CTU
CZECH TECHNICAL UNIVERSITY IN PRAGUE

The story of Hierarchical Semi-Sparse cubes in HDF5

Jiří Nádvorník, Petr Škoda, Pavel Tvrdík

HUG meeting 2022

1.6.2022

cerit scientific cloud

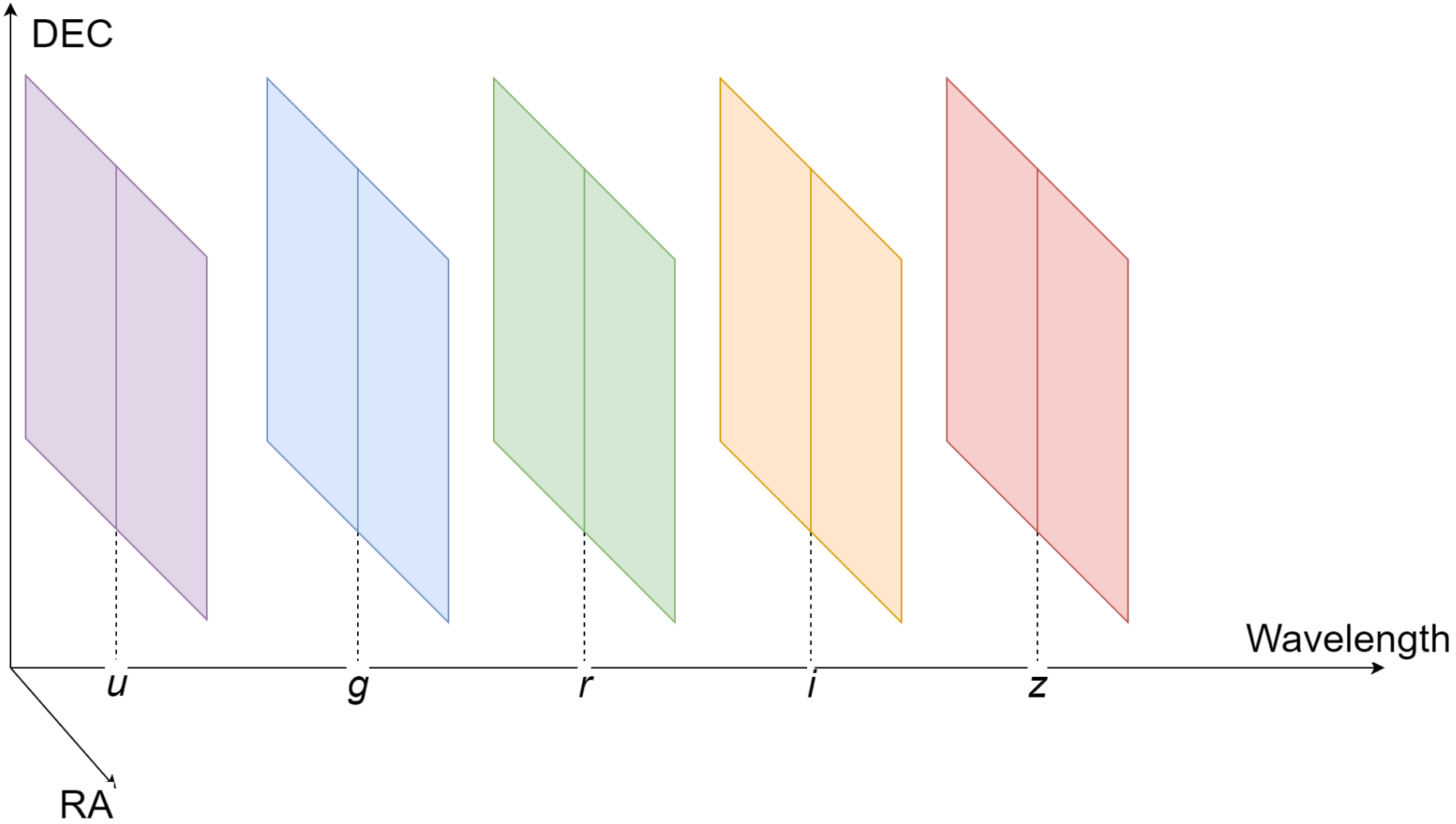
CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE
RESEARCH CENTER FOR INFORMATICS
rci.cvut.cz

EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research, Development and Education
MINISTRY OF EDUCATION, YOUTH AND SPORTS

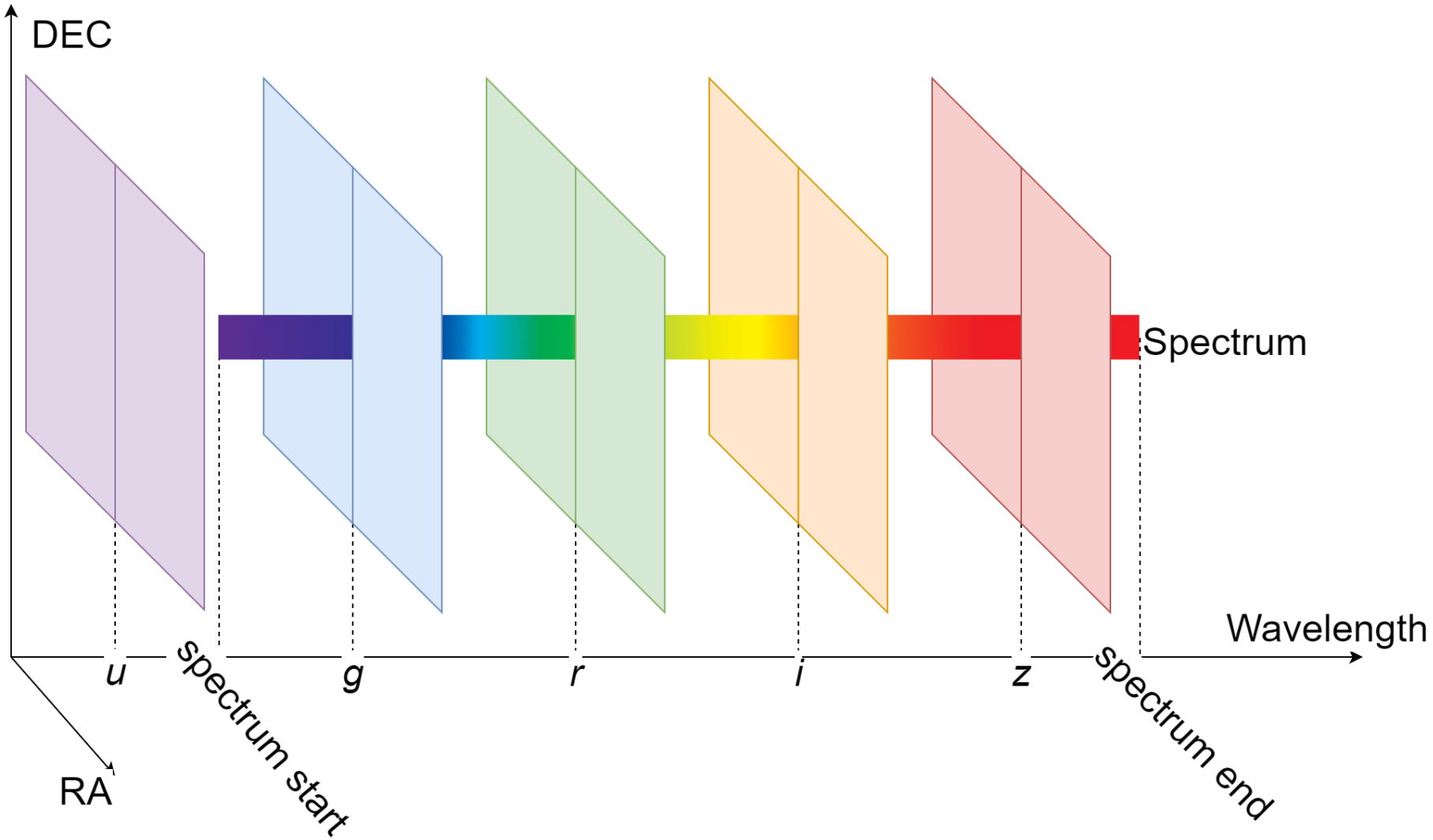
Introduction

- We need HDF5 for visualization and machine learning on 4D astronomical data (combinations of spectra and images).
- Our use case introduces the following challenges:
  - Data “semi-sparsity”
  - Hierarchical resolution access (similar to GoogleMaps rendering)
  - Need for pre-computed measurement uncertainties
  - Heterogeneous data (spectra and images are coming from different instruments)
- At the current stage, we have implemented a HiSS-Cube framework on top of h5py to support our requirements.
  - We are now working on making the framework scalable via parallel HDF5 library.
- This talk focuses on describing our use case and requirements, our work done so far, and lessons learned for both h5py and HDF5 libraries.

