Celeritas performance

Amanda Lund

Computational Science Division
Argonne National Laboratory

HSF Detector Simulation on GPU Community Meeting
May 3–6, 2022

Celeritas core team:

Philippe Canal, Stefano Tognini, Tom Evans, Soon Yun Jun,
Guilherme Lima, Amanda Lund, Vincent Pascuzzi, Paul Romano

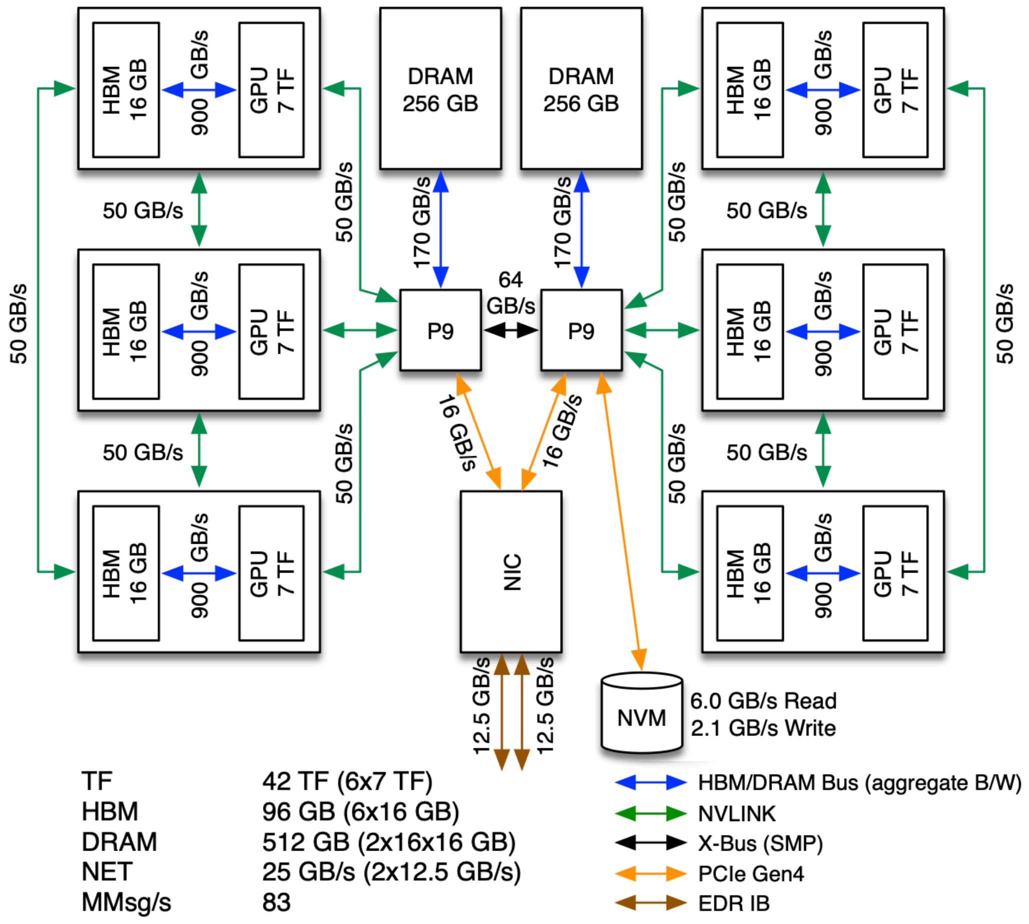
Performance metrics

- Figure of Merit (FOM) based on computation per unit energy (e.g. PFLOP / kWh)
  - Assumes perfect scalability of multicore
  - Does not account for theoretical peak FLOP rate
  - Requires a GPU/CPU performance equivalence of ~70
- Two use cases
  - Access to an existing GPU-accelerated system: **160 GPU core equivalence** (current MC performance achieved in ECP ExaSMR effort on Summit)
  - Justification to replace CPU with GPU system: **2x** (for same power consumption)

