

Artifact Overview Document

JAX vs Kokkos Performance Comparison: A Magnetohydrodynamic Solver Case Study

Antoun T., Bourgeois R., Tremblin P., Kokh S., Bobin J.

Euro-Par 2026 — Artifact Evaluation Submission

Getting Started Guide

Artifact repository

```
git clone https://github.com/thierryantoun/jax-mhd
cd jax-mhd
git checkout mhd_jax
```

Operating system

Experiments were conducted on **Red Hat Enterprise Linux 9.6** (Jean-Zay, IDRIS) and equivalent RHEL-based systems on Adastra (CINES).

Component	NVIDIA / Jean-Zay	AMD / Adastra
Python	3.11.12	3.11.5
JAX	0.6.0	0.5.0
JAXlib	0.6.0	0.5.0
CUDA	12.8.0	ROCm 6.3.3
cmake	3.25.2	3.26.5
gcc	11.4.1	8.5.0
Kokkos	4.7.00	4.7.00

Installation

```
pip install --upgrade "jax[cuda12]" jaxlib # NVIDIA
pip install --upgrade "jax[rocm]" jaxlib # AMD
```

For ark-2-mhd (C++/Kokkos reference), clone and initialize submodules:

```
git clone https://github.com/rbourgeois33/ark-2-mhd
cd ark-2-mhd
git submodule update --init lib/kokkos
git submodule update --init lib/kokkos-tools
git submodule update --init lib/simd-math
git fetch origin
git checkout perf-experiment
```

Sanity check

```
python3 -c "import jax; print('JAX version:', jax.__version__); print('Devices:', jax.devices())"
```

Expected output on GPU: JAX version: 0.6.0 and [CudaDevice(id=0)]. On CPU: [CpuDevice(id=0)].

Configuration	Grid	GPU	JAX peak memory
CPU (validation)	64^3	Any CPU	< 1 GB
Full	256^3	NVIDIA V100 32 GB AMD MI250X 64 GB	~ 18 GB
Full	512×256^2	NVIDIA A100/H100 80 GB AMD MI300A 192 GB	~ 26–38 GB

Full results were obtained on Jean-Zay (IDRIS) and Adastra (CINES), which are not accessible to reviewers. The CPU mode (64^3) allows qualitative validation of all reported trends without a GPU.

Step-by-Step Instructions to Reproduce Results

Running the JAX solver

The repository contains three implementations corresponding to the three optimization steps of the paper: `2nd_order_while`, `2nd_order_scan`, and `2nd_order_scan_concat`. Each has its own `main.py` and `orszag-tang.ini`.

All experiments are launched via `run.sh` at the root of the repository:

```
./run.sh <mode> <impl>

# mode : cpu / gpu / all
# impl : while / scan / scan_concat (default: scan_concat)
```

Examples:

```
./run.sh cpu scan_concat      # if you run on cpu
./run.sh gpu while
./run.sh gpu scan
./run.sh gpu scan_concat
```

Results are saved in `results/<impl>_<mode>.txt`.

Running ark-2-mhd

Build for your target architecture following `README.md` in the git repo. Once built, run from the build directory using the `otang.ini` that is in the `perf-experiment` branch:

```
./main ../otang.ini ../IO_root.yml
```

Configuration

Each `orszag-tang.ini` uses the 3D Orszag-Tang vortex setup. To reproduce the full-scale results, set `resolution_x/y/z = 256` (or $512 \times 256 \times 256$ for A100/H100/MI300A) directly in the `.ini` file of the chosen implementation.

Figure	Command	Metric
Fig. 1 (initial JAX)	<code>./run.sh gpu while</code>	MCUPS
Fig. 2 (scan vs while)	<code>./run.sh gpu scan</code>	MCUPS
Fig. 5 (concatenation)	<code>./run.sh gpu scan_concat</code>	MCUPS
Fig. 6 (compile time)	any of the above	printed at startup
Fig. 7 (JAX vs Kokkos)	<code>./run.sh gpu scan_concat</code>	MCUPS

Mapping: commands → paper figures

Memory results for `ark-2-mhd` were obtained with `kp_memory_events` from `kokkos-tools`. JAX peak memory was measured with NVIDIA Nsight Systems.

Expected outputs and execution times

When a run completes successfully, the output looks like:

```
Using double precision
Execution time: 57.64
Simulation complete
Final time reached: 0.4535
nb_iterations: 1423
Total runtime: 57.64 seconds
Performance: 870 million cell updates per second (MCUPS)
Compilation time: 2.66 seconds
```

For full performance results across all architectures, refer to Figs. 1, 2, 5, and 7 of the paper. Performance variability of $\pm 5\%$ is expected due to hardware load differences.

Compliance Notes

Criterion	Status	Details
Root access	None	<code>pip install</code> only
Artifact size	< 1 GB	Source code only
Execution time	< 15 min	CPU mode (64 ³)
OS	Linux	Red Hat Enterprise Linux 9.6 (Jean-Zay)
HPC access	N/A	CPU mode provided for reviewers
Proprietary SW	None	JAX, Kokkos, Python — all open source

For questions during the Technical Clarification Window, please contact the authors via the submission system.