EOSDIS
NASA'S EARTH OBSERVING SYSTEM
DATA AND INFORMATION SYSTEM

OPeNDAP[1] and HDF5[2] in the Cloud: Techniques and Best Practices for Serving HDF5 data

Winter ESIP 2019

Elena Pourmal
Technical Project Lead
epourmal@hdfgroup.org

James Gallagher
Technical Project Lead
jgallagher@opendap.org

This work was supported by NASA/GSFC under Raytheon Co. contract number NNG15HZ39C.
This document does not contain technology or Technical Data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

1 Open-source Project for a Network Data Access Protocol
2 Hierarchical Data Format 5

Outline

- Considerations for HDF5 files
  - HDF5 and netCDF-4 interoperability
  - CF[1] conventions
  - HDF5 best practices
    - General considerations
    - Advanced HDF5 features
- Considerations for Amazon S3[2] access via OPeNDAP
  - Reduce initial latency
  - Optimize transfer time
  - Reduce post-transfer processing

1 Climate Forecast Conventions
2 Amazon Simple Storage Service

Considerations for HDF5 files

HDF5/ netCDF-4[1] Interoperability

- To assure maximum interoperability with netCDF-4 use the following:
  - Hierarchical HDF5 file (no loops and multiple parents in a file structure)
  - Creation order on links and attributes
  - Simple numeric datatypes
  - Gzip compression
  - Dataset with dimension scale attached to each dimension

[1] Network Common Data Form, version 4

HDF5/ netCDF-4 Interoperability (cont’d)

- Attributes
  - Scalar or one-dimensional attributes
  - Avoid using these datatypes:
    - Fixed size strings
    - Compound datatype
    - Long double
  - Use UTF-8[1] encoding for names encoding
  - Attribute name should begin with a letter and be composed of letters, digits, and underscores
  - See Unidata documentation

[1] Unicode Transformation Format 8-bit

HDF5/ netCDF-4 Interoperability (cont'd)

- How to achieve netCDF-4 compatibility?
  - Use netCDF-4 library
  - Use augmentation and editing tools
    - Tools modify the existing HDF5 file!

Augmentation and Editing Tools

- Use tools to make HDF5 file netCDF-4 compatible
  - HDF-EOS5[1] augmentation tool
    - Adds dimension scales
  - HDF5 JPSS[2] augmentation tool
    - Hides unsupported data (region references)
    - Adds dimension scales
    - Adds geolocation information
  - HDF5 editing tool (attributes only)
    - Add/modify attributes to make them CF compliant

[1] Hierarchical Data Format - Earth Observing System library and file format, version 5

