Blosc2

Fast And Flexible Handling Of N-Dim & Sparse Data

Francesc Alted / @FrancescAlted

The Blosc Development Team / @Blosc2

CEO ironArray / @ironArray

PyData Global
Dec 6th 2023

Agenda

Intro to Blosc2

Extending Blosc2 Via Plugins

Automatic Compression Tuning with Btune

Conclusions

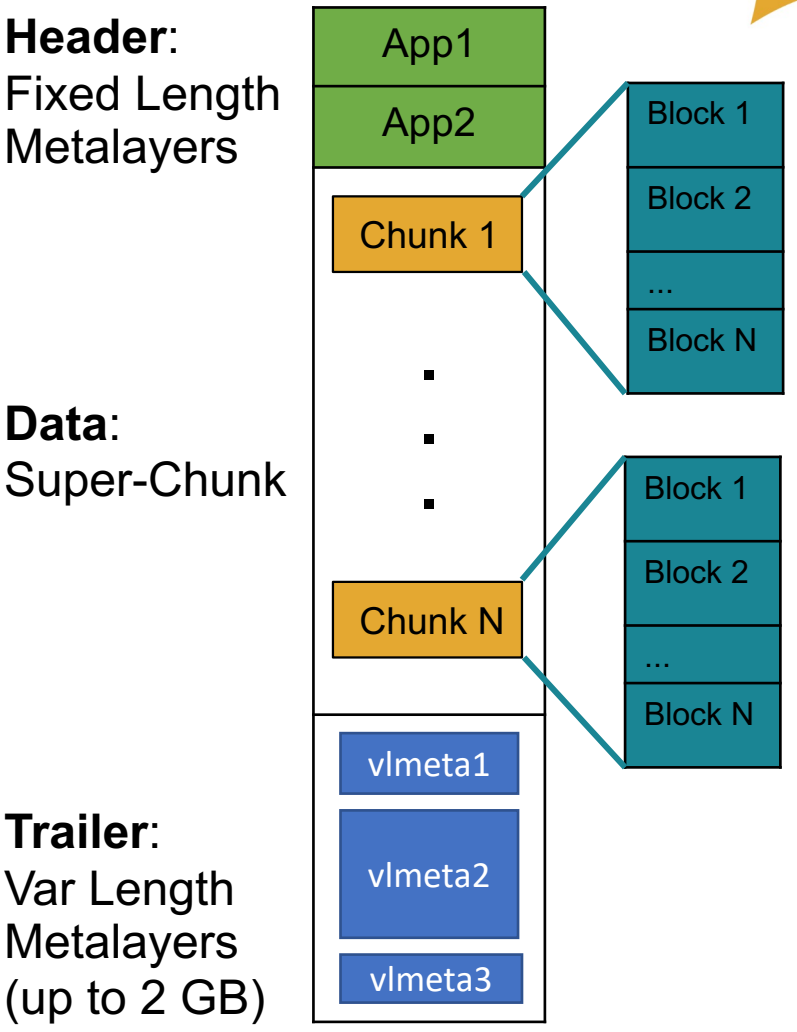
Intro to Blosc2

A highly effective library (C and Python) for handling multidimensional and sparse datasets

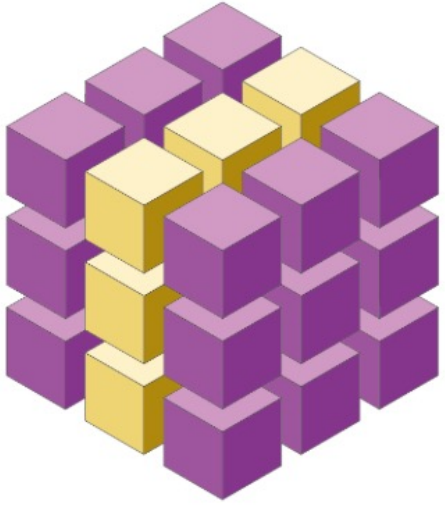
https://www.blosc.org

What Is Blosc2?

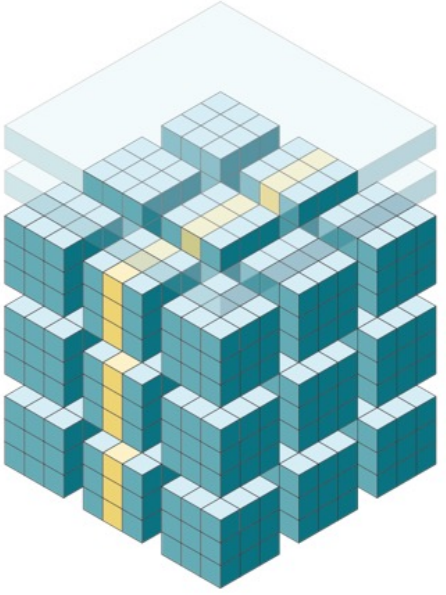
- ✓ C and Python libraries
- ✓ Simple format
- ✓ 63-bit containers
- ✓ Fully multidimensional double partitioning
- ✓ Supports different codecs and filters
- ✓ Metalayers for adding info for apps and users



Leveraging A Second Partition In NDim Data



HDF5 / Zarr / others



Blosc2

Much more selective and faster queries!