Hardware

- Focus on server-level hardware used in datacenters and DOE machines
- Performance results on Summit (OLCF)

System Specs	*Summit* Supercomputer
Peak system performance	200 PF
Number of nodes	4608
Node	2 IBM POWER9 CPUs 6 NVIDIA Tesla V100 GPUs
Memory per node	512 GB DDR4 + 96 GB HBM2
On-node interconnect	NVIDIA NVLink
System interconnect	Mellanox dual-port EDR IB (25 Gb/s)
Power consumption	13 MW



Summit node architecture

Summit User Guide — OLCF User Documentation
https://docs.olcf.ornl.gov/systems/summit_user_guide.html

Benchmark problem

- TestEm3 — simplified calorimeter
  - 50 alternating layers of $PbWO_4$ and lAr
  - 10,000 10 GeV electron primaries
- Equivalent configurations of Celeritas/Geant4/AdePT
  - No magnetic field
  - Disabled multiple scattering, energy loss fluctuations, Rayleigh scattering
  - Excludes initialization time
- No spline interpolation in Celeritas
  - ~3% performance penalty for Geant4 with spline
  - Compensate by using 8× cross section grid points: <2% slower

Initial performance results

- Per-node performance
- 1–2 batches of 6 simultaneous runs on Summit
  - CPU: multithreaded with 7 cores
  - GPU: one CPU core per GPU
- 40× faster with GPUs
  - Apples-to-apples: Celeritas CPU vs GPU
  - Similar order-of-magnitude improvement irrespective of code
  - 280 CPU core to GPU equivalence

Wall time per primary (ms)

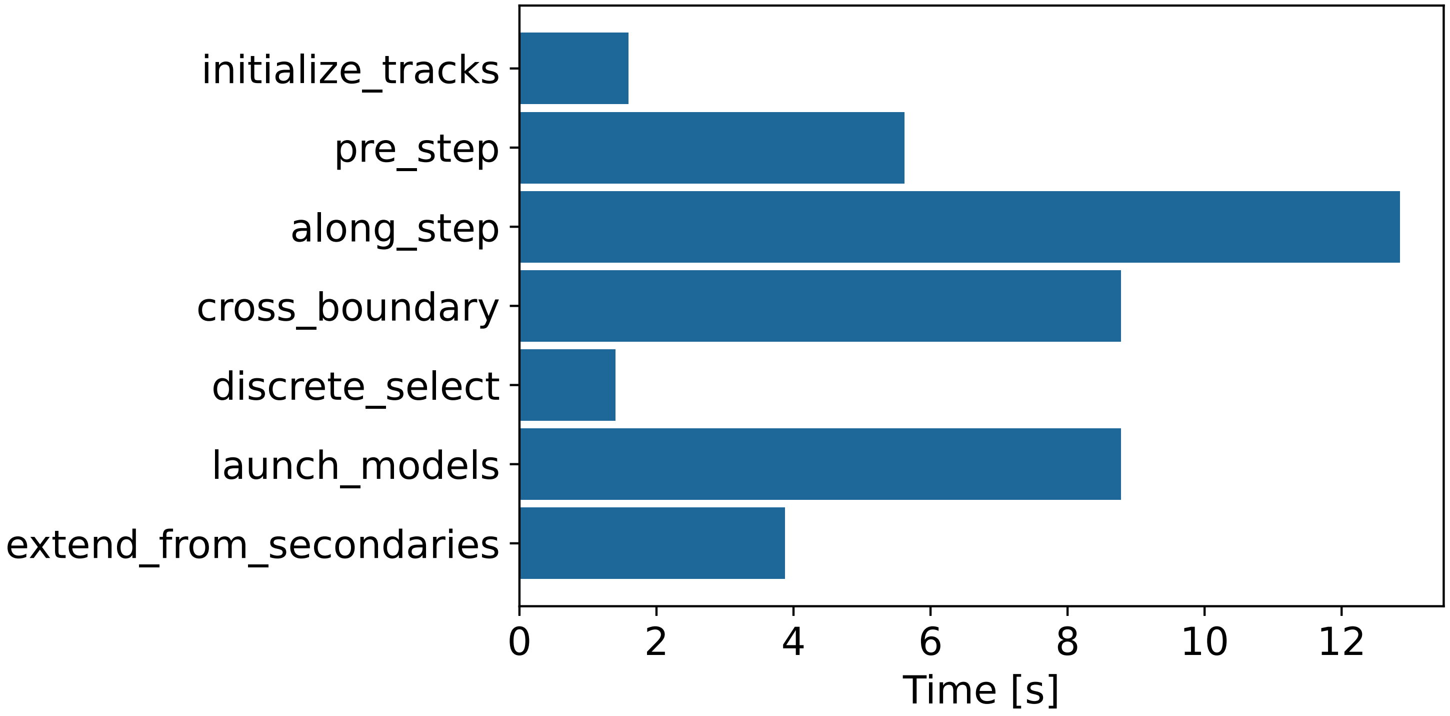
	geo	arch	mean	σ
Geant4 10.7.1	Geant4	CPU	2.9	0.1170
AdePT 68508ef7 (sethrj/adept/summit, 2 May 2022)	VecGeom	GPU	0.0850	0.0005
Celeritas 8d83ebab (29 Apr 2022)	ORANGE	CPU	2.09	0.0192
		GPU	0.046	0.0012
	VecGeom	CPU	1.95	0.0352
		GPU	0.0627	0.0004