Semi-sparse 4D cube

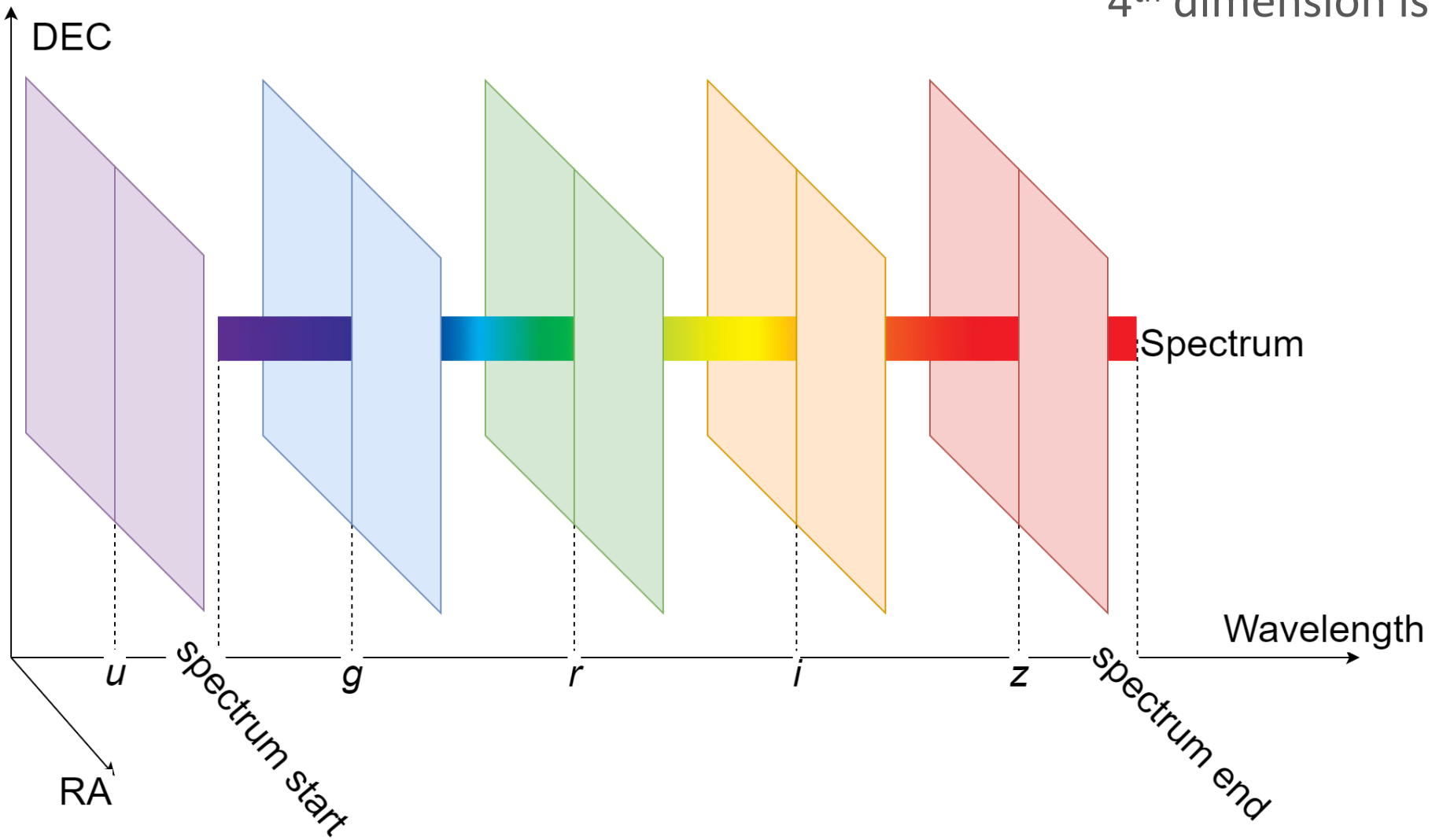


Semi-sparse 4D cube

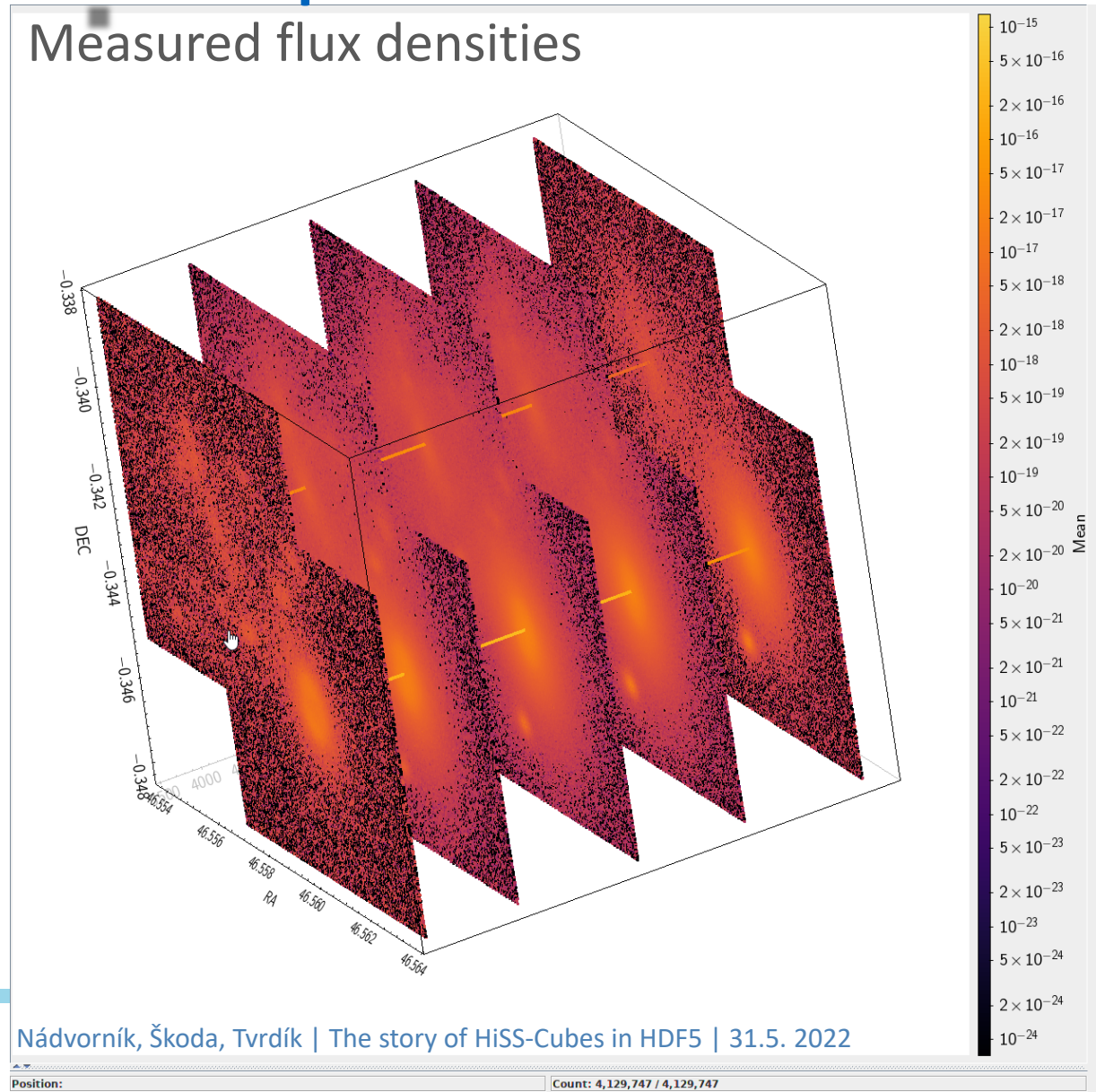


Semi-sparse 4D cube

4th dimension is time

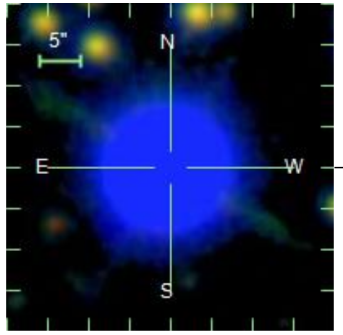


2 semi-sparse 4D cubes

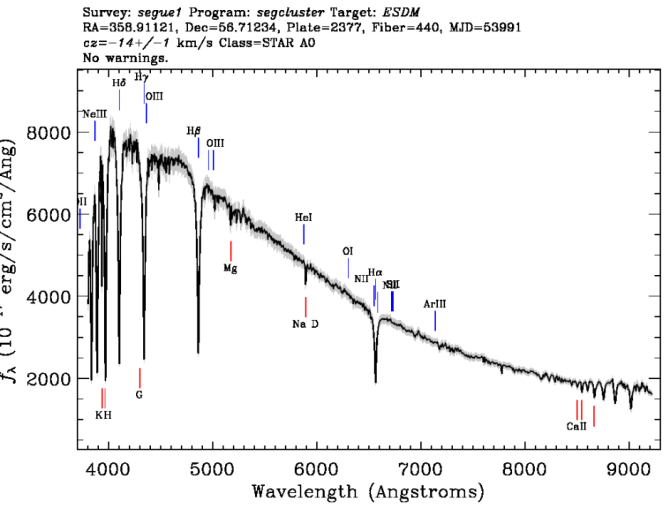


4th dimension is time

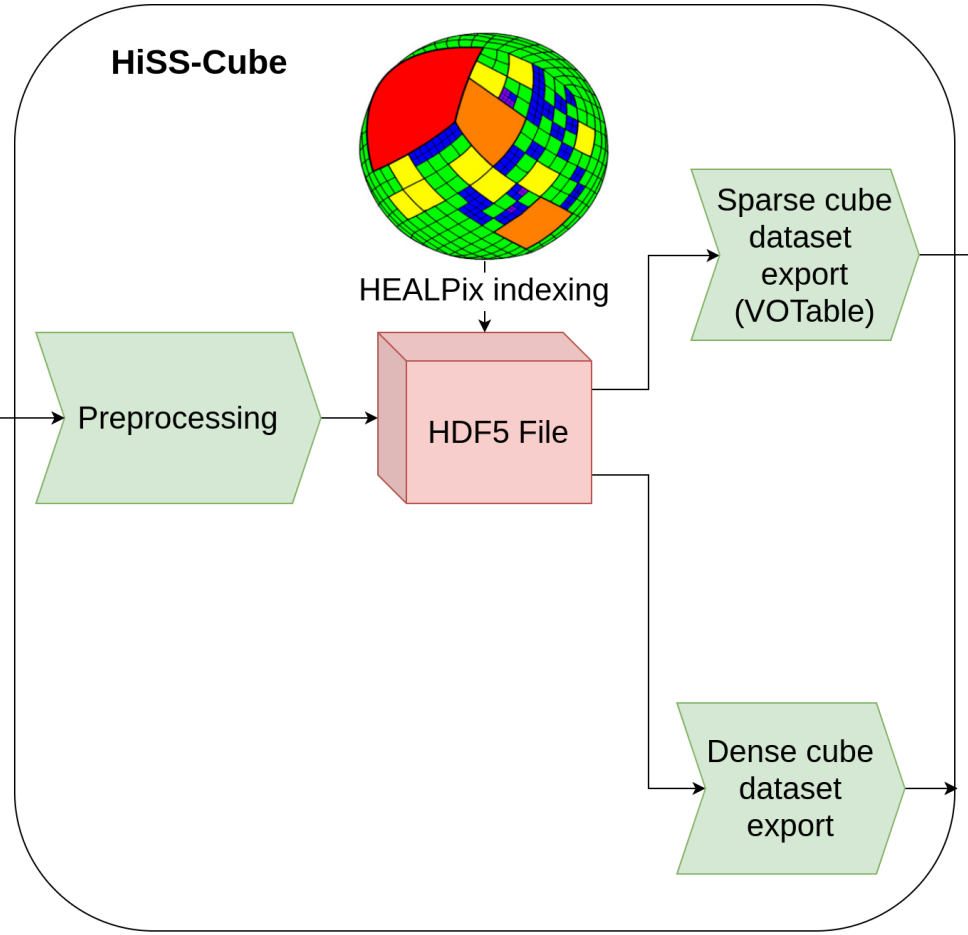
HiSS-Cube framework data flow

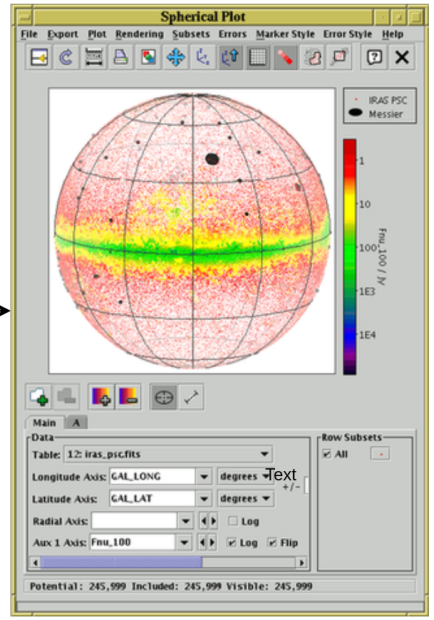


SDSS Image FITS