NDArray: NDim Arrays for Python

```python
import blosc2

a = blosc2.full((4, 4), fill_value=9)
a.resize((5, 7))
a[3:5, 2:7] = 8
print(a[:])
```

Output:

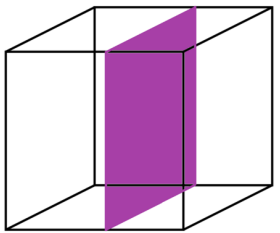
```
[[9 9 9 9 0 0 0]
 [9 9 9 9 0 0 0]
 [9 9 9 9 0 0 0]
 [9 9 8 8 8 8 8]
 [0 0 8 8 8 8 8]]
```

Features:

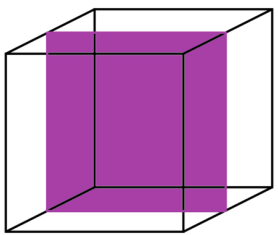
- Create arrays in memory or on disk
- Flexible resize (including shrinking)
- Efficient conversion from/to NumPy
- Mimic NumPy API

https://www.blosc.org/python-blosc2/reference/ndarray_api.html

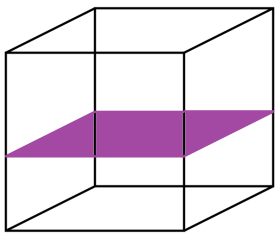
Making Other Formats More Efficient
The HDF5 Case: PyTables and h5py