Number of primaries per run

Geant4	Geant4	CPU	1E+04
AdePT	VecGeom	GPU	1E+05
Celeritas	ORANGE	CPU	1E+03
		GPU	1E+05
	VecGeom	CPU	1E+03
		GPU	1E+05

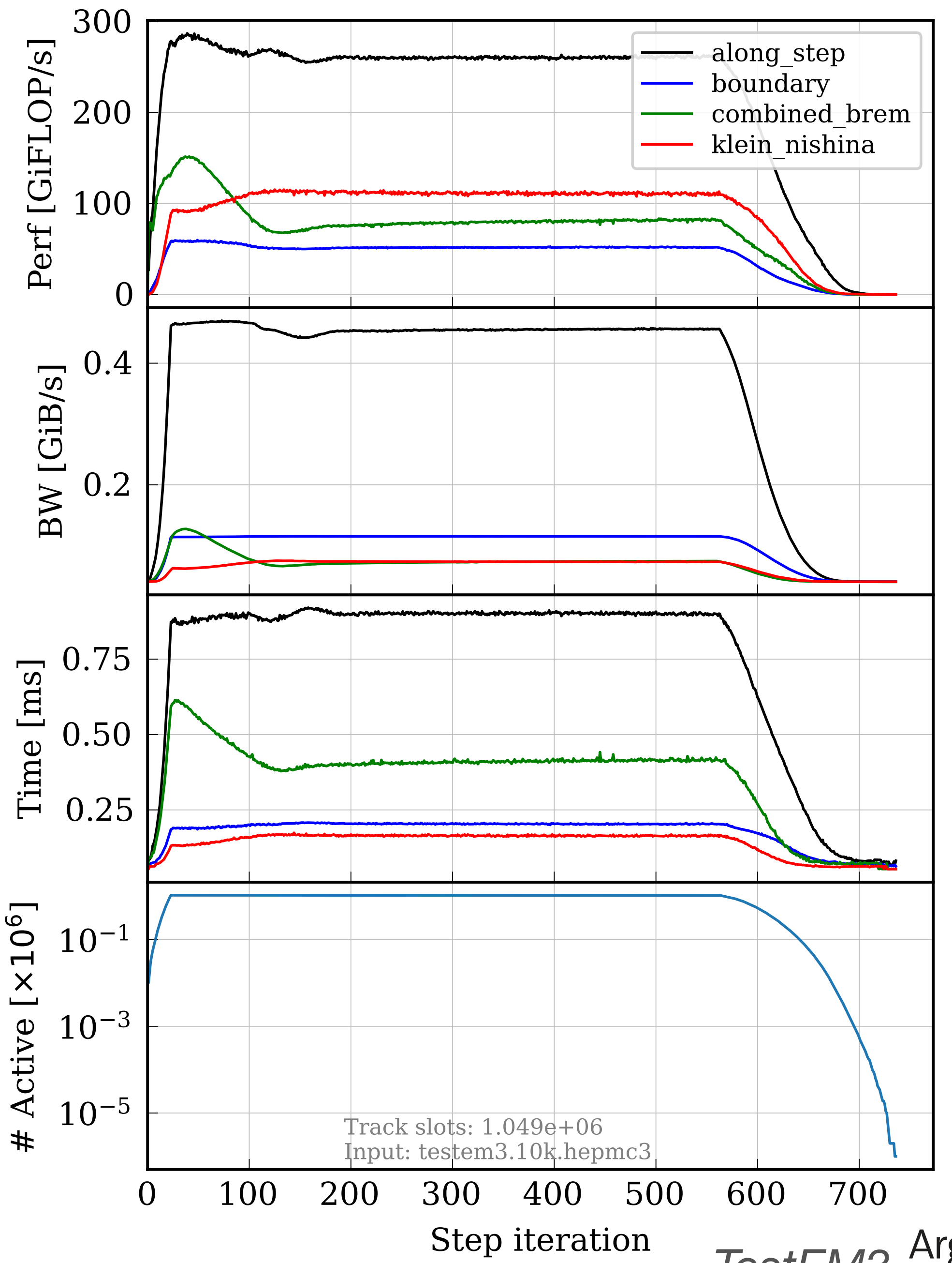
Detailed timing

```
extend_from_primaries              ▷ Copy primaries to device, create track initializers
while Tracks are alive do
    initialize_tracks              ▷ Create new tracks in empty slots
    pre_step                       ▷ Sample mean free path, calculate step limits
    along_step                     ▷ Propagation, slowing down
    boundary                       ▷ Cross a geometry boundary
    discrete_select                ▷ Discrete model selection
    launch_models                  ▷ Launch interaction kernels for applicable models
    extend_from_secondaries        ▷ Create track initializers from secondaries
end while
```



Performance analysis

- NVIDIA Tesla V100 (Summit)
  - Peak theoretical performance: 7.8 TFLOP/s (double-precision)
  - Peak theoretical bandwidth: 900 GB/s
- Bandwidth use and FLOP/s vary across kernels
- All well below peak theoretical capability of V100
- Memory latency bound