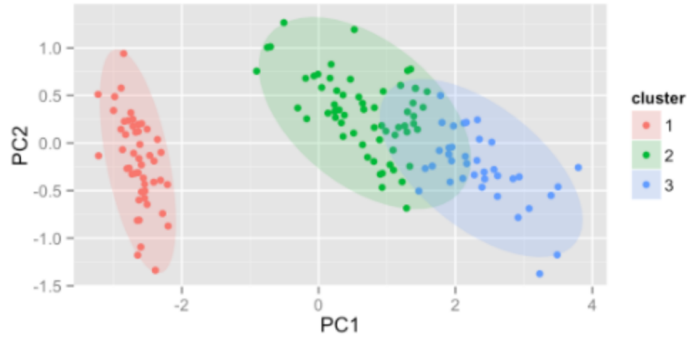


SDSS Spectrum FITS





TOPCAT visualization

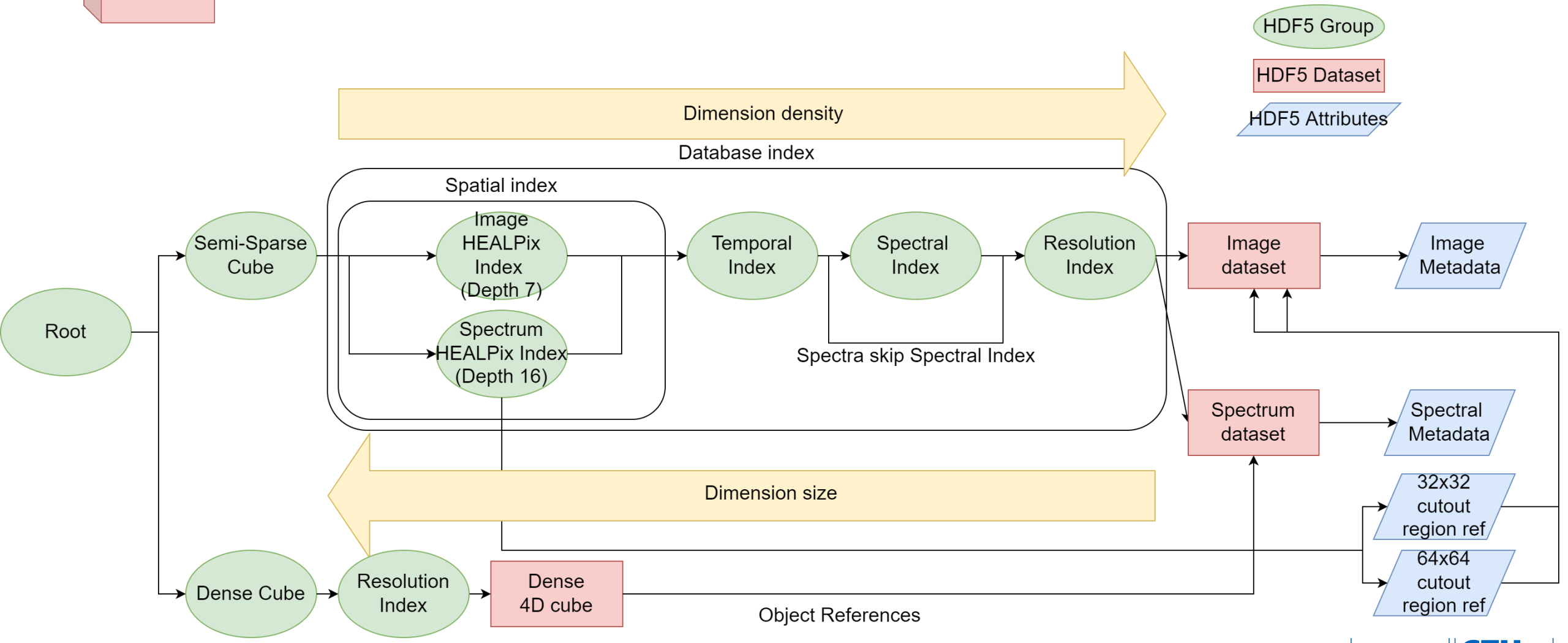


Machine learning algorithms

CZECH TECHNICAL UNIVERSITY IN PRAGUE

HDF5 File

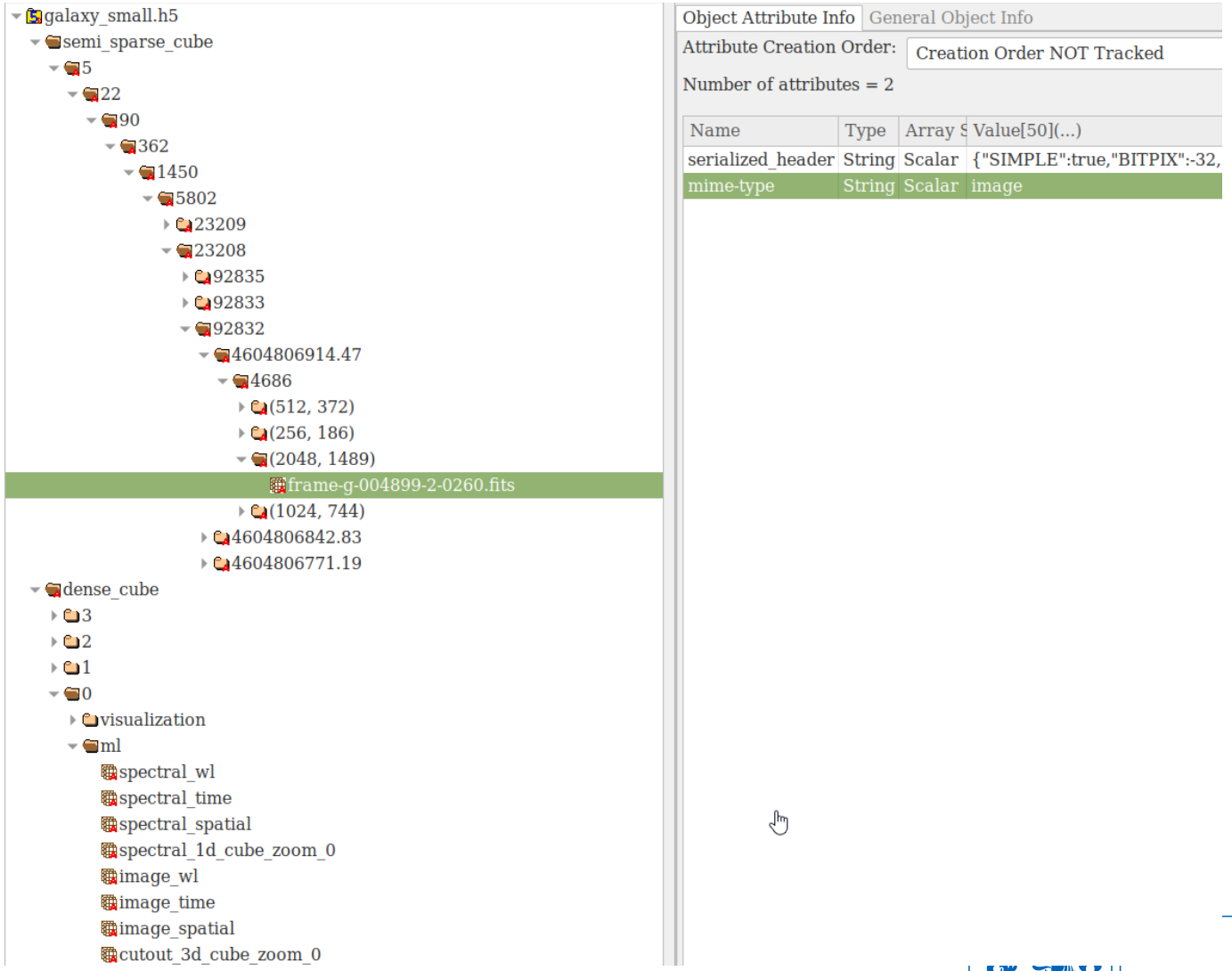
- detailed model



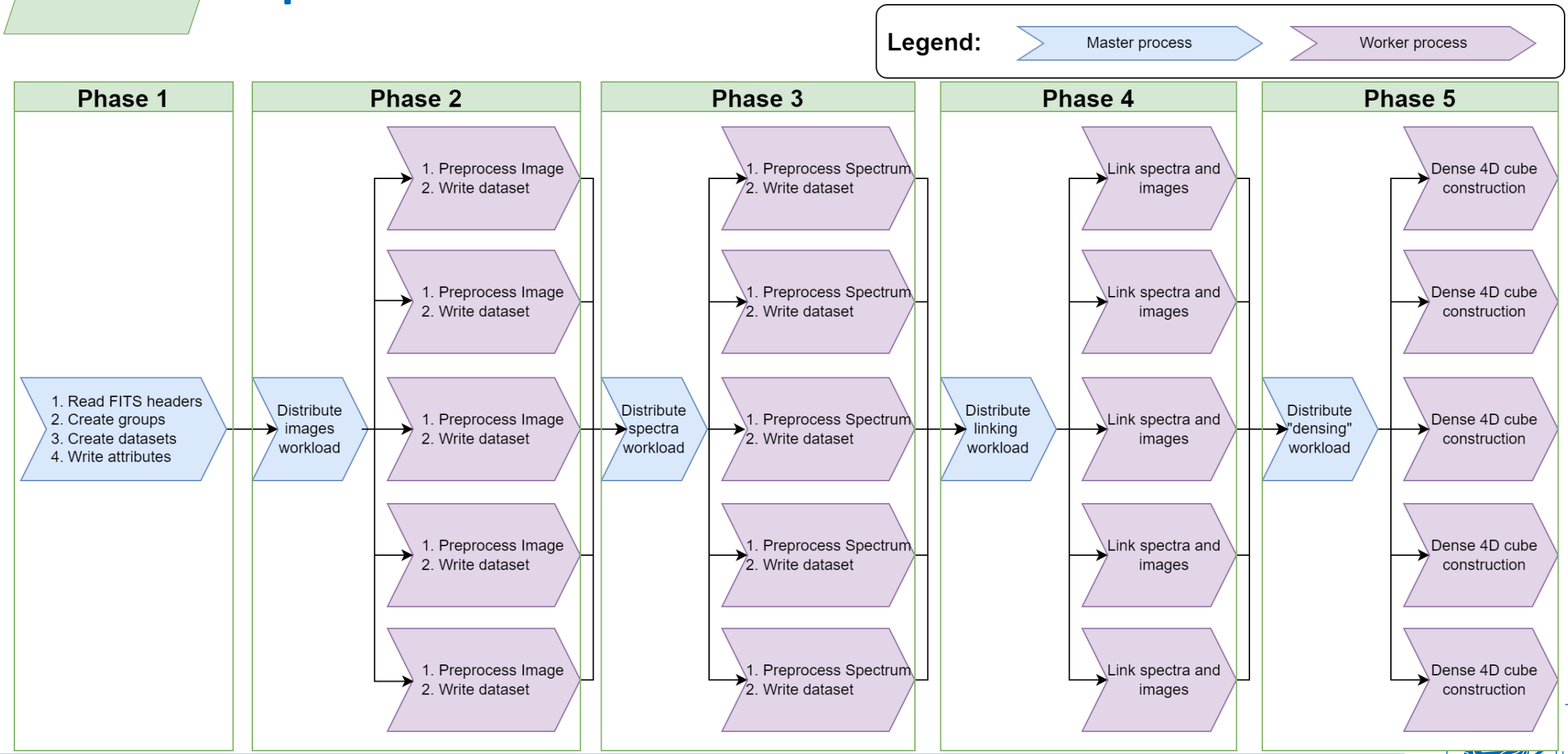
CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE

- structure in HDFView

- semi_sparse_cube group contains the index tree in a human readable form as a tree of groups.