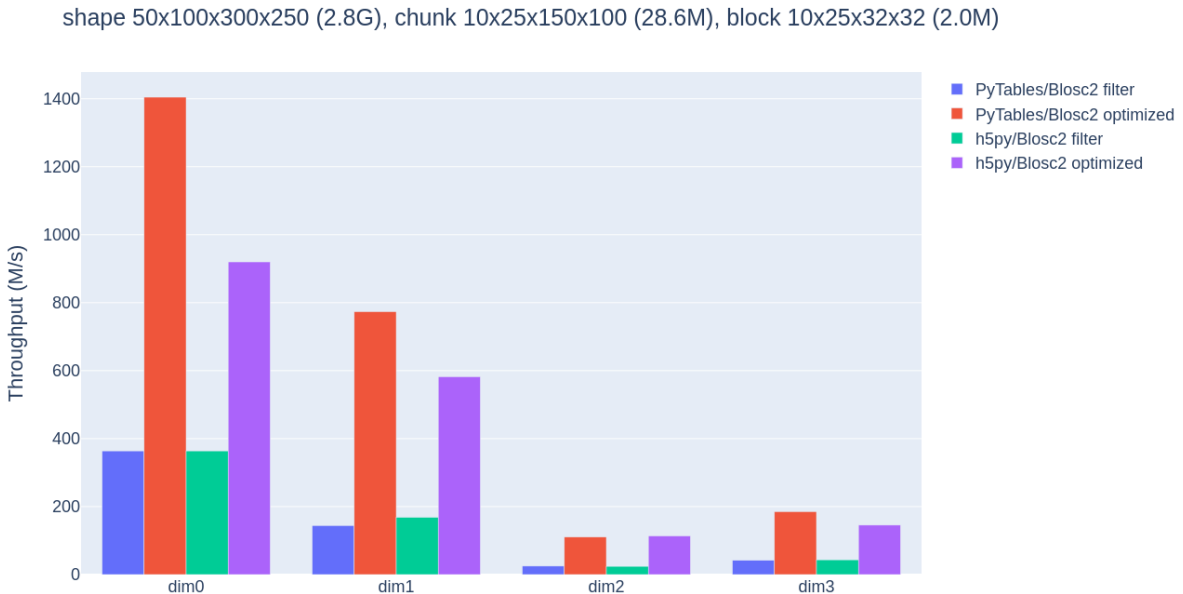
constant dim0



constant dim1



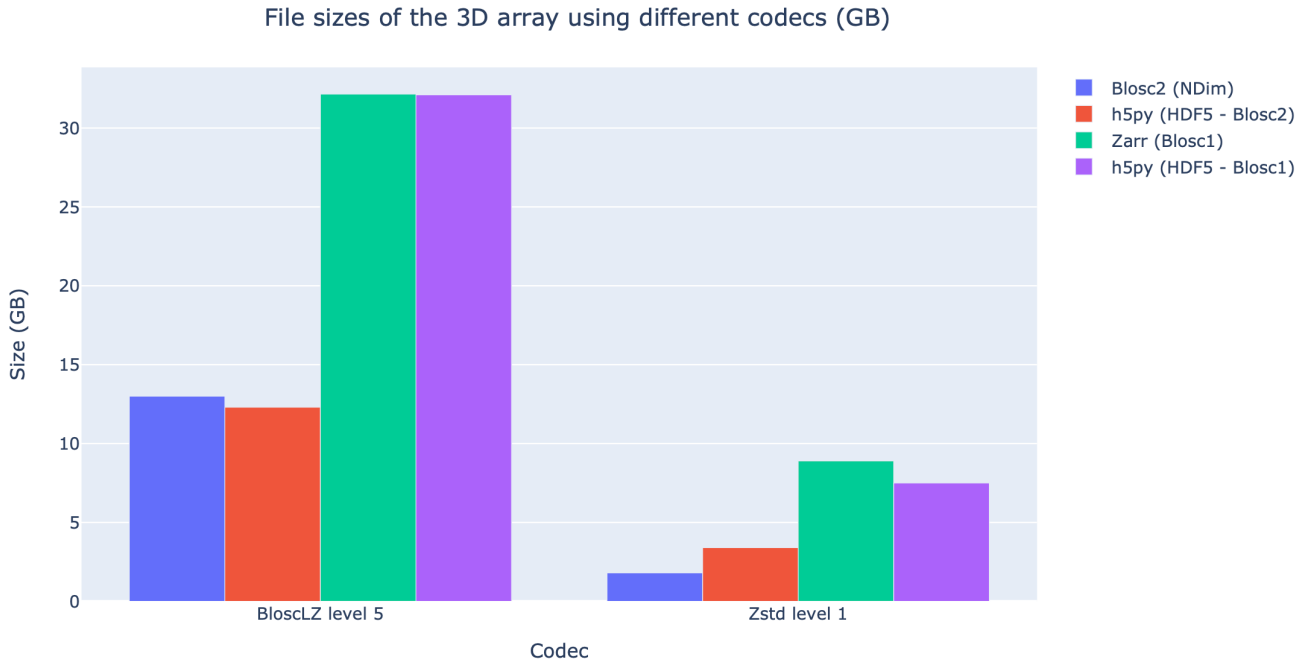
constant dim2



Faster slicing due to higher data selectivity in double partitioning

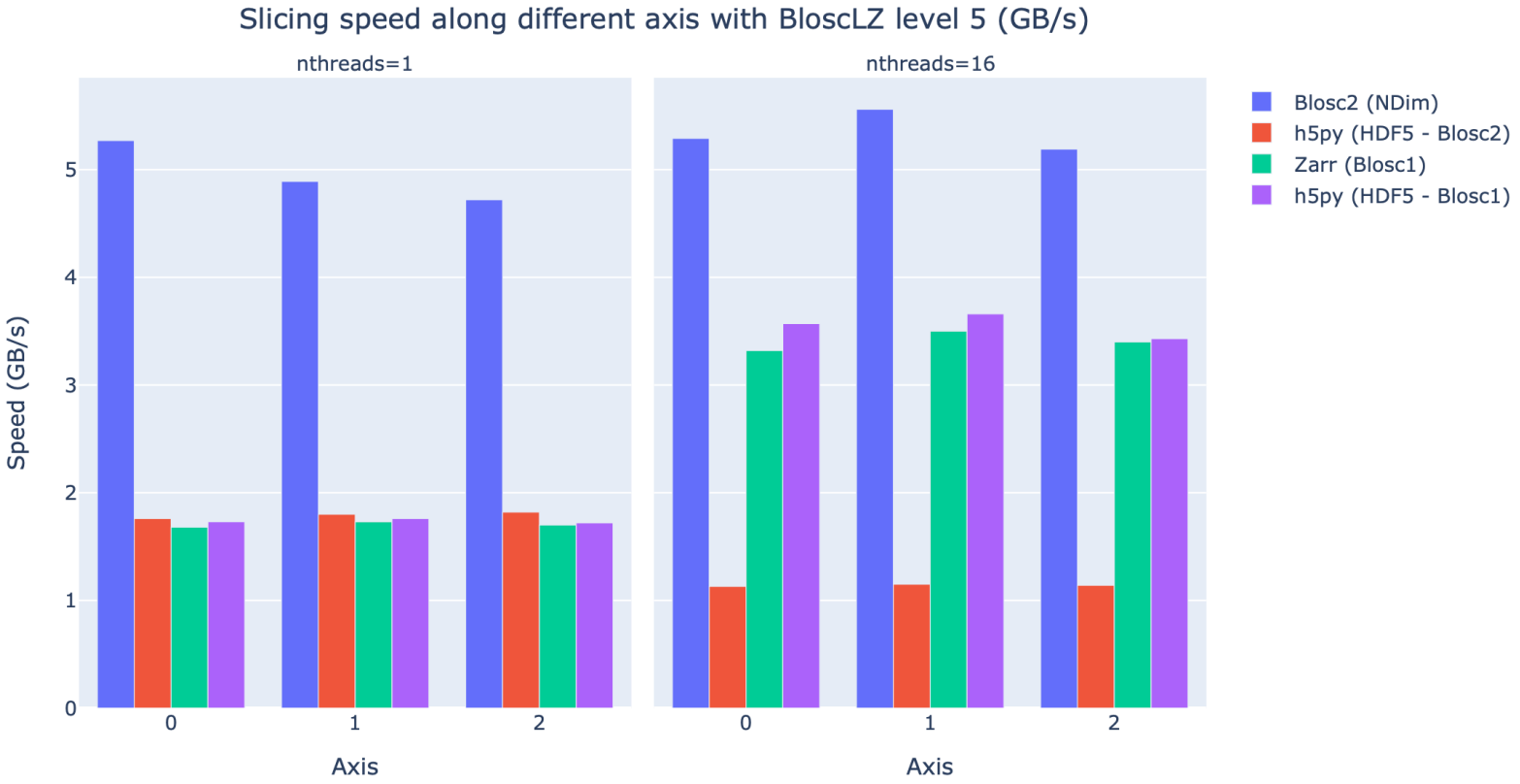
Support for h5py via https://github.com/Blosc/b2h5py