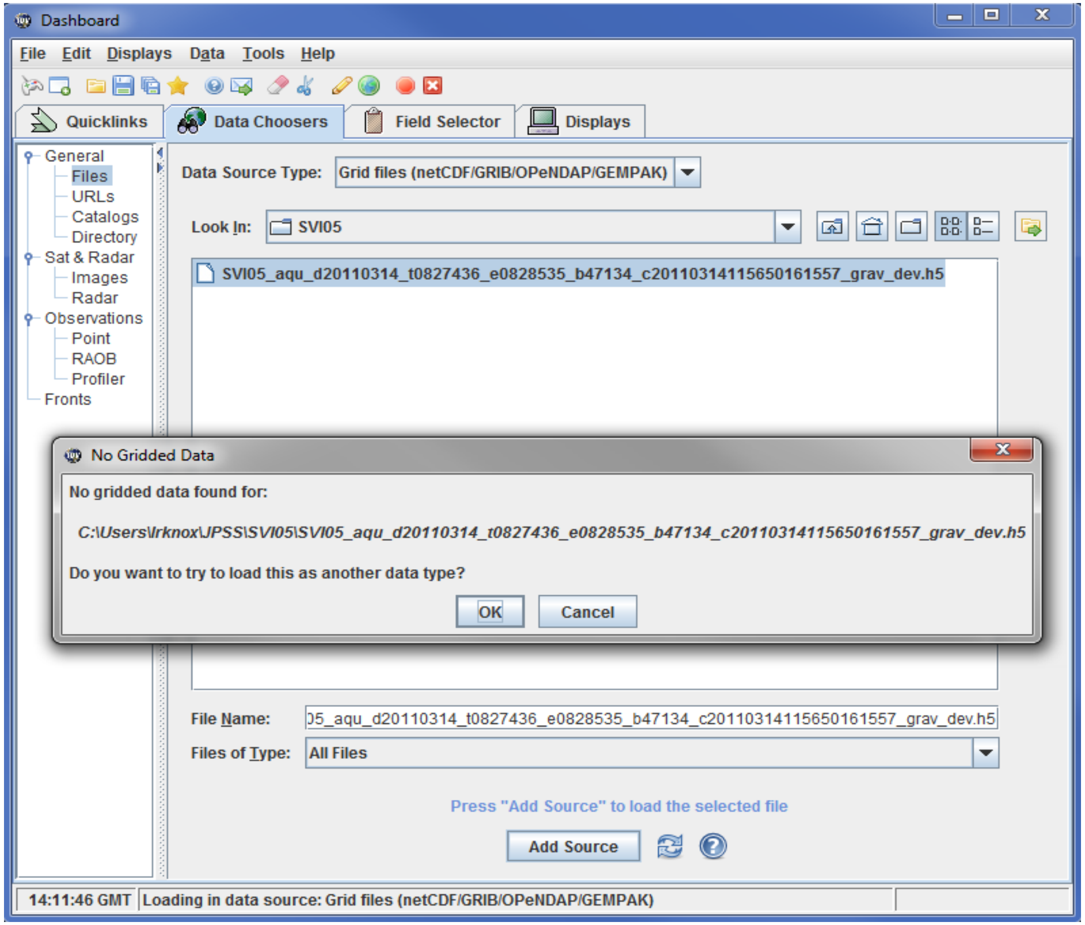
[2] Joint Polar Satellite System

Climate and Forecast (CF) Metadata

- HDF5 attributes
  - long_name
  - units
  - coordinates
  - scale_factor
- To learn more
  - ESDS-RFC-028v1.3 *“Data Interoperability Recommendations for Earth Science”*, Charles S. Zender, Peter J.T. Leonard, July 2016
  - 423-ESO-036 “Dataset Interoperability Recommendations: Part 2” (DRAFT)

Example: Visualizing JPSS data with IDV[1]

Before



Data Source Type: Grid files (netCDF/GRIB/OPeNDAP/GEMPAK)

Look In: SVI05

SVI05_aqu_d20110314_t0827436_e0828535_b47134_c20110314115650161557_grav_dev.h5

No gridded data found for:

C:\Users\lrknox\JPSS\SVI05\SVI05_aqu_d20110314_t0827436_e0828535_b47134_c20110314115650161557_grav_dev.h5

Do you want to try to load this as another data type?

File Name: 05_aqu_d20110314_t0827436_e0828535_b47134_c20110314115650161557_grav_dev.h5

