Finanziato dall'Unione europea NextGenerationEU
Ministero dell'Università e della Ricerca
Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA
ICSC Centro Nazionale di Ricerca in HPC, Big Data and Quantum Computing

BrahMap: *A scalable map-making framework for the future CMB experiments*

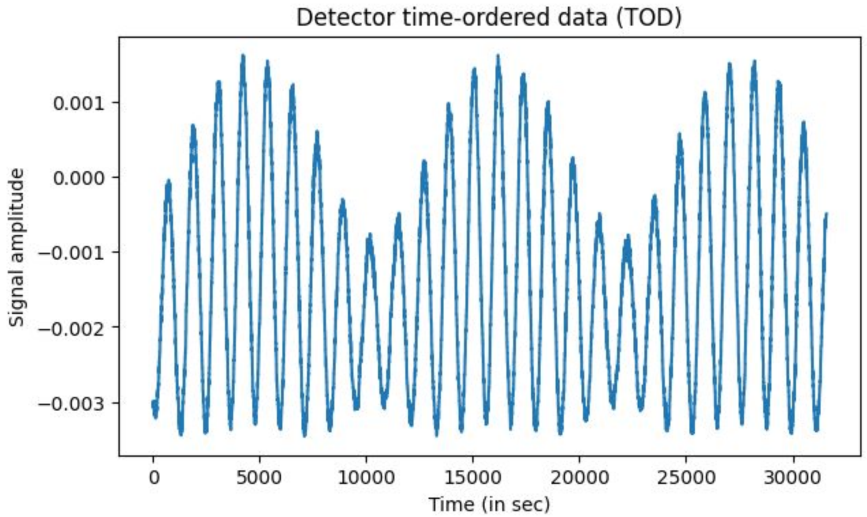
Avinash Anand[1], Giuseppe Puglisi[2]

[1]University of Rome "Tor Vergata", [2]University of Catania

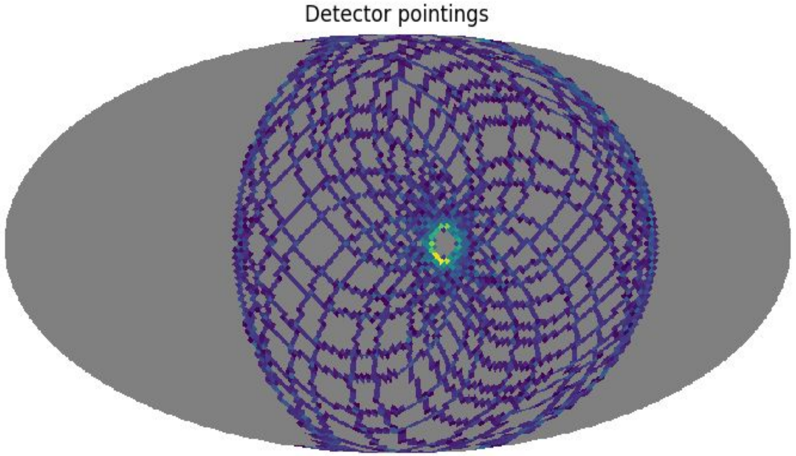
Spoke 3 II Technical Workshop, Bologna Dec 17 -19, 2024

Scientific Rationale

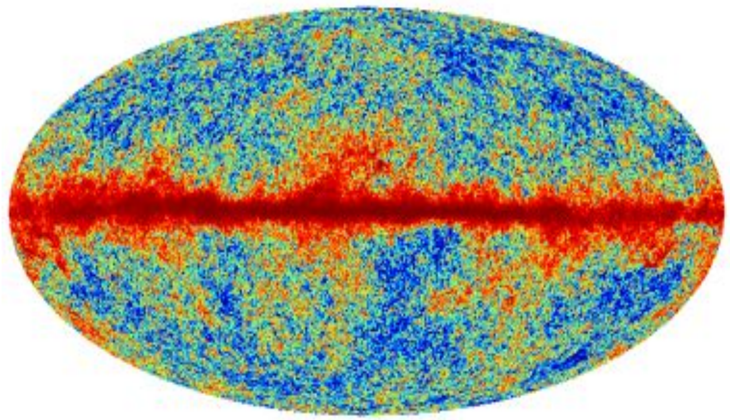
- Future CMB experiments: Targeting the B-mode polarization of CMB
- Detectors: $O(10^3)$ - $O(10^5)$ in number with a very high sampling rate
- Data acquisition: ~250 TB (from space) to ~10 PB (from ground)
- First step of analysis: Reduction of time-series data to sky maps *aka* Map-making
- Map-making goals:
  - Reduction of enormous amount of data in a reasonable timeframe
  - Mitigation of instrumental systematics
  - Removal of both un-correlated and correlated noise