- Beneath the lowest (<resolution> group) level of the tree are the datasets as equivalents to imported FITS files.
- dense_cube contains contiguous dense 4D cubes (or other precomputed datasets) for fast machine learning and visualization access.
  - The dense_cube is usually up to 10% of the size of a file.



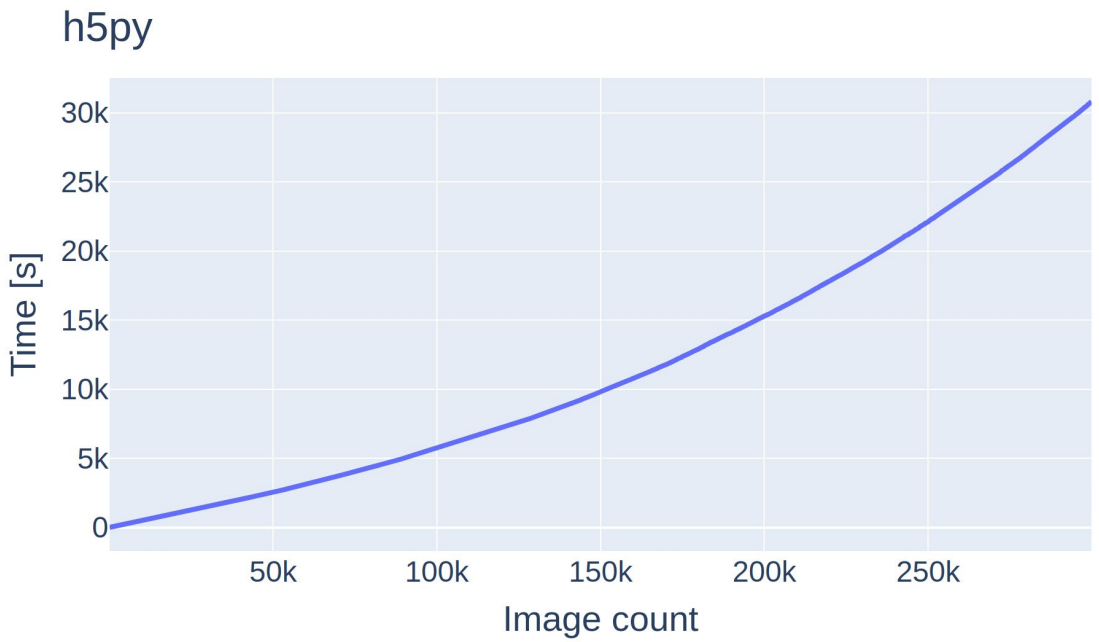
Preprocessing – parallel architecture detail