Files of Type: All Files

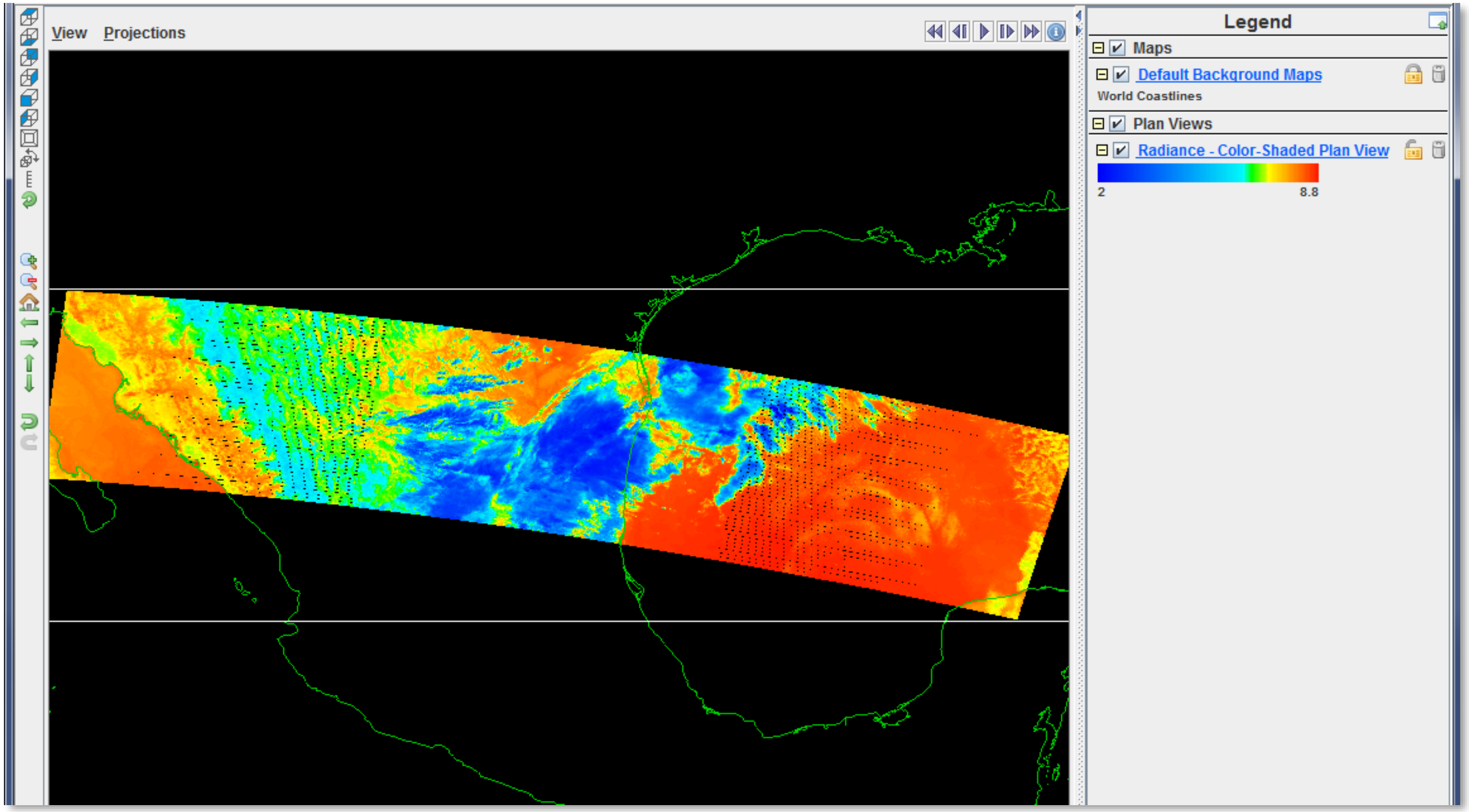
Press "Add Source" to load the selected file

14:11:46 GMT | Loading in data source: Grid files (netCDF/GRIB/OPeNDAP/GEMPAK)