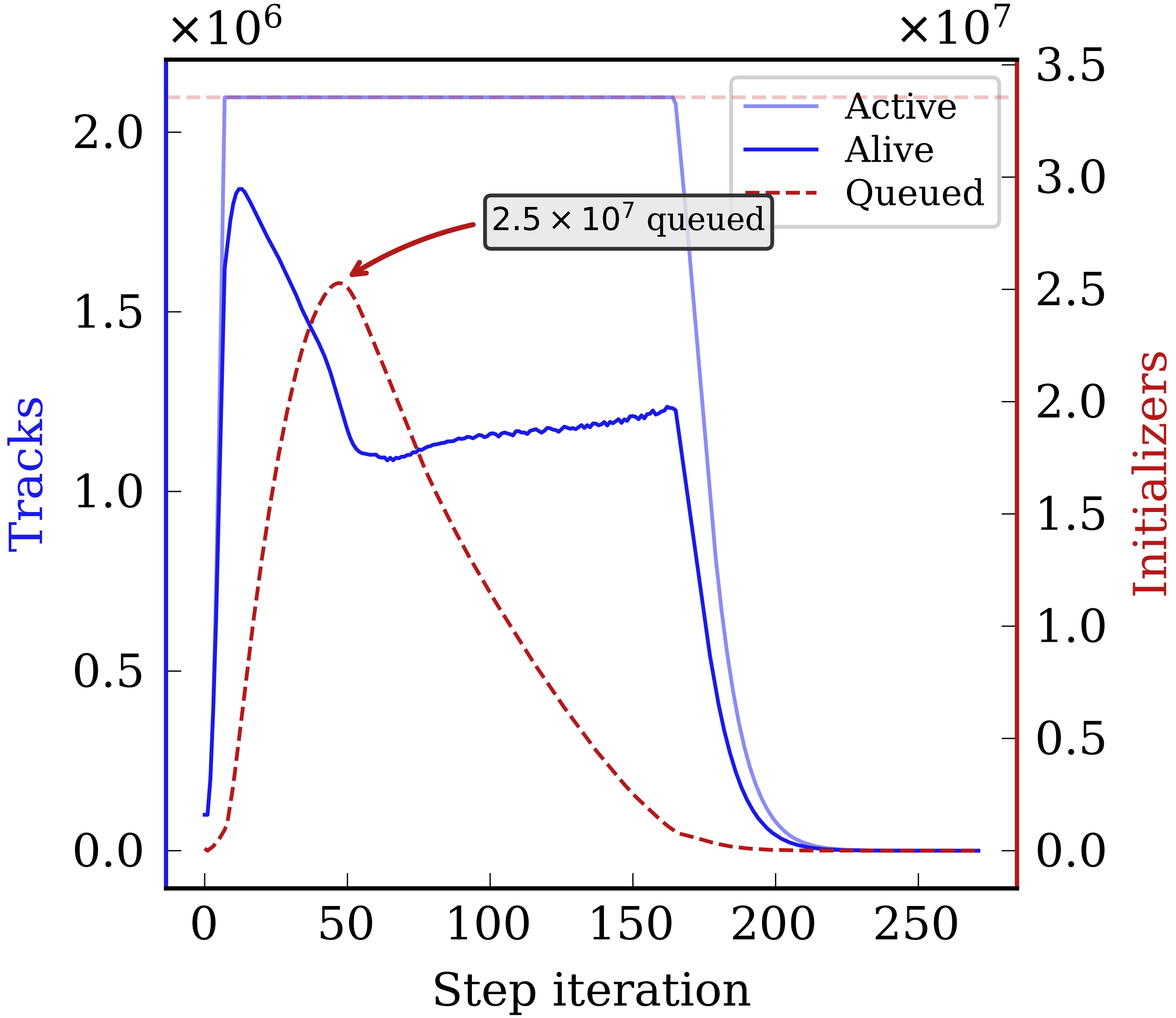
TestEM3

Caveats

- Single element per material
- Hardwired for no magnetic field
- Different PRNGs (Celeritas uses XORWOW, Geant4 uses MixMax)
- Non-optimal algorithm and data structures for CPU parallelism
- No optimization work performed yet
- Simulation results are reproducible, but have arbitrary track IDs
- Experimental workflows may need to batch multiple events together to achieve peak GPU performance

Stepping behavior (memory)