+



→



- First step of analysis: Reduction of time-series data to sky maps *aka* Map-making
- Map-making goals:
  - Reduction of enormous amount of data in a reasonable timeframe
  - Mitigation of instrumental systematics
  - Removal of both un-correlated and correlated noise

Technical Objectives, Methodologies and Solutions

Data model of the sky signal

$$d_{p,t} = I_p + Q_p \cos 2\phi_t + U_p \sin 2\phi_t + n_t$$

- $d_{p,t}$ → Signal measured by the detector
- ϕ_t → Detector polarization angle
- I_p, Q_p, U_p → CMB stokes parameters
- n_t → Un-correlated and correlated noise contribution

Data model in matrix equation form

$$d = P \cdot s + n$$

- d → Signal vector
- n → noise vector
- s → Sky map vector
- P → Pointing matrix (sparse matrix with 3 non-zero elements per row)

Minimizing the sum of square of residuals

$$S = (d - P \cdot \hat{s})^T N^{-1} (d - P \cdot \hat{s})$$

- $N \rightarrow n_t \times n_t$ Noise covariance matrix

Generalized least-square solution (GLS)

$$\hat{s} = (P^T N^{-1} P)^{-1} P^T N^{-1} d$$

Assuming the noise is stationary

Technical Objectives, Methodologies and Solutions

$$d = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{n_t-2} \\ d_{n_t-1} \\ d_{n_t} \end{bmatrix} \quad P = \begin{bmatrix} \dots & \dots & \dots & \dots & \dots & 1 & \cos 2\phi_{t_1} & \sin 2\phi_{t_1} & \dots \\ \dots & 1 & \cos 2\phi_{t_2} & \sin 2\phi_{t_2} & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & 1 & \cos 2\phi_{t_3} & \sin 2\phi_{t_3} & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & 1 & \cos 2\phi_{t_{n_t-2}} & \sin 2\phi_{t_{n_t-2}} & \dots \\ \dots & 1 & \cos 2\phi_{t_{n_t-1}} & \sin 2\phi_{t_{n_t-1}} & \dots & \dots & \dots & \dots & \dots \\ \dots & 1 & \cos 2\phi_{t_{n_t}} & \sin 2\phi_{t_{n_t}} & \dots & \dots & \dots & \dots & \dots \end{bmatrix} \quad s = \begin{bmatrix} I_1 \\ Q_1 \\ U_1 \\ \vdots \\ I_{N_p} \\ Q_{N_p} \\ U_{N_p} \end{bmatrix} \quad n = \begin{bmatrix} n_1 \\ n_2 \\ n_3 \\ \vdots \\ n_{n_t-2} \\ n_{n_t-1} \\ n_{n_t} \end{bmatrix}$$

Minimizing the sum of square of residuals

$$S = (d - P \cdot \hat{s})^T N^{-1} (d - P \cdot \hat{s})$$

- $N \rightarrow n_t \, x \, n_t$ Noise covariance matrix

Generalized least-square solution (GLS)

$$\hat{s} = \left(P^T N^{-1} P\right)^{-1} P^T N^{-1} d$$

Assuming the noise is stationary

Technical Objectives, Methodologies and Solutions

- **BrahMap** : **A scalable map-making framework for future CMB experiments**

- A modular and object-oriented map-making framework based on COSMOMAP2[1,2]
- Python3 interface with C++ backend for compute-intensive parts

- Optimization to squeeze the most out of the supercomputing resources
- Scalability across multiple computing nodes
- Offloading the computations to multiple GPUs