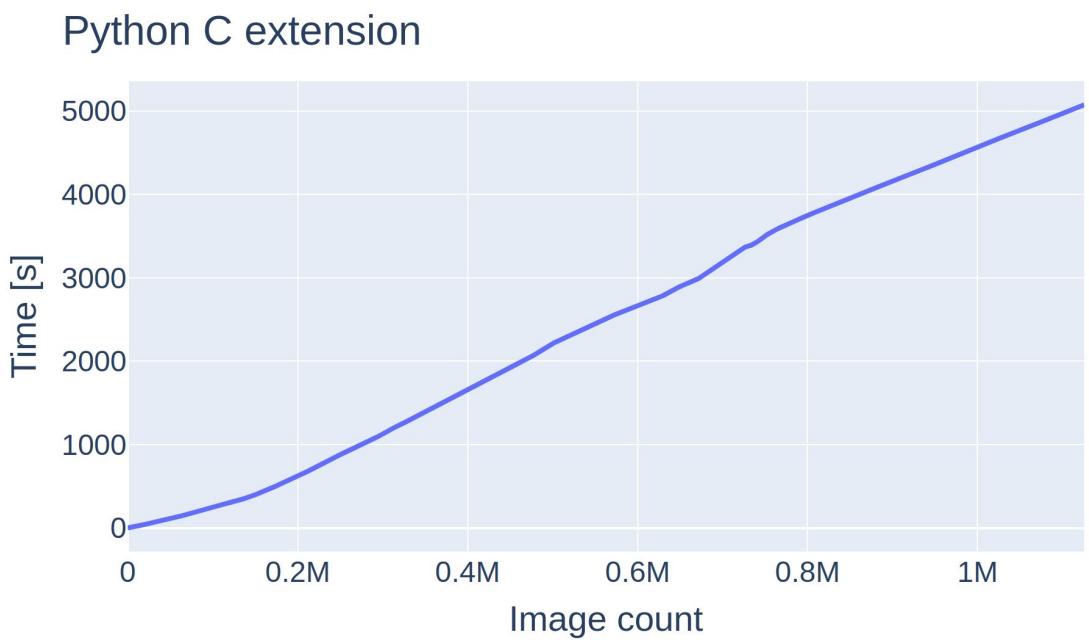


Testing data

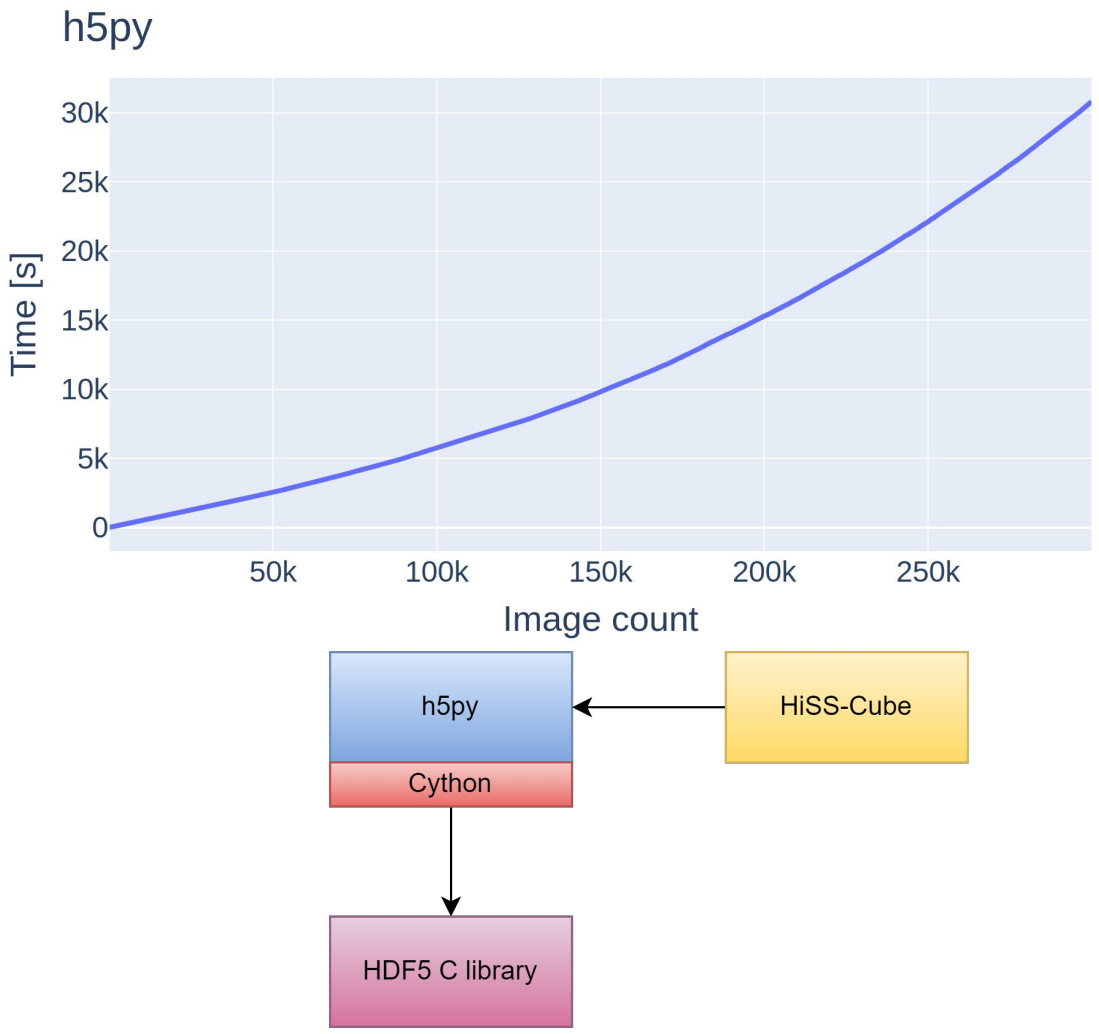
- Input data:
  - SDSS “Stripe 82” images
    - 15 TB uncompressed
    - ~1 million FITS files
  - SDSS spectra (BOSS)
    - 700 GB
    - ~4 million FITS files
- HDF5 file (after Phase 1):
  - ~60 TB
  - ~100 million groups
  - ~20 million datasets