Blosc2 File Sizes for Sparse Arrays



- Data represents Milky Way stars in a big 3D cube (from Gaia mission)
- Cube with 8 trillion cells (7.3 TB!) and 0.5 billion stars (10,000x sparsity).
- Blosc2 + Zstd packs the entire 3D Gaia array in less than 2 GB (cr 4000x !)

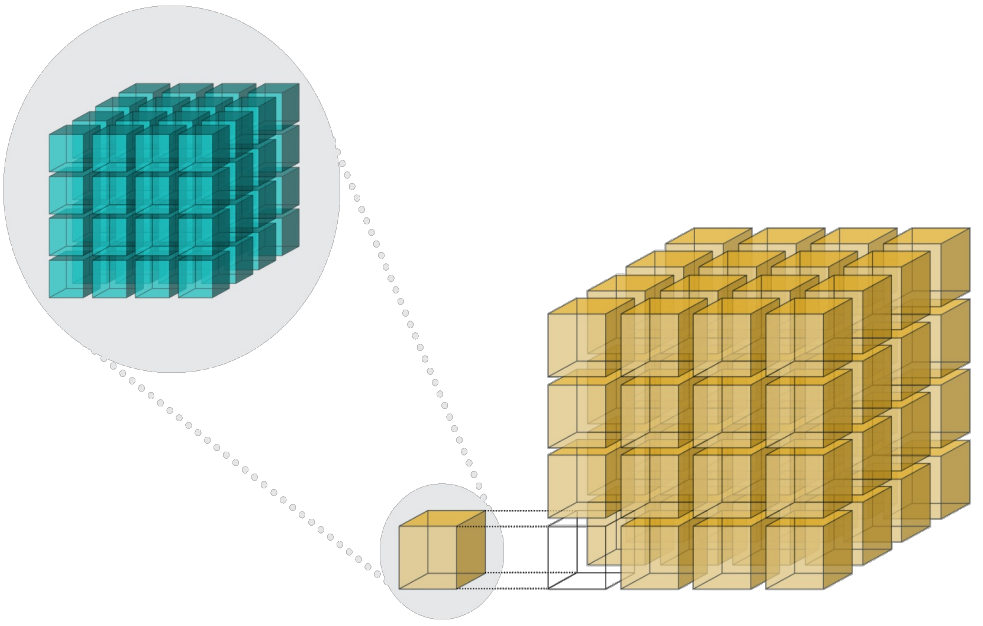
Read Performance for Sparse Arrays



- Better sparse support for Blosc2 makes it faster
- Higher data selectivity in double partitioning

C-Blosc2: Multidimensions For C

- ✓ C library with a low level API for other languages
- ✓ Leverage Blosc2 multidimensional capabilities from other languages (Rust, Julia, R...)
- ✓ The NDArray Python class is a shallow wrapper