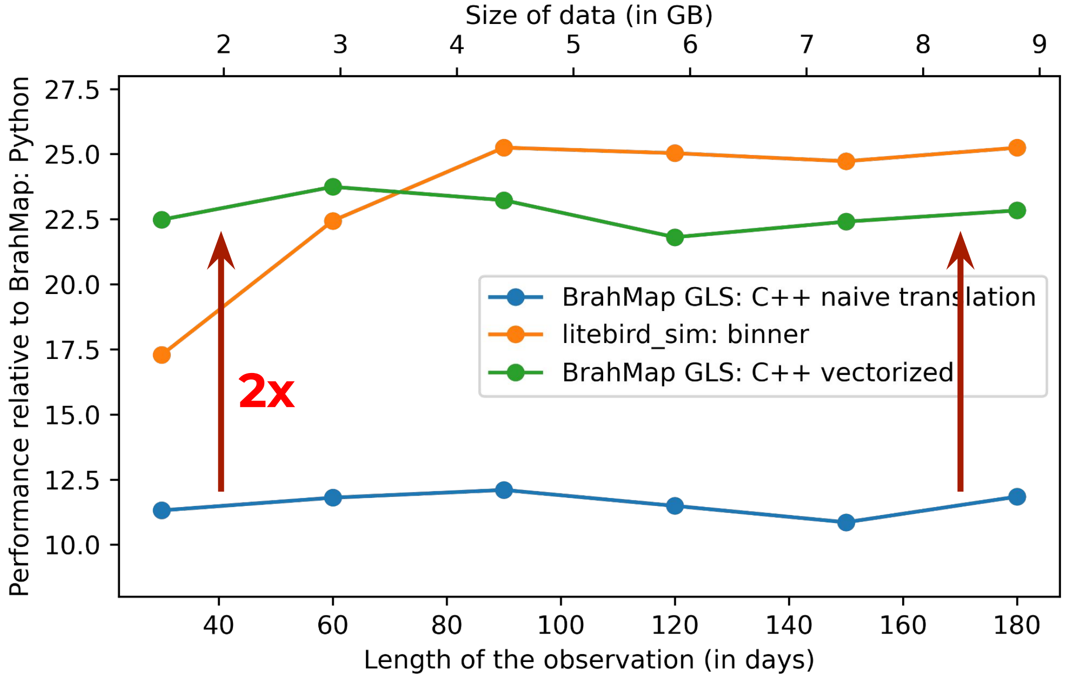
[1]Puglisi, G., et al. "*Iterative map-making with two-level preconditioning for polarized cosmic microwave background data sets - A worked example for ground-based experiments.*" A&A, 618 (2018) A62, https://doi.org/10.1051/0004-6361/201832710
[2]https://github.com/giuspugl/COSMOMAP2

Main Results: Until Elba

Milestone	Target
M6	- Conversion of code base from Python 2 to Python 3 - Debugging and validation
M7	- Writing computationally extensive parts to C++ with pybind11 - **11x** performance gain over previous version
M8	- Identification of bottlenecks - Code refactoring and vectorization - **2x** performance gain over previous version
M9	- Code parallelization with MPI

Main Results: Until Elba

- **22x** faster than the original Python version
- Comprehensive test suite: >180 tests
- Code refactoring employing the best-programming practices
- Code release: https://github.com/anand-avinash/BrahMap
- Documentation: https://anand-avinash.github.io/BrahMap