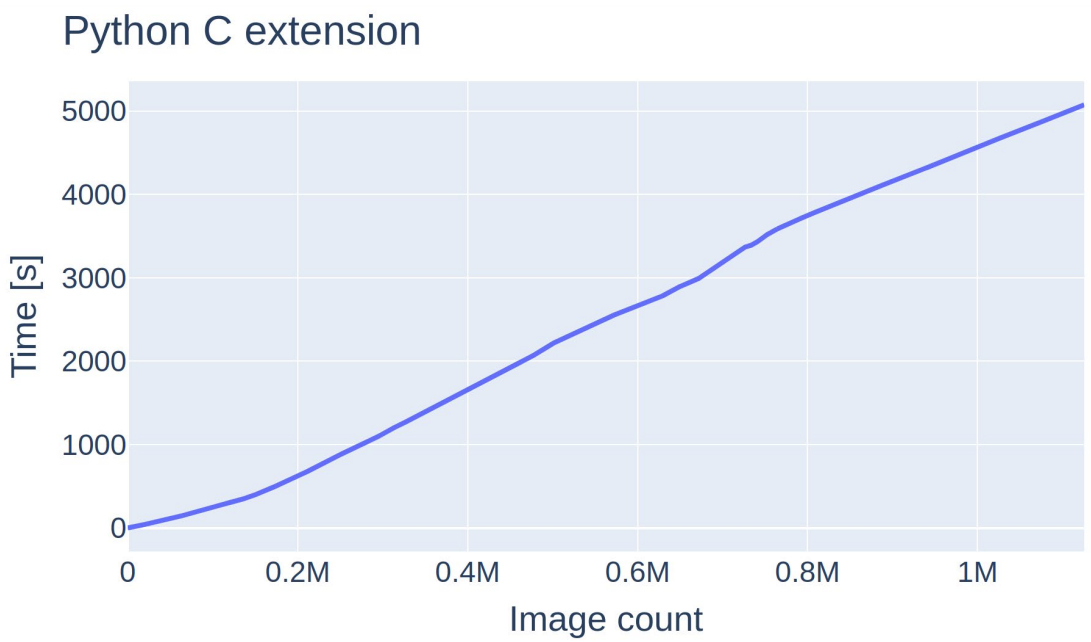
Total runtime of Phase 1 for all images



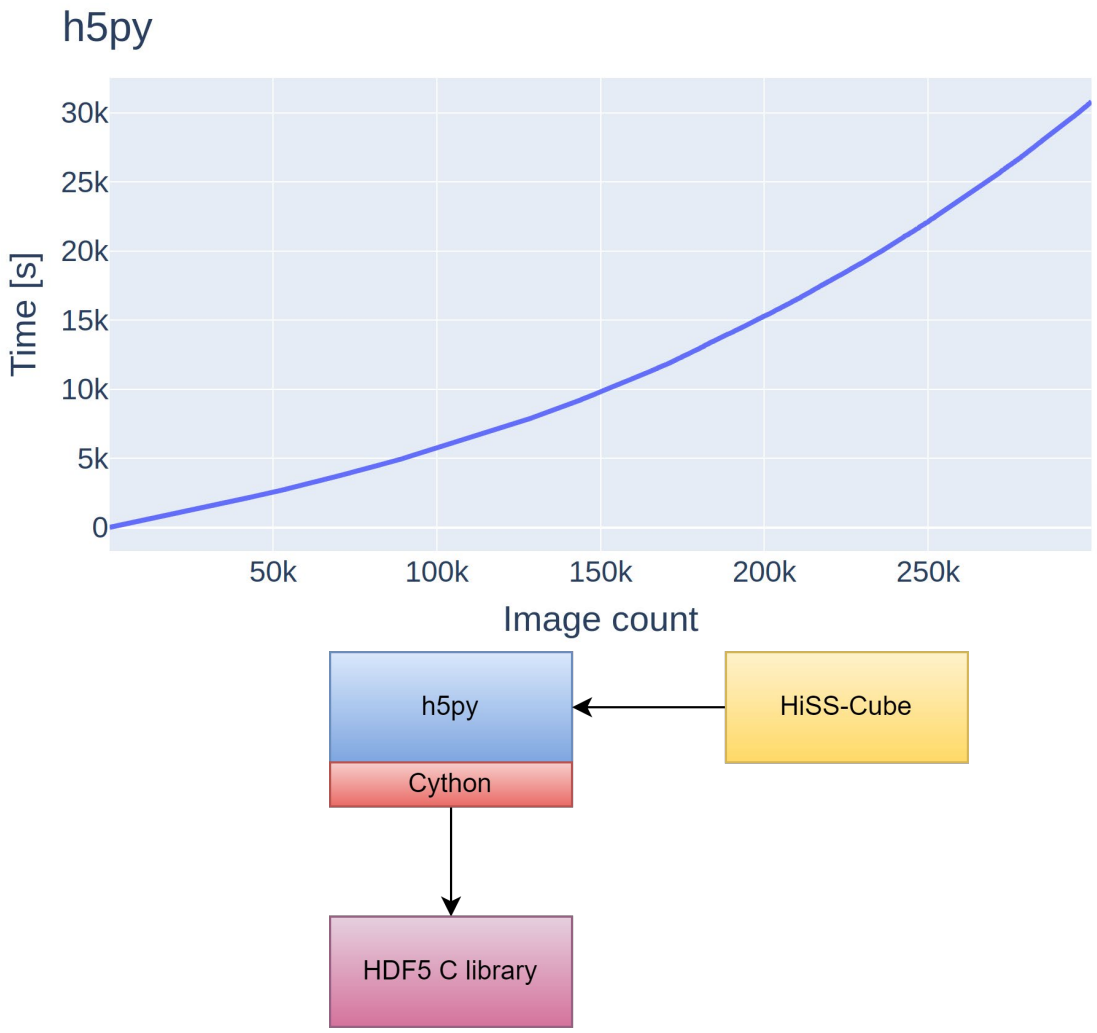


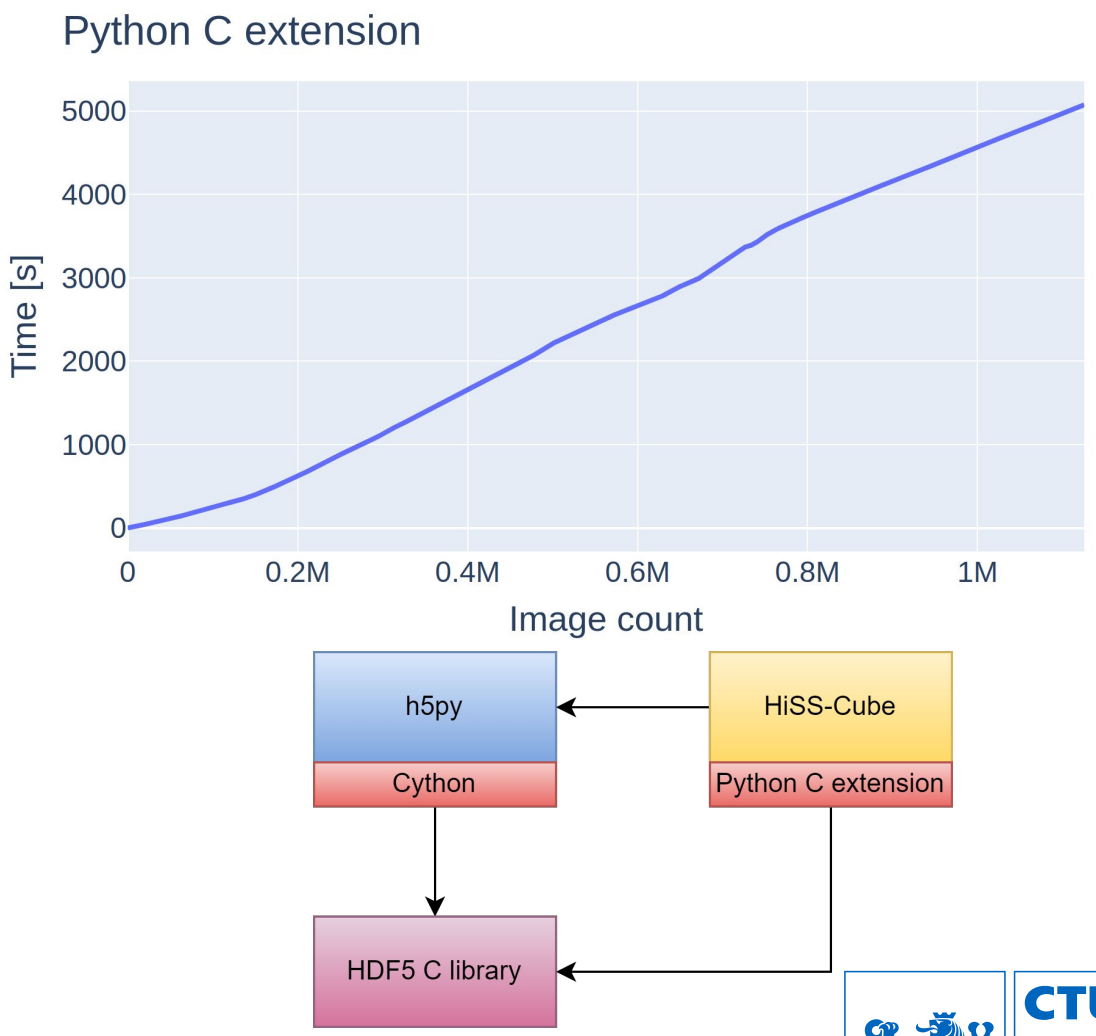
Total runtime of Phase 1 for all images





Total runtime of Phase 1 for all images





Normalized runtime of Phase 1 per 100 images

CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE

Phase 2+3: parallel HDF5 write bandwidth

- Peak bandwidth of the underlying GPFS measured with iozone:
  - 18167 MB/s sequential write
  - 2802 MB/s random write

Cluster hardware: 8 nodes	
CPU:	2x AMD EPYC 7543
Memory:	512GB
Disk:	2x 480GB SSD SATA 8x 7.3TiB SSD NVME
Network:	2x 10Gbps NIC

HiSS-Cube write bandwidth:

- Chunked dataset layout (128, 128, 2)

Number of workers	Write bandwidth Total [MB/s]	Efficiency / worker
8	200.60	100%
16	386.26	96%
32	677.15	84%
48	771.15	64%
64	667.05	42%
128	480.33	15%

- Contiguous dataset layout

Number of workers	Write bandwidth Total [MB/s]	Efficiency / worker
8	288.68	100%
16	563.49	98%
32	1071.14	93%
48	1455.16	84%
64	1911.22	83%
128	1587.49	34%

Read bandwidth of dense 4D cube dataset

- Peak bandwidth of theunderlying GPFS measured with iozone:
  - 22 154 MB/s sequential read
- HiSS-Cube read bandwidth on the 8-node cluster reading ~4 TB 4D dense cube dataset:

Number of workers	Time [s]	Read bandwidth per node [MB/s]	Read bandwidth total [MB/s]	Efficiency / worker
1	1503.2	2724.93	2724.93	100%
2	772.0	2652.96	5305.91	97%
3	558.3	2443.21	7329.64	90%
4	402.8	2542.08	10168.31	93%
5	329.7	2484.62	12423.11	91%
6	296.8	2299.93	13799.57	84%
7	279.1	2096.22	14673.51	77%
8	246.0	2081.54	16652.34	76%
16	252.2	1015.10	16241.53	37%
256	257.8	62.06	15887.04	2%

Summary

1. Scalability of HiSS-Cube to PBs of data needs further optimizations, but according to our tests so far, it is feasible overall.
2. Implementing Semi-Sparse data support on top of HDF5 is viable.
   a. Indexing the data is still viable in human-readable format for 60 TB file.
3. h5py library is mostly sufficient but we needed to rewrite some parts to Python C extensions to optimize it.

- Future work
  - Phase 1 scalability
    - HDF5 file allocation is not scaling to PBs of data currently.
    - Partial parallelization might help.
    - Moving away from human-readable group structure to dataset-based index.
  - Phase 2-3 scalability
    - Performance of writes to HiSS-Cube is still significantly under thr peak performance of GPFS.

Acknowledgements

- We would like to thank Gerd Heber for very outgoing approach and time spent investigating all issues connected with our work.
- This research is supported by the project OP VVV, Research Center for Informatics, CZ.02.1.01/0.0/0.0/16_019/0000765 of the Czech Ministry of Education, Youth and Sports.
- Computational resources were supplied by the project "e-Infrastruktura CZ" (e-INFRA CZ LM2018140) supported by the Czech Ministry of Education, Youth and Sports.