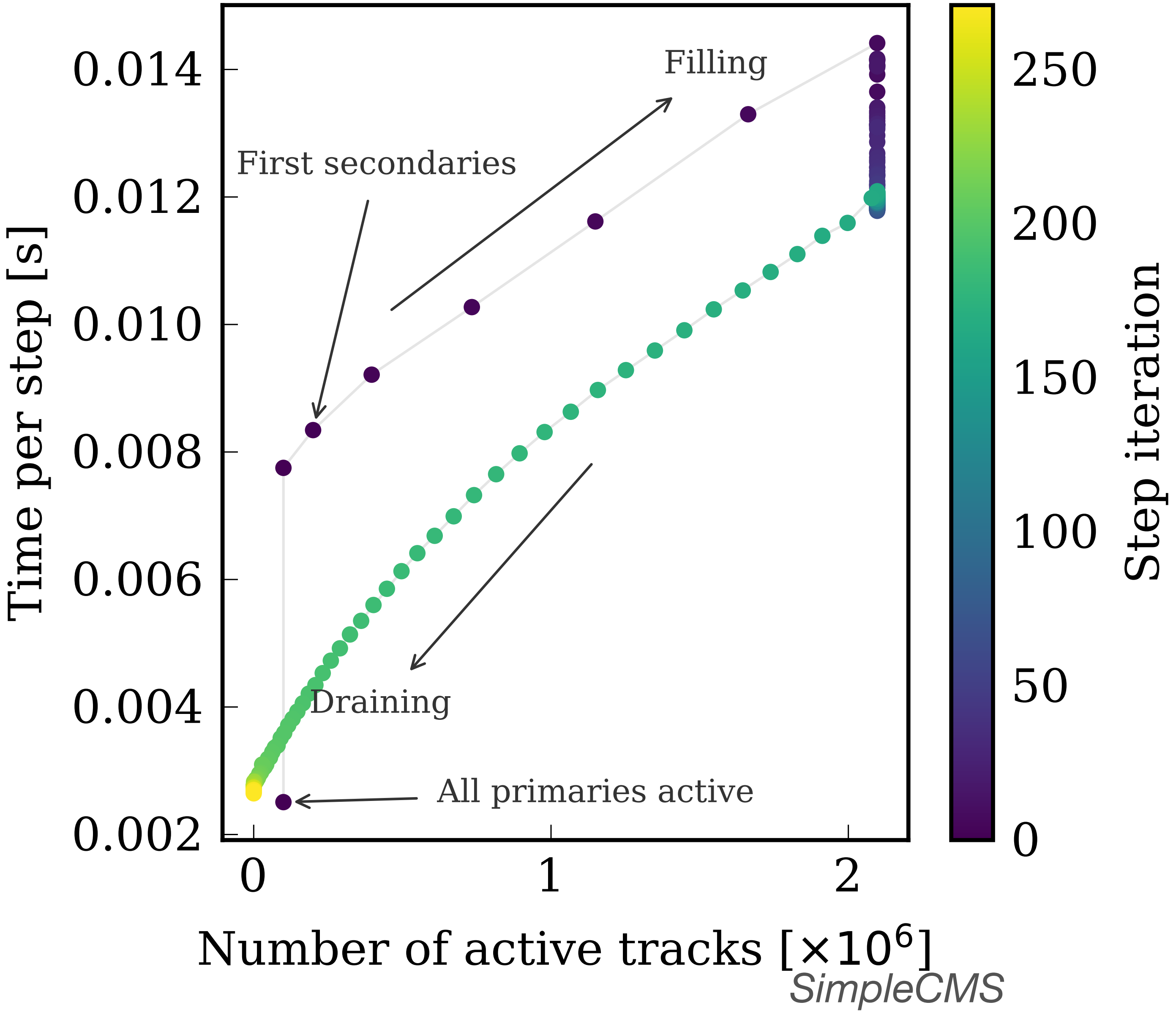
- Tracks: particles in flight
- Initializers: queued secondaries
- Memory capacity limits both



SimpleCMS

Stepping behavior (time)

- Distribution of particles and energies changes per step
- Some physics are faster than others



Performance optimizations — profiling still preliminary

- ☒ Preallocate one secondary per track
  - Saw ~13% speedup in Klein-Nishina demo-interactor…
  - *but ~8% slowdown in transport loop!*
- ☐ Partition rather than mask threads
- ☐ Sort track initializers by energy, particle type…
- ☐ Optimize kernel size