Blosc2 as an Accelerated Computation Engine

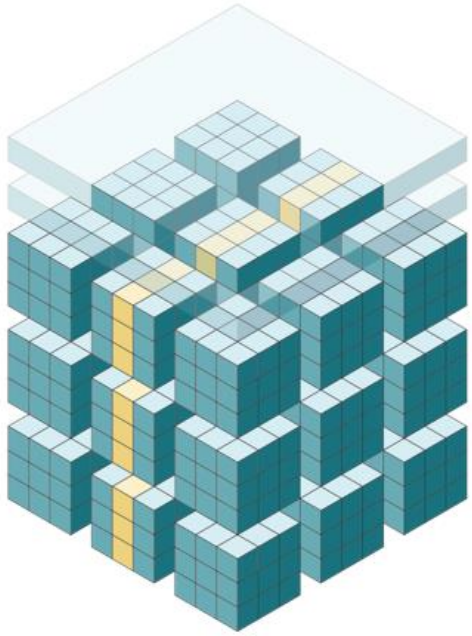
Francesc Alted / @FrancescAlted@masto.social

The Blosc Development Team / @Blosc2@fosstodon.org

CEO ironArray / francesc@ironarray.io

NumFOCUS Project Summit ’25

October 22nd 2025



Blosc2

Blosc1: Better compression for binary data

Blosc2: Blosc1 + accelerated computation for tensors

https://www.blosc.org/

Blosc2 for Computing

Blosc2 goes beyond compression

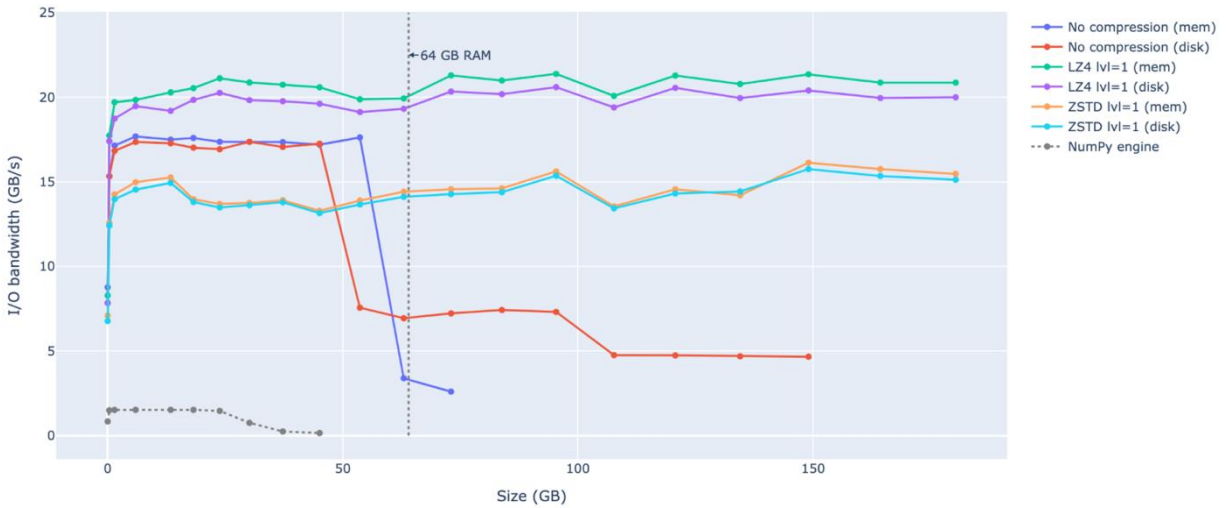
Accelerating Computation

- Optimized use of resources in modern CPUs
  - Multiple cores: use multi-threading
  - Hierarchical cache / memory / disk storage: use blocking
- Stand on the shoulders of efficient libraries
  - Numexpr (with optional Intel MKL)
  - Intel MKL, Accelerate, OpenBlas, BLIS (linear algebra, via NumPy)

Blosc2 does not support cluster computation
(use Dask/Spark for this)

Lazy Evaluation: Example

```
import numpy as np
import blosc2


N = 1000_000  # this will vary
dtype = np.float64
```