Q&A

Backup slides

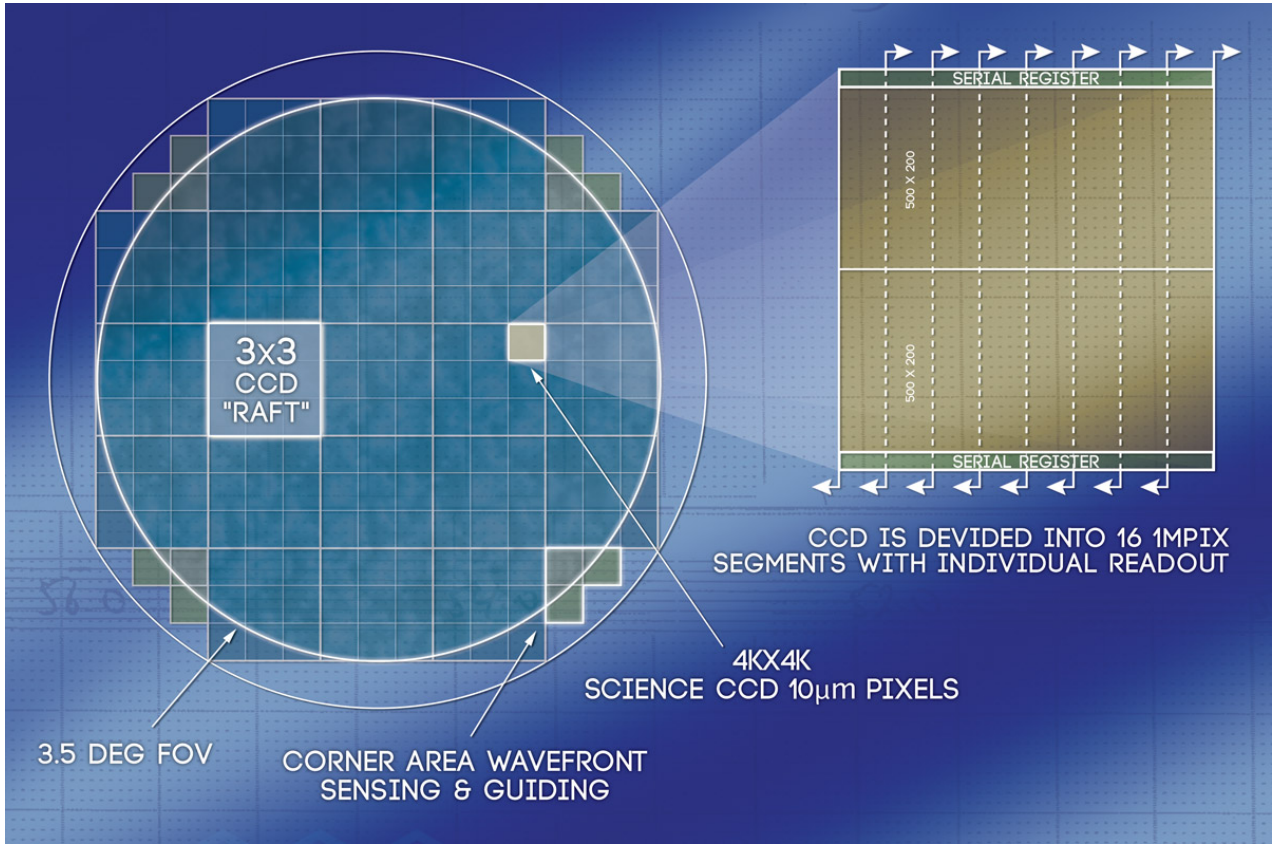
cerit scientific cloud
CTU CZECH TECHNICAL UNIVERSITY IN PRAGUE
RESEARCH CENTER FOR INFORMATICS
rci.cvut.cz
EUROPEAN UNION
European Structural and Investment Funds
Operational Programme Research, Development and Education
MINISTRY OF EDUCATION, YOUTH AND SPORTS

Key requirements on HiSS-Cube

- Uncertainty
  - Storing uncertainties (sigma, error values) along with the data is necessary for fast machine learning processing.
- Hierarchical (multi-resolution) access
  - Both machine learning and later visualization of original data can work much faster on lower resolutions with only small degradation of reliability.
  - HiSS-Cube needs to support access for:
    - Visualization (random access)
    - Machine learning (contiguous access)
- Combining heterogenous semi-sparse data
  - The data is coming from different instruments.
  - The resulting 4D coordinate space is very sparse, but contains dense regions.
- Scalability
  - The system needs to horizontally scale to support petabytes of data.

Desired scalability

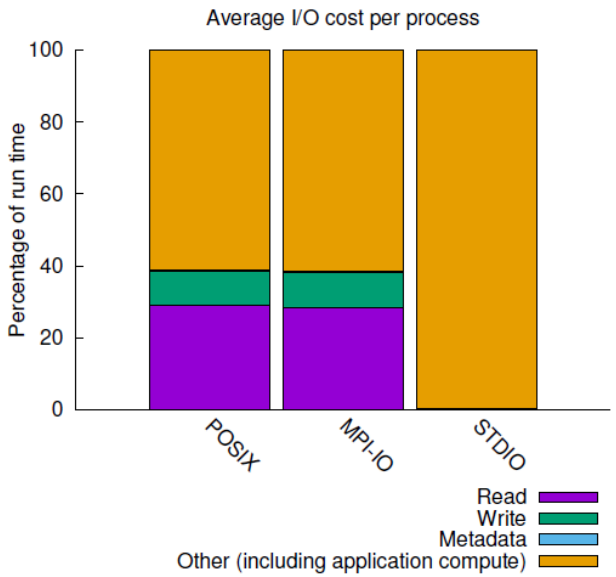
- Images
  - LSST (Rubin) observatory will produce 20TB of images per night
  - 1-2 PB for Data Release 1 (DR1), 15 PB for DR11
  - 3.2 Gigapixel per image
- Spectra
  - Instruments will commonly produce 50-100 million spectra per data release
- Images and spectra from multiple instruments can be combined together.

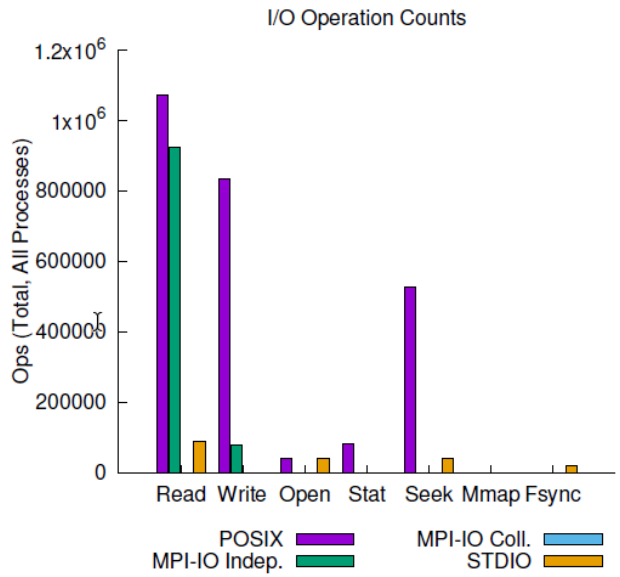


Performance testing – Phase 1-3 I/O profiling

jobid: 925167	uid: 1001	nprocs: 129	runtime: 489 seconds

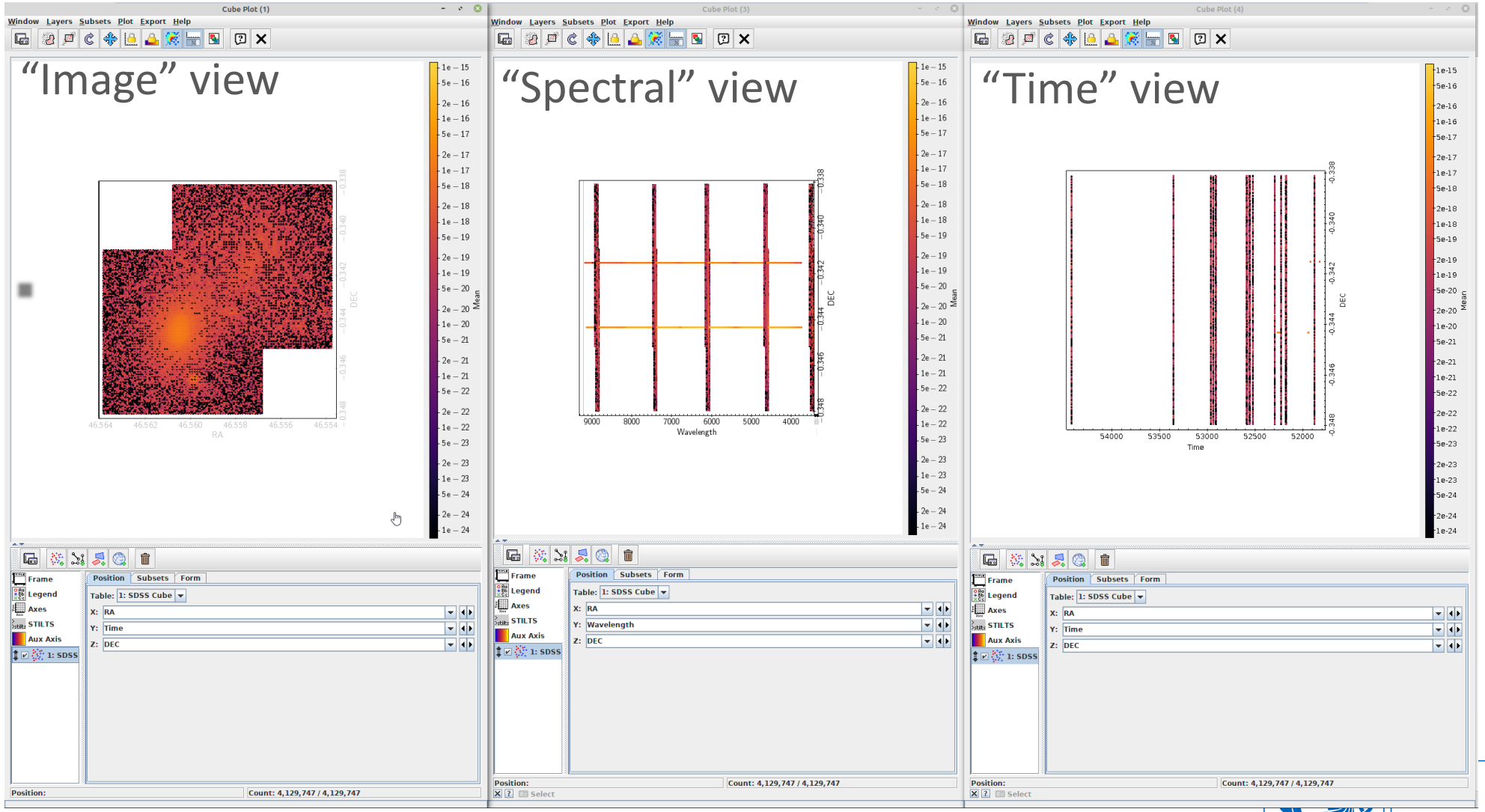
I/O performance *estimate* (at the MPI-IO layer): transferred 310720.8 MiB at 1469.38 MiB/s
I/O performance *estimate* (at the STDIO layer): transferred 194.1 MiB at 1.77 MiB/s