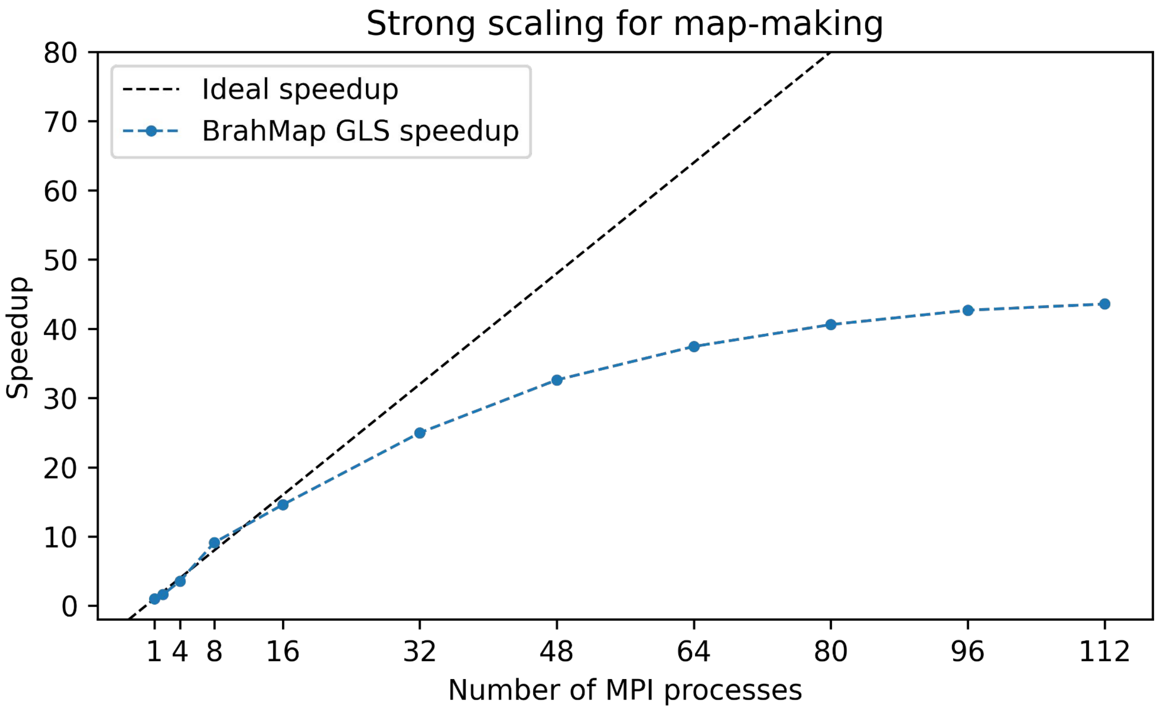


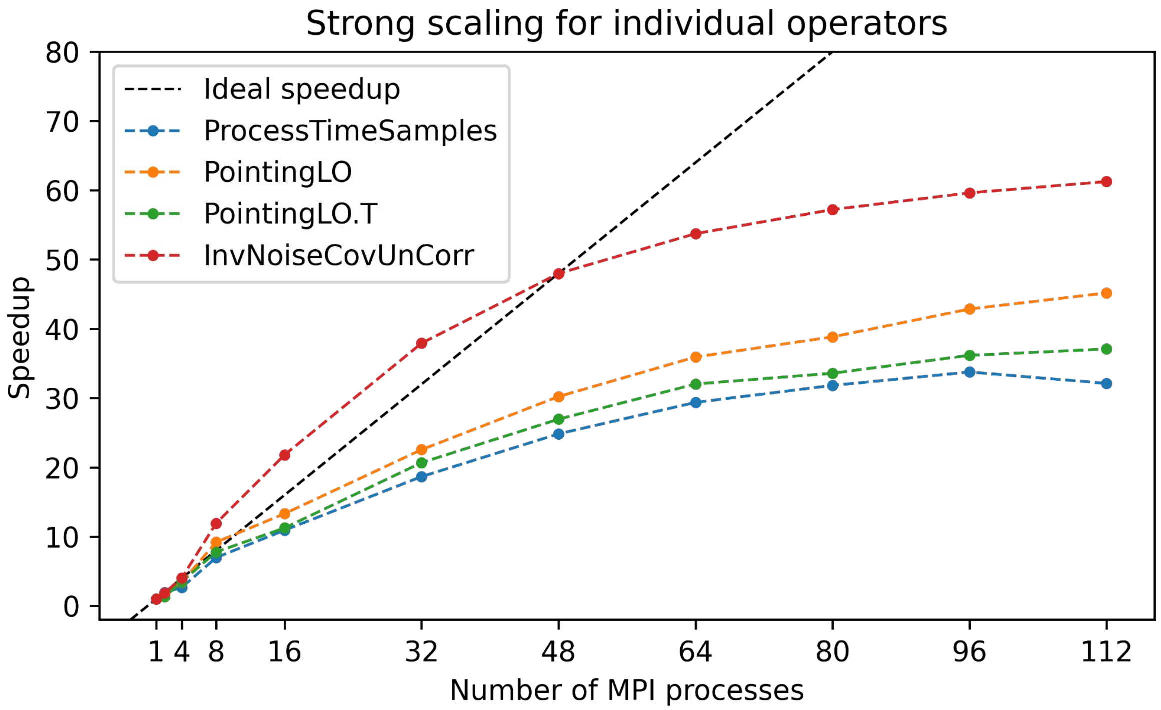
Main Results: Post Elba

- Parallelization with MPI
  - Passing MPI communicator from Python to C++
  - Handling MPI communications explicitly on C++ side
  - Updating Python installation script `setup.py`