https://www.blosc.org/c-blosc2/reference/b2nd.html

Dynamic Plugins

Extending Blosc2 to your needs

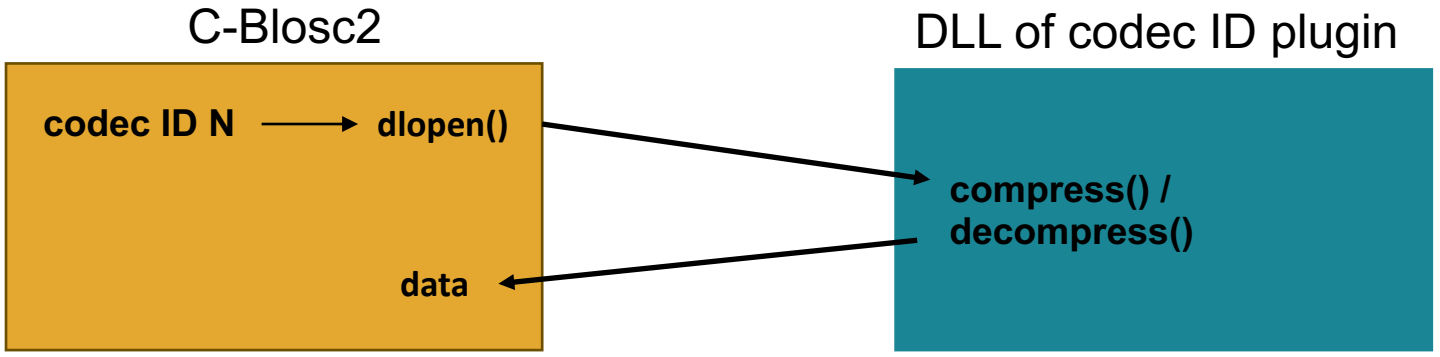
Extending Blosc2 Via Dynamic Plugins

- Blosc2 supports a wide range of codecs and filters right out-of-the-box.
- However, there is always a need that is not well covered by the standard suite.
- Blosc2 allows users to implement their own ones and register them for being used automatically:

https://www.blosc.org/posts/registering-plugins/

Loading Plugins Dynamically
How Does It Work?

Whenever C-Blosc2 receives a request for using **dynamic codec ID N**, it will **dynamically load** its **DLL library** using **dlopen()/LoadLibrary()** calls:



https://www.blosc.org/posts/dynamic-plugins/

Pros and Cons of Dynamic Plugins

Pros

- Very easy to install:

```
$ pip install blosc2-grok
```

- Does not bloat the C-Blosc2 library as other plugin managers do (hdf5plugin, numcodecs...).
- Support for C, C++, Rust plugins. Only requisite is to expose a C API!

Cons

- Somewhat more work to create. But we are providing an example with detailed instructions:

  https://github.com/Blosc/blosc2_plugin_example#readme

Practical Example: blosc2-grok
A Plugin for JPEG 2000 (Lossy)

- High quality lossy compression for images.
- It uses grok, an open source JPEG200 (and HTJ2K) implementation by Aaron Boxer.
- Packed and distributed as a Python wheel:
    - `$ pip install blosc2-grok`

Example of use:

https://github.com/Blosc/blosc2_grok/blob/main/examples/params.py

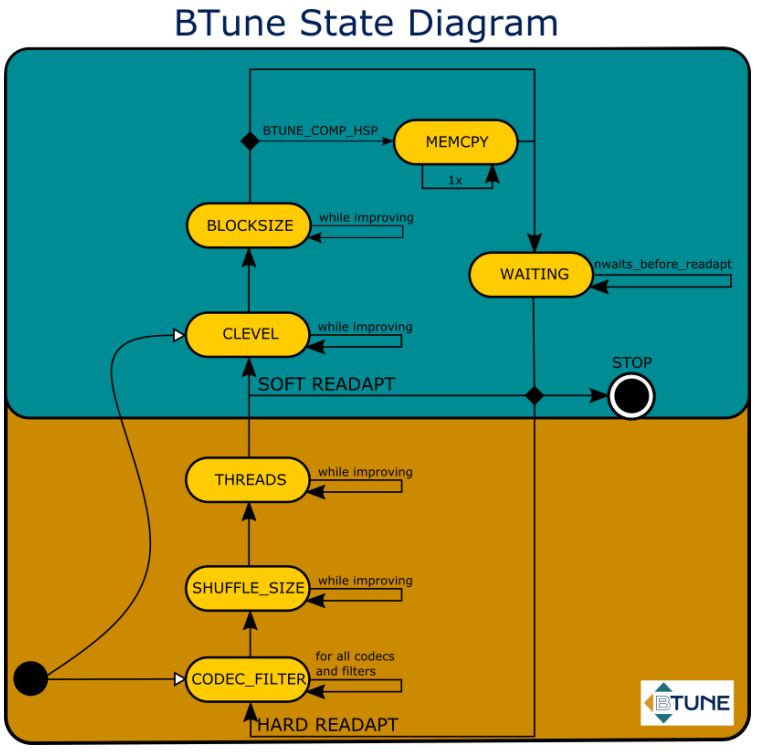
Lossless VS Lossy Compression With blosc2-grok

Grok lossless (690 KB)

Grok lossy (230 KB)