EOSDIS

[1] Integrated Data Viewer

Example: Visualizing JPSS data with IDV

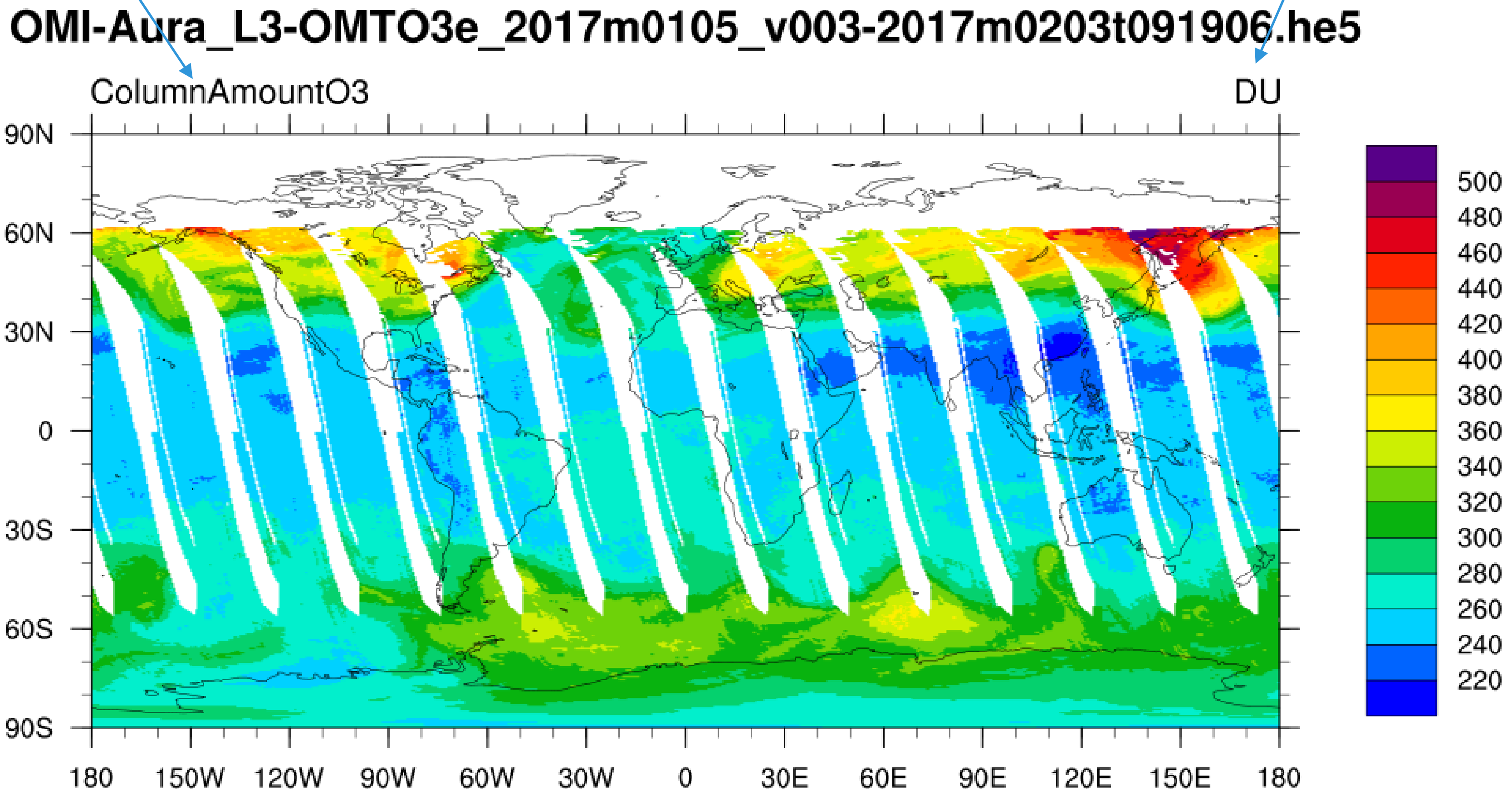


After augmentation and adding CF `scale_factor` and `units` attributes

Example: CF attributes

long_name

units



HDF5 Best Practices

- I*f possible, use the latest HDF5 library releases*
  - Bug fixes
  - Security updated
  - Storage, memory usage and performance optimizations
- Check new features and read how to move to the latest releases without breaking your code.

HDF5 Best Practices

- *Many optimizations are not available by default*
  - To reduce storage size for HDF5 files with many objects and allow better access time set file format to HDF5 1.8.
  - To reduce access time and storage overhead for chunked datasets set file format to HDF5 1.10.
    - Constant look-up speed for chunked datasets with fixed size dimensions

HDF5 Best Practices (cont’d)

- Avoid data duplication by using
  - HDF5 virtual datasets
    - Aggregate data stored in different arrays (HDF5 datasets)
  - Shared attributes, datatypes, dimensional and compression information
    - HDF5 objects can share attributes and other characteristics saving space in the file
  - Contact help@hdfgroup.org to help you with organizing data in HDF5

Considerations for S3 access via OPeNDAP

Hyrax in the Cloud

Hyrax now serves data from S3

- HDF5 data files are subset directly
  - Only those values needed are read
  - This applies to netCDF4, too
  - Extendable to other formats, but…
- Other file types are read from S3 and cached on spinning disk
- Metadata are cached on spinning disk

Considerations for S3 access via OPeNDAP

- Reduce initial latency
- Optimize transfer time
- Reduce post-transfer processing

Reduce Initial Latency

- Reuse connections to S3
- S3 is accessed using HTTP[1]
  - But HTTP uses TCP socket connections
- Open HTTP connections before they are needed
- Re-Open connections that time-out
- Use HTTP 2!
- *This has nothing to do with HDF5 or legacy data. All software that make repeated use of S3 should do this*