    - Compatibility with both OpenMPI and MPICH
    - Seamless package installation

Main Results: Post Elba

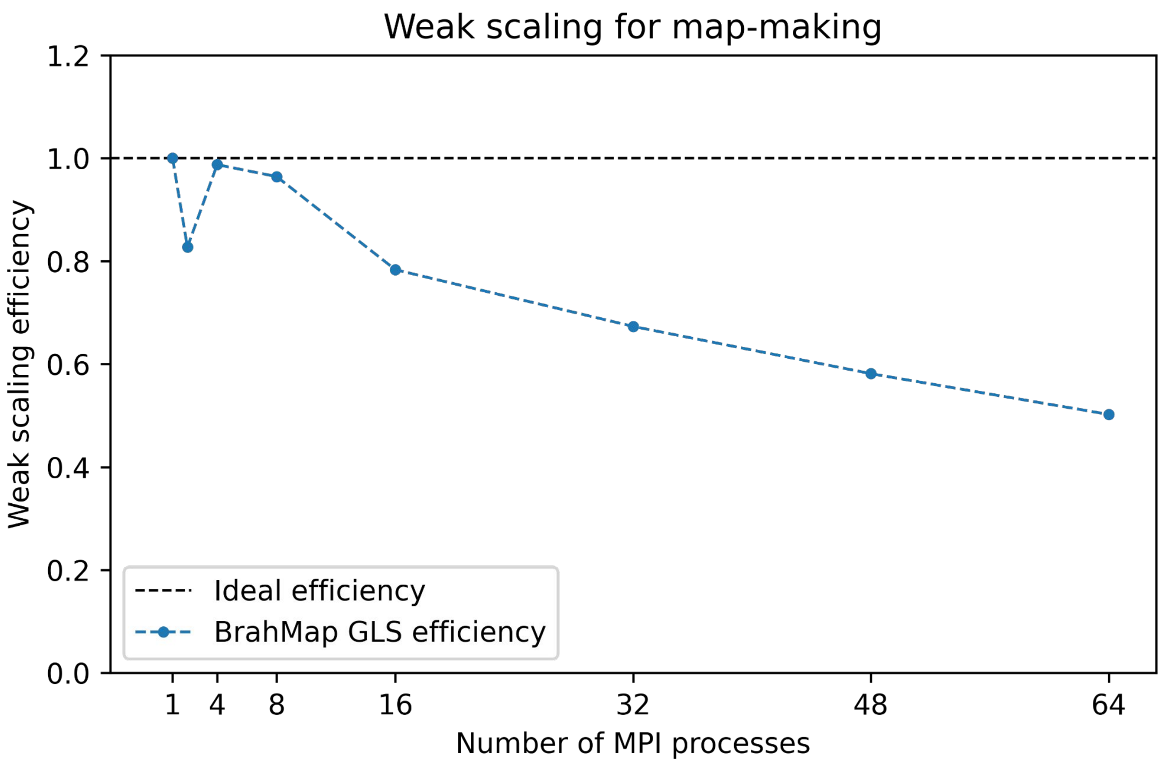
- MPI strong scaling
- Speedup has flat profile for large N_{procs} - problem size becomes very small