BTUNE

Making Compression Better

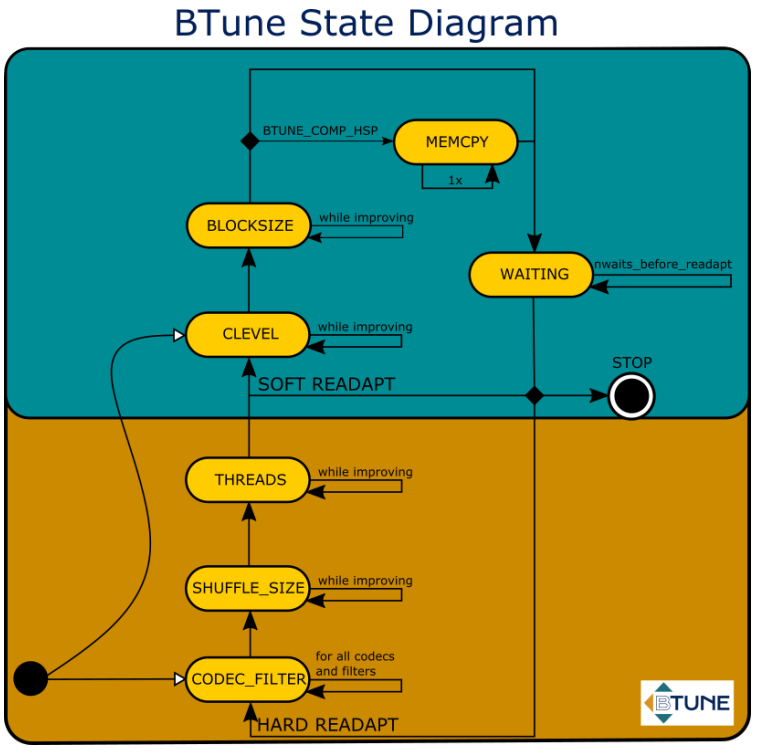


Fine Tuning Compression Performance

- BTune can fine tune the different parameters of the underlying Blosc2 storage to perform as best as possible.

- Can also be trained to find the best codec & filter with deep learning.

https://btune.blosc.org



Btune Operation Modes

- **Btune Genetic**: Use a genetic algorithm to test different parameters to select the best combination.

- **Btune Models (AI)**: The Blosc Development Team helps you find a Neural Net Model adapted to your datasets for faster operation.

- **Btune Studio**: Use the training package locally to generate your own models for your datasets by yourself.

Installing & Using the Btune Plugin

We provide with binary wheels:

```
$ pip install blosc2-btune
```

Using it is easy:

```
$ BTUNE_TRADEOFF=0.5 BTUNE_TRACE=1 python create_ndarray.py
```

Tutorials

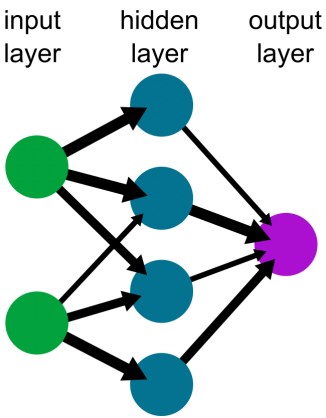
Basics: https://github.com/Blosc/Btune-Genetic-tutorial

Training: https://github.com/Blosc/Btune-tutorial

Btune Models

- Btune is trained for your datasets and can infer, in real time, the right combination of codec and filter that suits the requirements: favor speed, favor compression ratio, or a trade-off.

