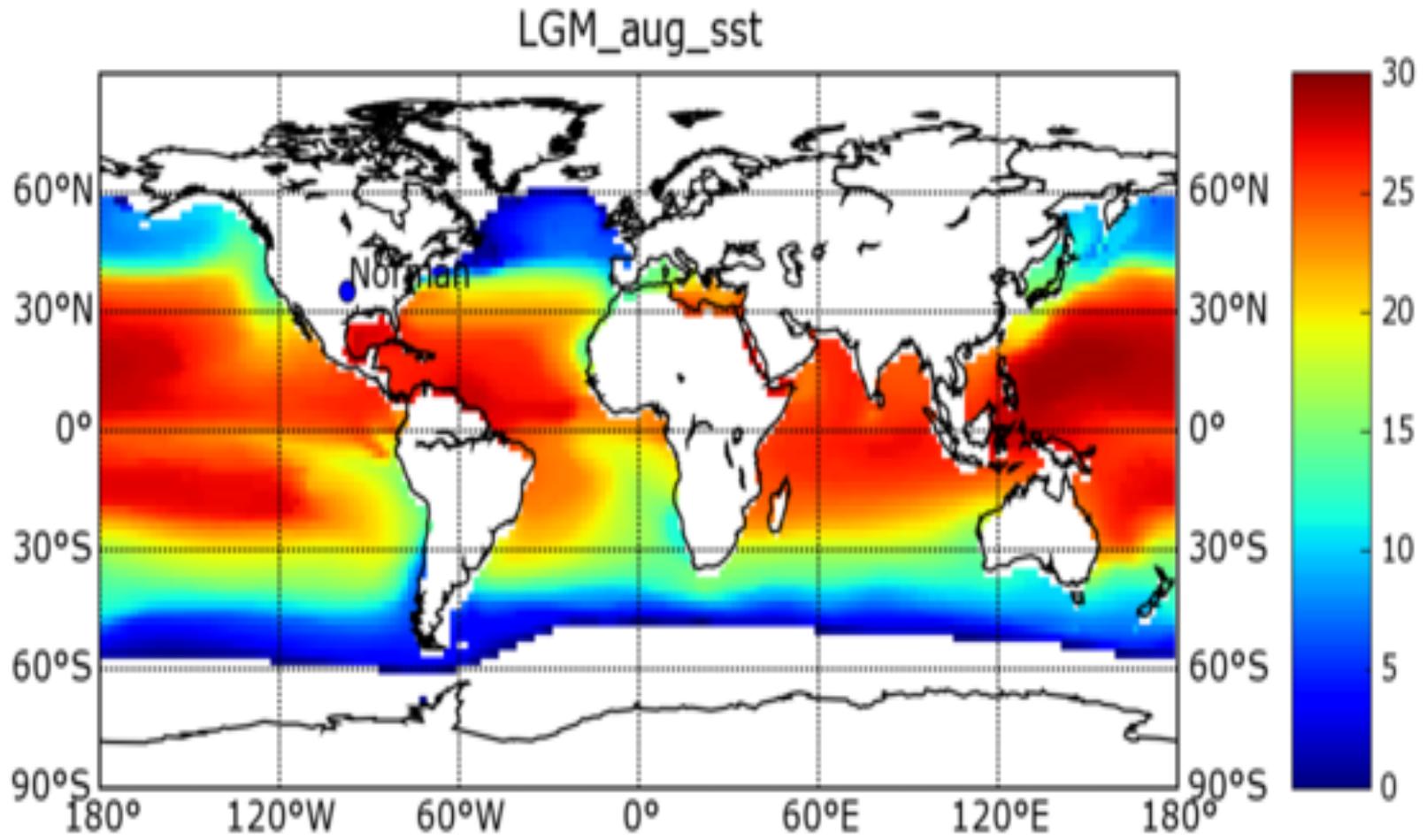
- A scientist might also like to store supporting information, such as the units, or some information about how the data was produced.
- The axis information (latitude, longitude, level, and time) would be stored as netCDF dimensions. Dimensions have a length and a name.
- The physical quantities (pressure, temperature) would be stored as netCDF variables. Variables are N-dimensional arrays of data, with a name and an associated set of netCDF dimensions.

NetCDF (continued)



- It is also customary to add one variable for each dimension, to hold the values along that axis.
- These variables are called “coordinate variables.” The latitude coordinate variable would be a one-dimensional variable (with latitude as its dimension), and it would hold the latitude values at each point along the axis.
- The additional bits of metadata would be stored as netCDF attributes.
- Attributes are always single values or one-dimensional arrays. (eg. a character string)

NetCDF (continued)



NetCDF (continued)



- Here is a typical sequence of netCDF calls used to create a new dataset:
 - `nc_create` /* create dataset: enter define mode */
 - `nc_def_dim` /* define dimensions: name and length */
 - `nc_def_var` /* define variables: from name, type, ... */
 - `nc_put_att` /* put attribute: assign attribute values */
 - `nc_enddef` /* end definitions: leave define mode */
 - `nc_put_var` /* provide values for variables */
 - `nc_close` /* close: save new netCDF dataset */



- **NetCDF CODE**



Final Thoughts

- Modules on Pleiades:
 - MPI-IO: mpi-sgi/mpt.2.06r6
 - HDF5: hdf5/1.8.7/gcc/mpt
 - NetCDF: netcdf/4.1.3/gcc/mpt

pfe: module load mpi-sgi/mpt.2.06r6 netcdf/4.1.3/gcc/mpt hdf5/1.8.7/gcc/mpt

- Questions?