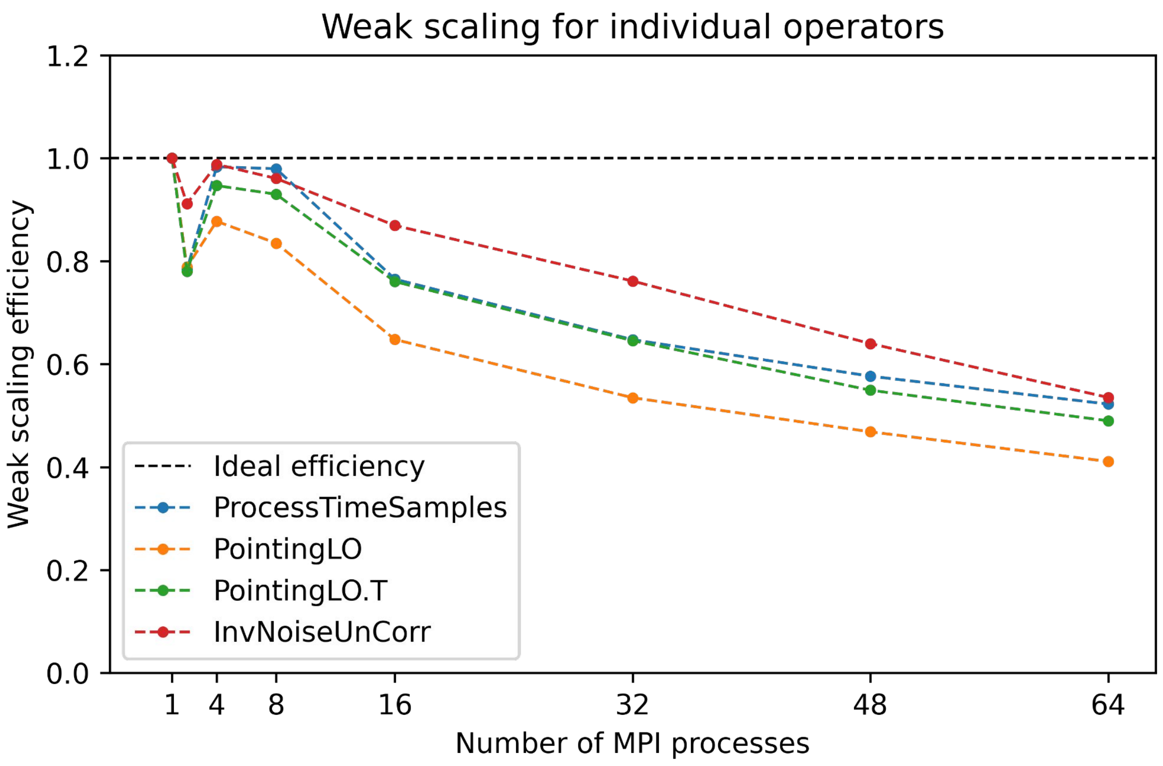




Main Results: Post Elba

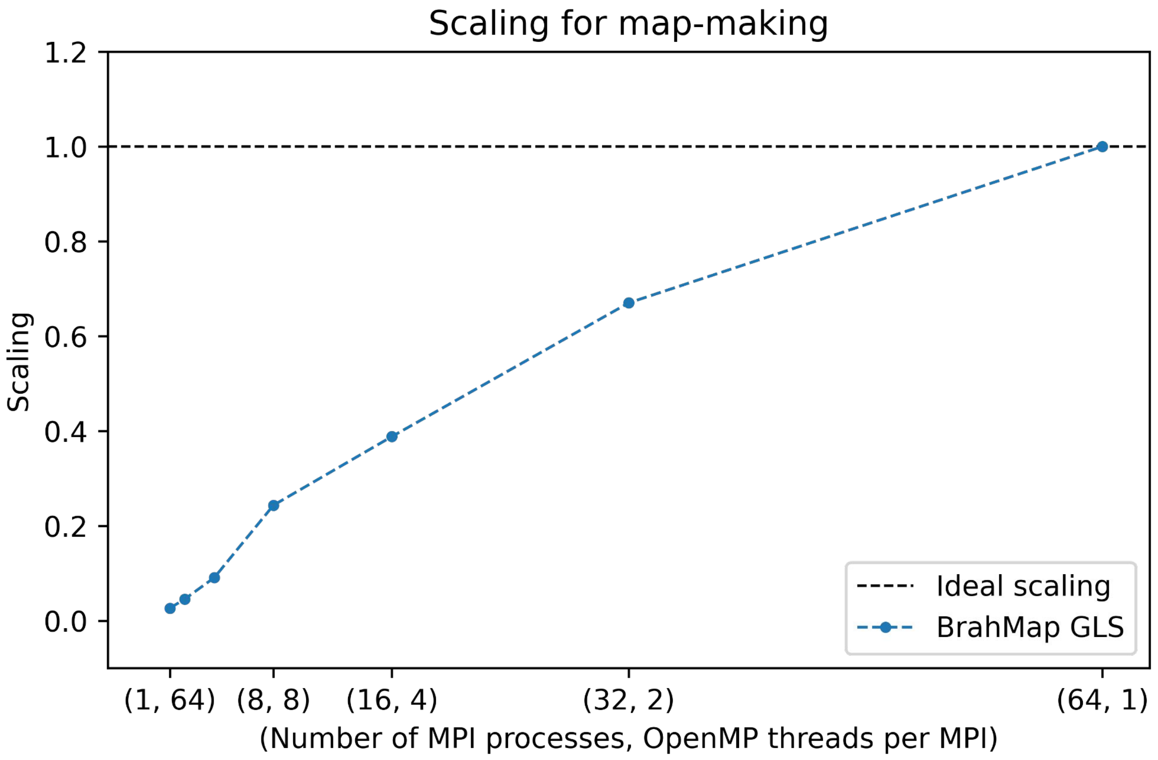
- MPI weak scaling
- Efficiency goes down - MPI communication overhead