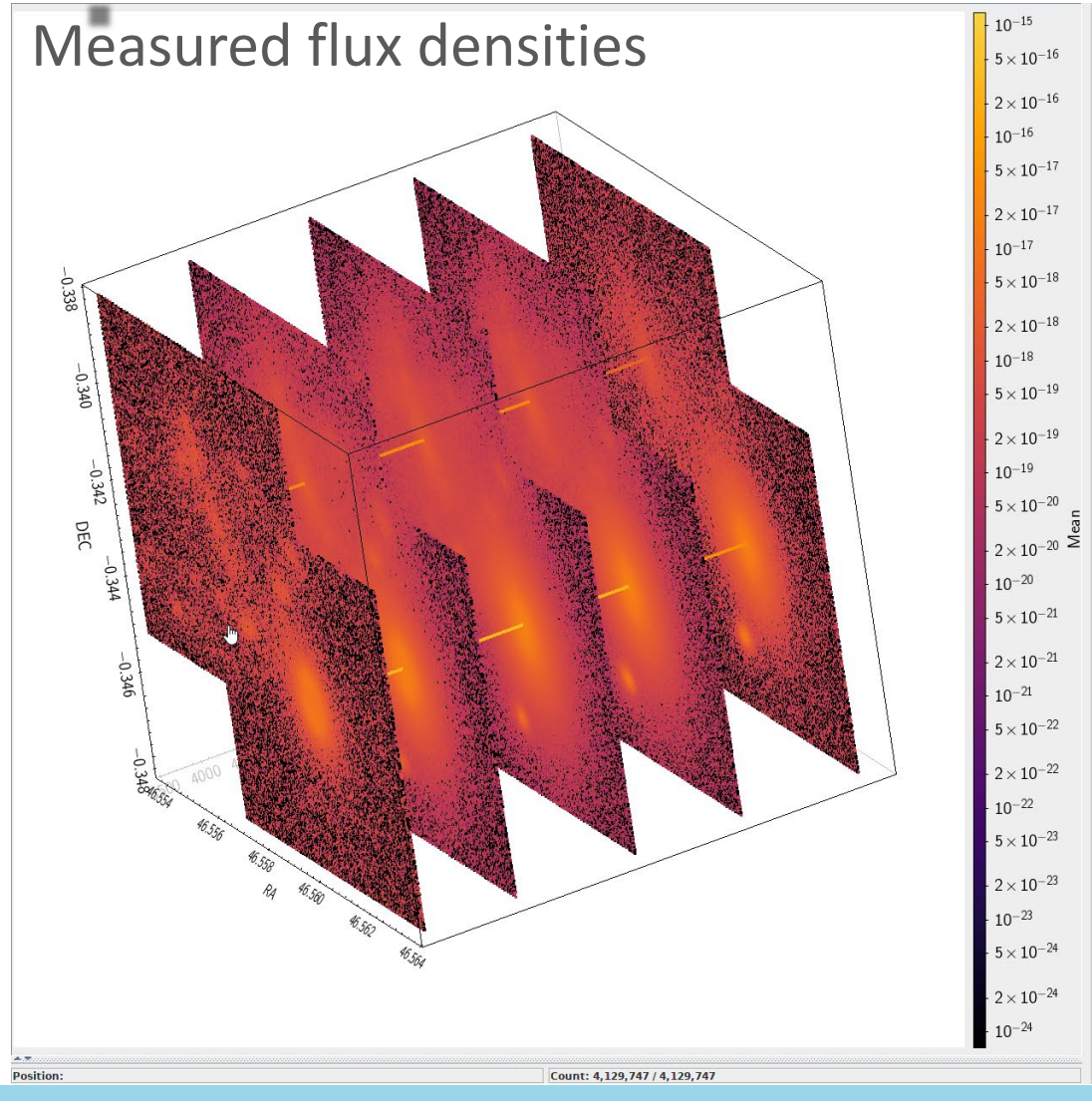
2 Semi-sparse 4D cube rotations and slices

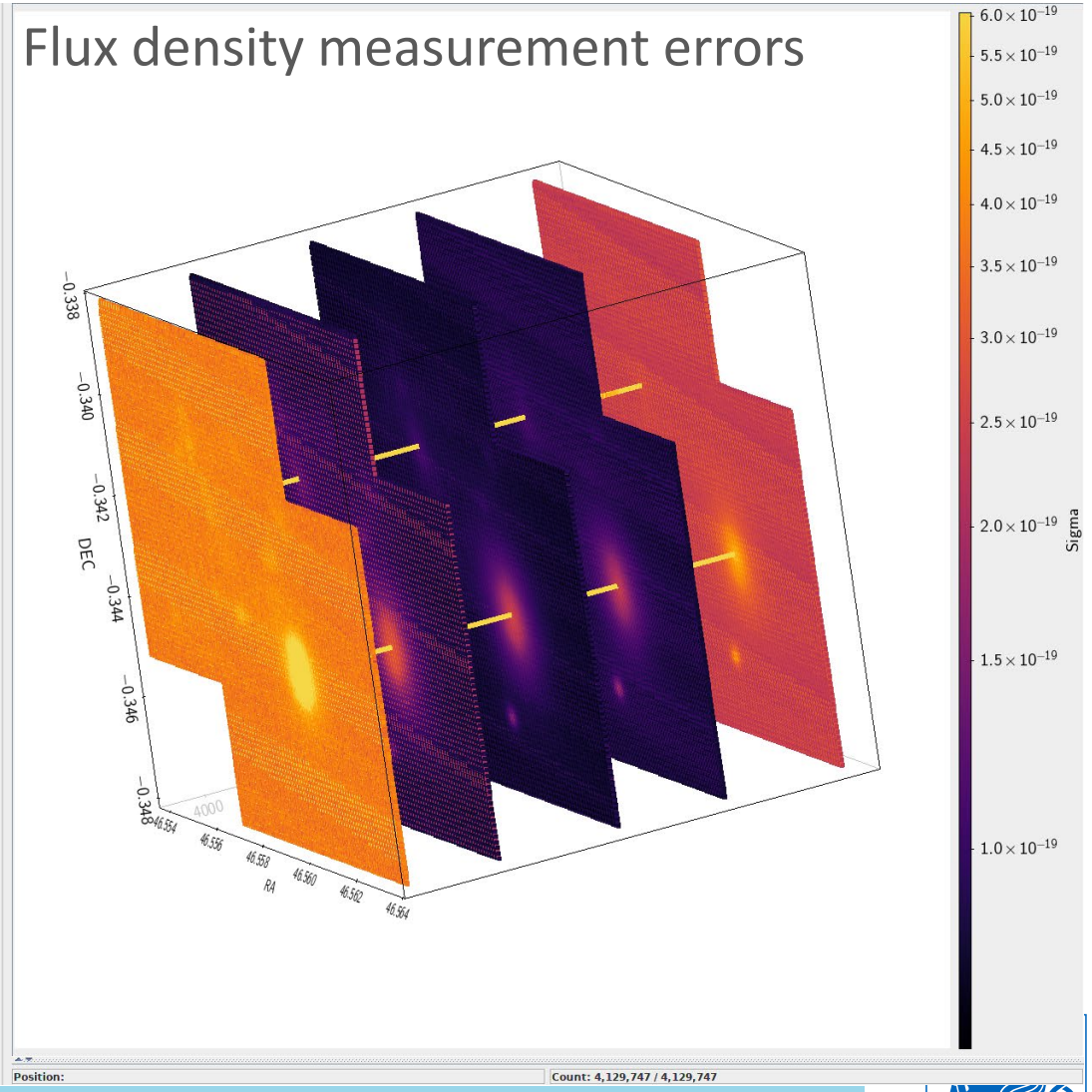
Measured flux densities



CZECH TECHNICAL UNIVERSITY IN PRAGUE

2 semi-sparse 4D cubes with errors





Hierarchical aspect of HiSS cubes

- We use HEALPix for spatial indexing (tessellation of sphere).
  - In combination with precomputed lower resolutions we can use the hierarchical aspect for:
    - Quick visualization of original data (images and spectra)
    - Quick training of machine learning models on lower accuracy data.
- The construction of lower resolutions for both images and spectra is non-trivial when we need to preserve scientific accuracy for the measurement errors.
  - Lower resolution is constructed via cubic interpolation and errors propagated accordingly.