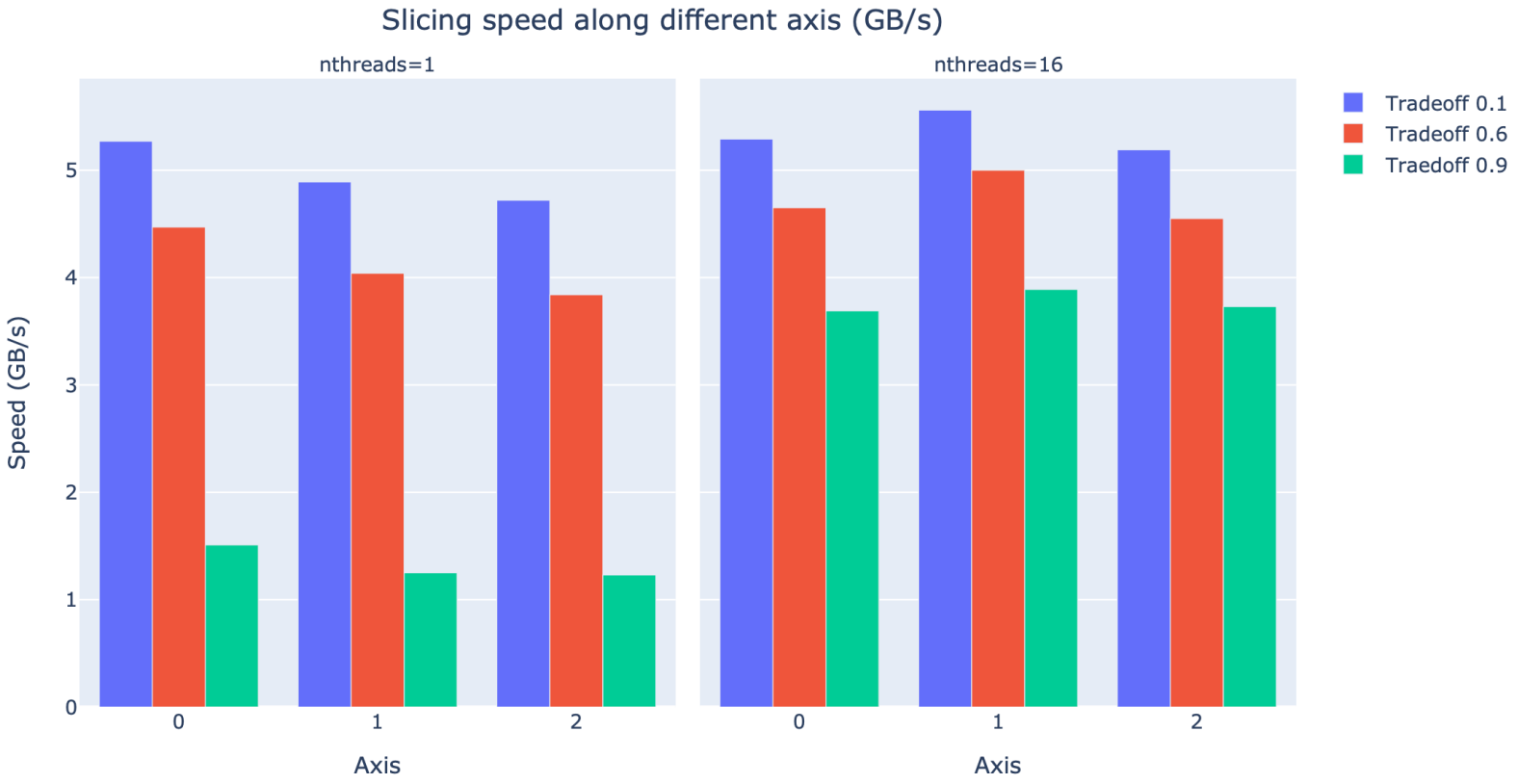
Neural Network Model



Predictions for Gaia dataset (decomp speed)

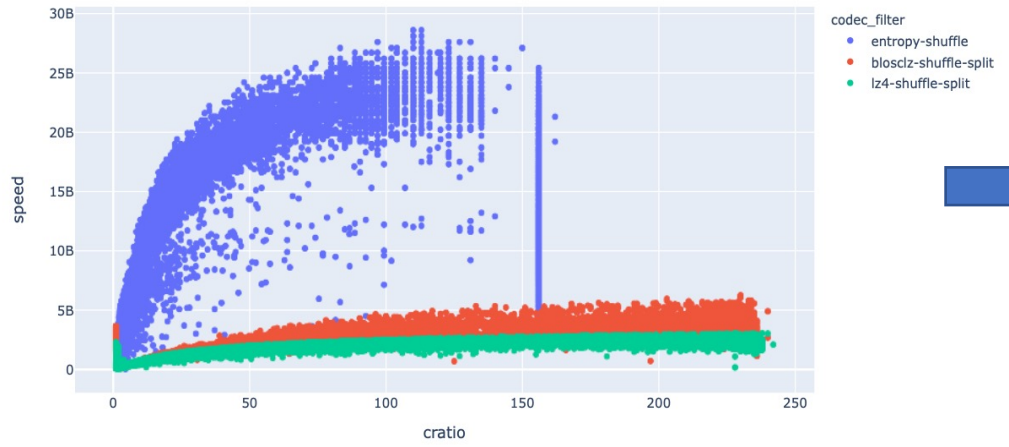
Tradeoff	Most predicted	Cratio	Cspeed	Dspeed
0.0	blosclz-nofilter-5	786.51	106.86	91.04
0.1	blosclz-nofilter-5	786.51	106.86	91.04
0.2	blosclz-nofilter-5	786.51	106.86	91.04
0.3	blosclz-nofilter-5	786.51	106.86	91.04
0.4	blosclz-nofilter-5	786.51	106.86	91.04
0.5	blosclz-nofilter-5	786.51	106.86	91.04
0.6	zstd-nofilter-9	8959.6	8.79	59.13
0.7	zstd-nofilter-9	8959.6	8.79	59.13
0.8	zstd-nofilter-9	8959.6	8.79	59.13
0.9	zstd-bitshuffle-9	10789.6	3.41	12.78
1.0	zstd-bitshuffle-9	10789.6	3.41	12.78