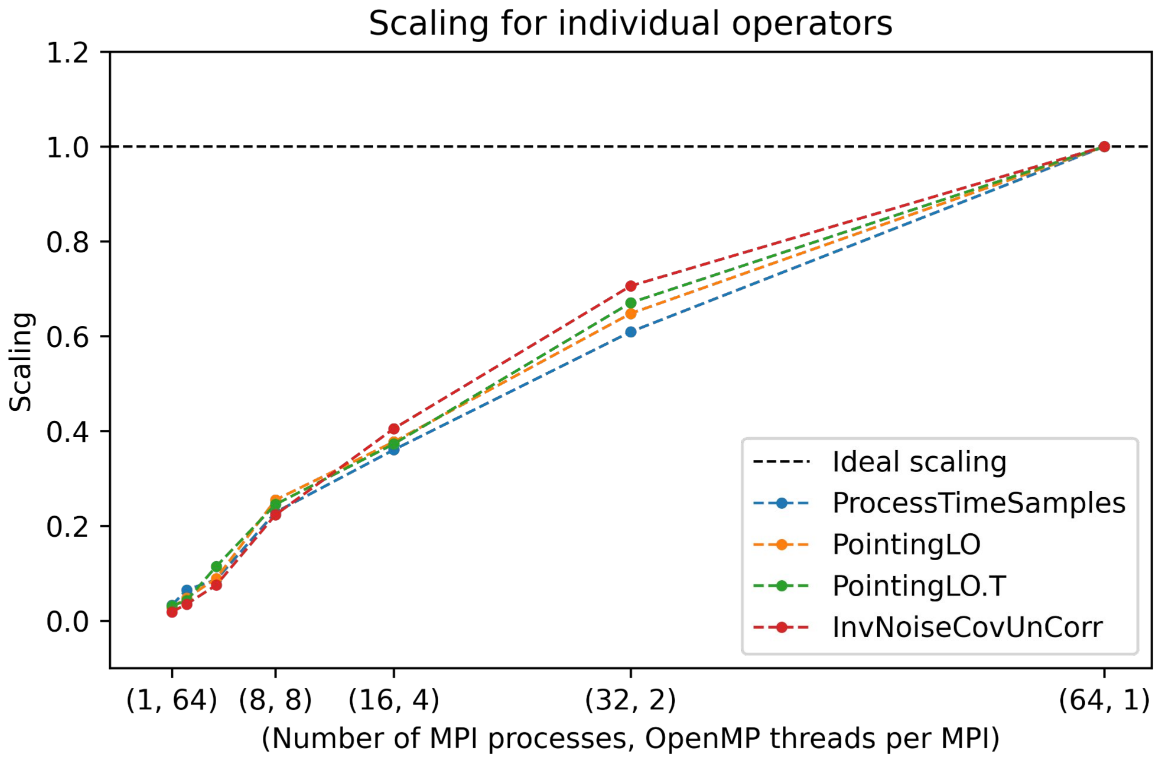




Main Results: Post Elba

- Hybrid parallelization with MPI and OpenMP
- Incompatible pair pair of pragmas - No multi-threading