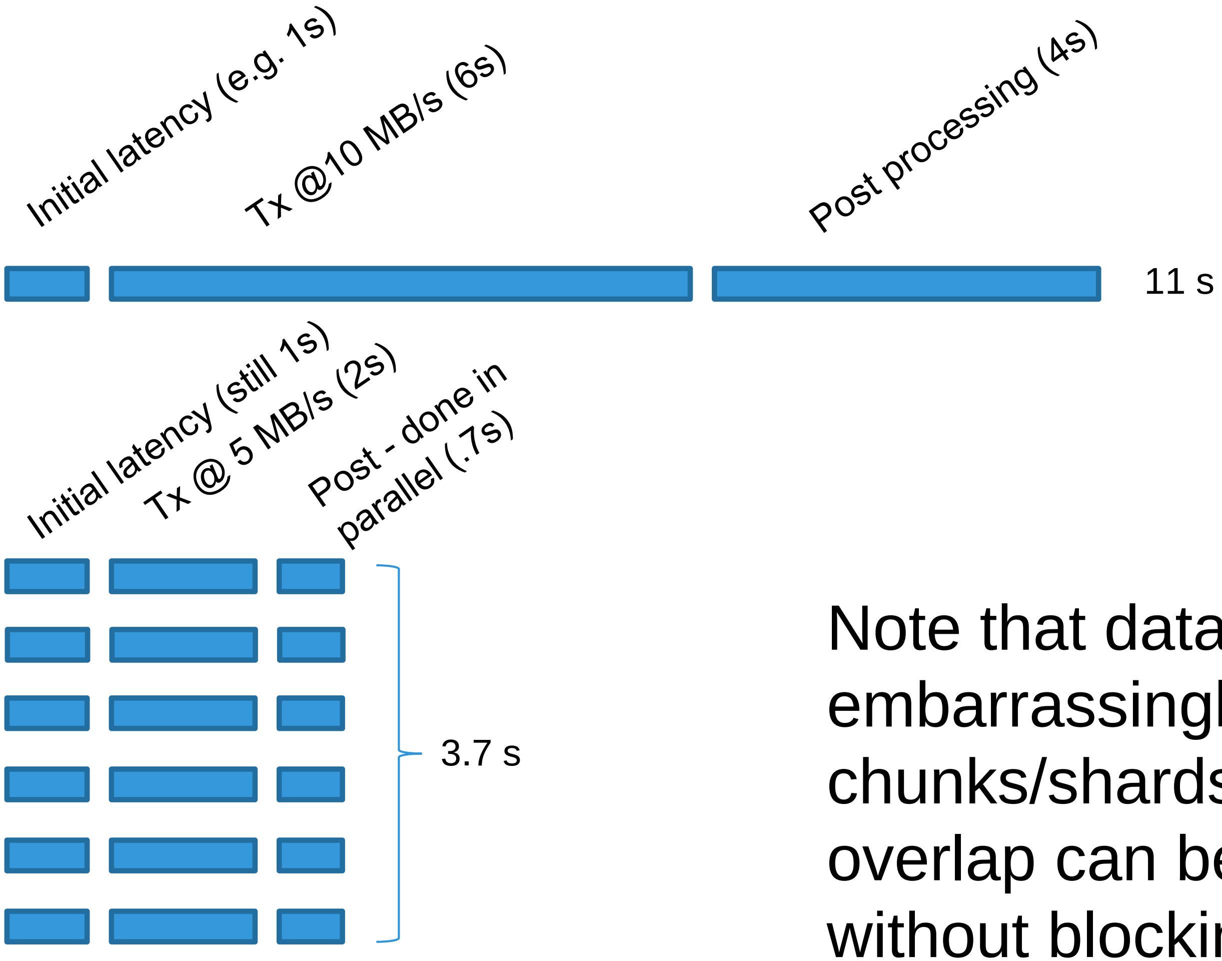
[1] **Hypertext Transfer Protocol**

Optimize Transfer Time

- Decompose data transfers into parallel operations
- S3 transfer rates *do* degrade with parallel access, but overall transfers are still faster
  - Single thread transfer rate: 10MB/s
  - Six (6) thread transfer rate: 5 MB/2*
- For data stored in a contiguous block, break it into ‘virtual’ shards/chunks

Parallel Transfer

Example: 60MB data transfer


Initial latency (e.g. 1s)
Tx @10 MB/s (6s)
Post processing (4s)
11 s
Initial latency (still 1s)
Tx @ 5 MB/s (2s)
Post - done in parallel (.7s)
3.7 s

Note that data access is embarrassingly parallel – chunks/shards that do not overlap can be inserted without blocking

Reduce Post-Transfer Time

- Perform post processing operations in parallel
  - But, do not use compression if possible
- For web services, provide data using its storage byte-order, not the server's – rely on the client to correct it
  - This eliminates 'touching' all of the data values.

This work was supported by NASA/GSFC under Raytheon Co. contract number NNG15HZ39C.

Raytheon

in partnership with

ati

Bellfore Consulting GROUP

Element 84

GENEX SYSTEMS

INNOVIM The Power of Innovation

MARYLAND HAWK CORPORATION

OPeNDAP

REAN CLOUD

Red Canyon Engineering & Software

STINGER GHAFFARIAN TECHNOLOGIES

The HDF Group

VALADOR®