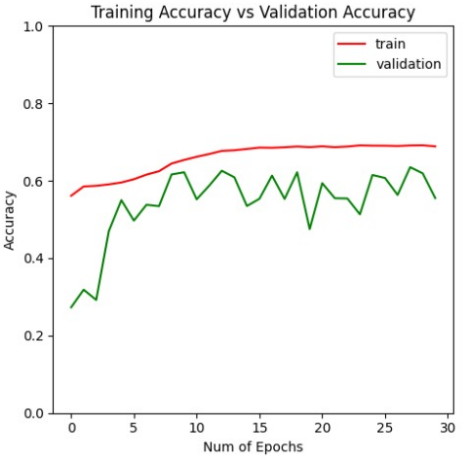
Models: Automatic Selection of Best Params



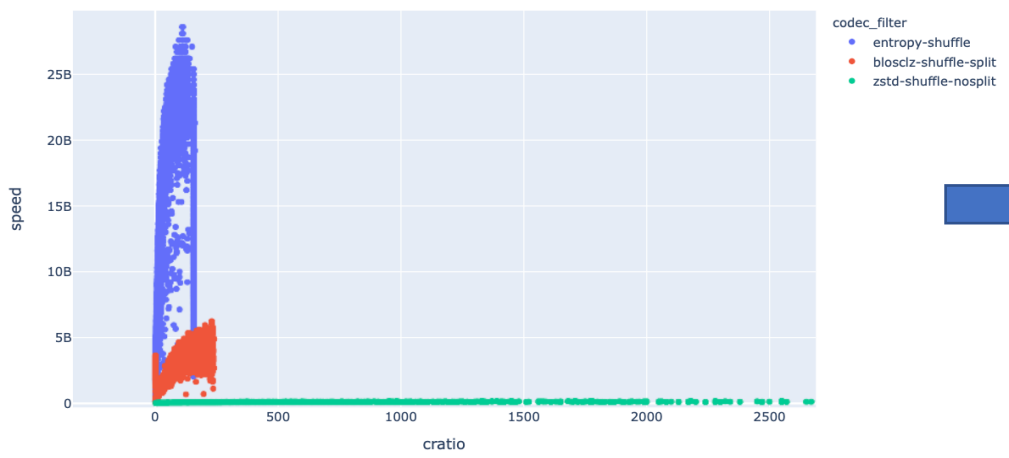
- Performance for different tradeoffs for decompressing
- Single threading is fine for tradeoffs favoring speed

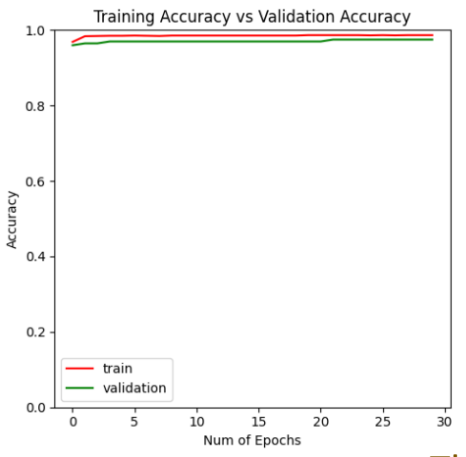
Better Predictions When Codecs Are Very Different





Codecs with similar features: Not good predictions





Codecs with different features: Much better predictions!

Thanks to Marta Aicart

Testimonials

"Blosc2 and Btune are fantastic tools that allow us to efficiently compress and load large volumes of data for the development of AI algorithms for clinical applications. In particular, the new NDArray structure became immensely useful when dealing with large spectral video sequences."

-- Leonardo Ayala, Div. Intelligent Medical Systems, German Cancer Research Center (DKFZ)

dkfz. DEUTSCHES KREBSFORSCHUNGSZENTRUM IN DER HELMHOLTZ-GEMEINSCHAFT

"Btune is a simple and highly effective tool. We tried this out with @LEAPSinitiative data and found some super useful spots in the parameter space of Blosc2 compression arguments! Awesome work, @Blosc2 team!"

-- Peter Steinbach, Helmholtz AI Consultants Team Lead for Matter Research @HZDR_Dresden

HELMHOLTZ AI | ARTIFICIAL INTELLIGENCE COOPERATION UNIT

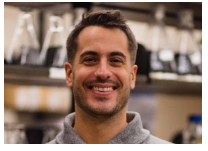
Conclusion

Blosc2 Highlights

- A novel way to **handle huge** (and potentially **sparse**) **NDim arrays** representing large volumes (tested up to 8 trillion cells).
- Can be extended via **dynamic plugins.**
- **Btune** allows for **automatic selection** of the best **compression parameters.**

Blosc2 represents a highly efficient and flexible tool for **compressing your data, your way**

Thanks to donors & contracts!



Jeff Hammerbacher

Without them, we could not have possibly put Blosc2 into production status: Blosc2 2.0.0 came out in June 2021; now at 2.11.3.

Recent NumFOCUS Award!

Project Sustainability Award in 2023 (Francesc Alted)

"In recognition of your outstanding contributions to the NumFOCUS Community."

Thank you! Questions?








We Make Compression Better