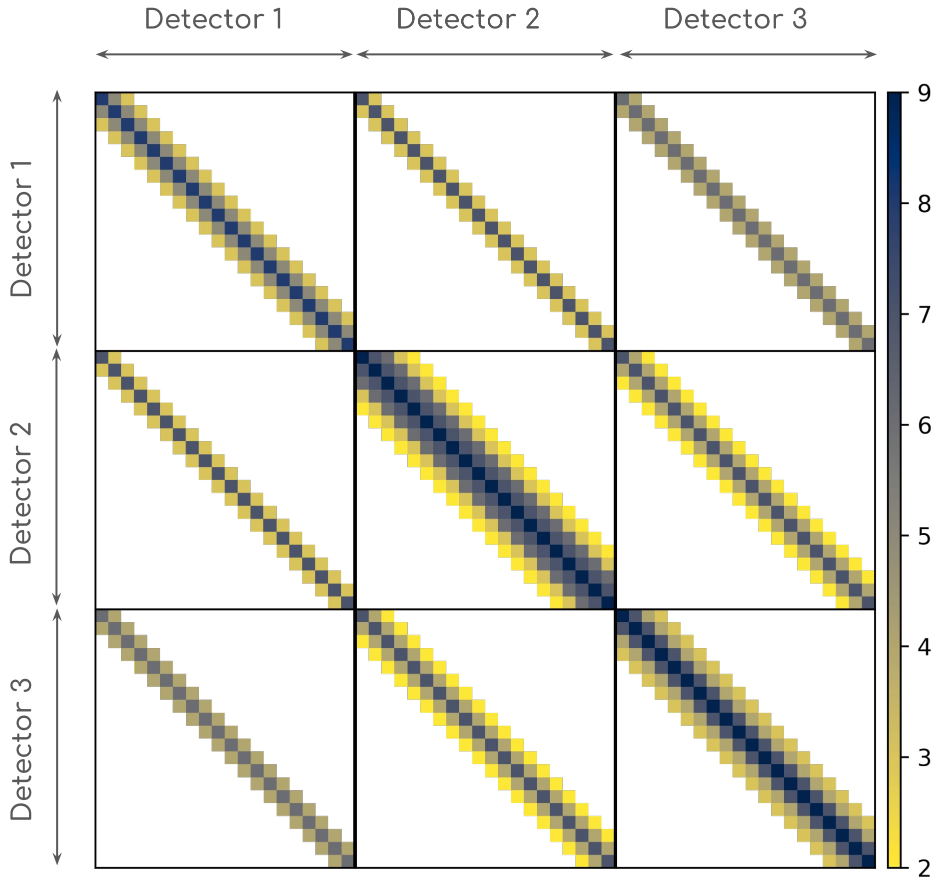




Ongoing Work

- Implementing more general noise covariance operators and their inverse

- A general noise covariance is symmetric Toeplitz block matrix
    - No explicit inversion
    - Plan:
        1. Implement an efficient Toeplitz operator
            - DFT based approach
            - DCT and DST based approach
        2. Invert the Toeplitz operator with suitable preconditioner



Final Steps

	- Documentation update
Now-Feb 2025	- **KPI**: Hybrid parallelization with OpenMP + MPI - Addressing the data race and other bottlenecks - **KPI**: Implementation of more general noise covariance

March 2025	- Profiling, benchmark and testing with large set of realistic simulations
	  - Function instrumentation on C++ side
	  - Time profiler on Python side

Apr - June 2025	- **KPI**: Offloading to GPUs
	  - cupy arrays on Python side
	  - Exposing cupy arrays to C++ with Array API