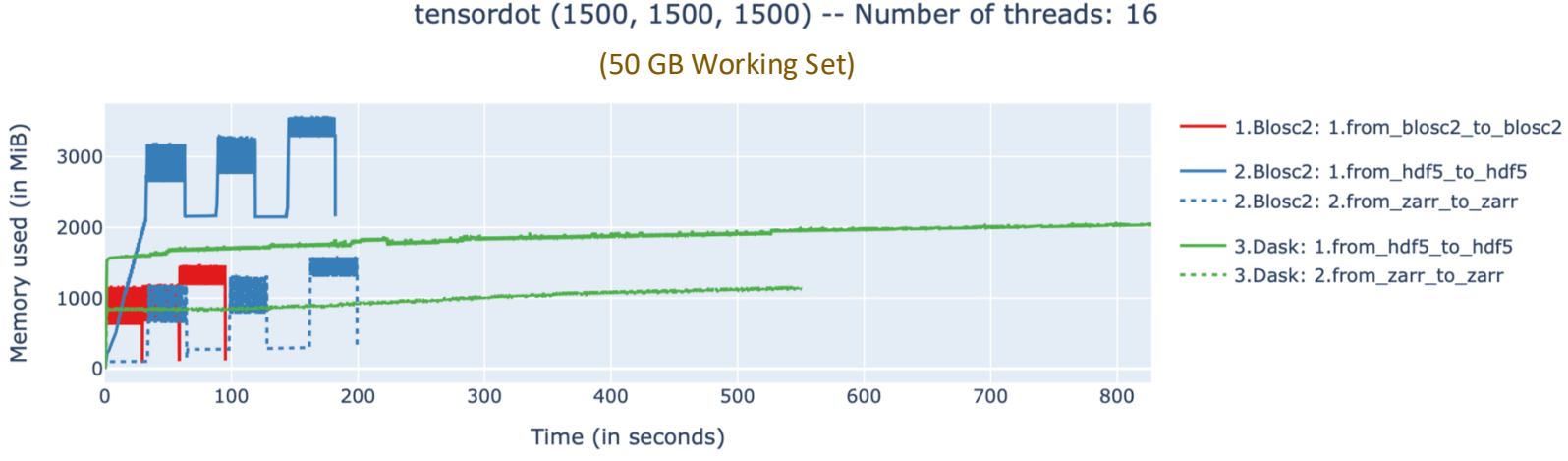



```
a = blosc2.linspace(0, 1, N * N, dtype=dtype, shape=(N, N))
b = blosc2.linspace(1, 2, N * N, dtype=dtype, shape=(N, N))
c = blosc2.linspace(0, 1, N, dtype=dtype, shape=(N,))  # 1D array; broadcasting is supported
out = np.sum(((a ** 3 + np.sin(a * 2)) < c) & (b > 0), axis=1).compute()
```

https://ironarray.io/blog/compute-bigger

Out-of-Core: Linear Algebra

```
# Assume a and b are large, on-disk blosc2/hdf5/zarr arrays

axis = (0, 1)

# Create a lazy expression object

lexpr = blosc2.lazyexpr("tensordot(a, b, axes=(axis, axis))")

# To execute it and save the result to a new persistent array:

out_blosc2 = lexpr.compute(urlpath="out.b2nd", mode="w")
```



https://www.blosc.org/posts/tensordot-pure-persistent/

Blosc2 Is Implementing the Array API

Problem:

- Python offers many choices of libraries and frameworks for numerical computing
- Many array containers: NumPy, Zarr, HDF5, PyTorch, JAX, Blosc2…

Mission:

- Offer a shared, interoperable API for all libraries and containers

The user can choose from a wealth of libraries,
without worrying about the API to use

Array API and Blosc2

- More than 100 functions, methods and attributes added so far (e.g. concat, stack, clip, sin, cos, expm1…)
- Reductions: sum, mean, std, var, all, any, min, max…
- Linear algebra: matmul, tensordot, vecdot, permut_dims…

Still a lot of work to do (especially in linear algebra)

Actively asking for funding to complete this effort

https://www.blosc.org/python-blosc2/development/roadmap.html

Synergies with NumFOCUS projects

The Blosc Development Team is always looking to collaborate with other projects to attract funding opportunities and help the wider Python ecosystem thrive!

Do you see Blosc2 helping your project to optimize storage or accelerate computations?

Let’s talk!

blosc@blosc.org

Thanks to Donors & Contractors!

Jeff Hammerbacher

Thanks! Questions?










blosc@blosc.org

Compress Better, Compute Bigger