Julia-based study of SoftKiller performance in high-pileup conditions

BROWN
Department of Physics

Ema Dimitrova (Brown University), Matt LeBlanc (Brown University)

Scan to view the full code on GitHub

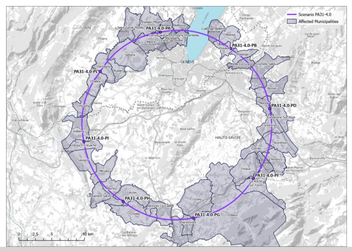
SCAN ME

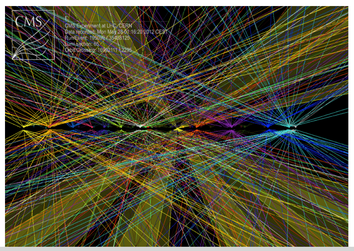
Overview

- We reimplemented SoftKiller, a pileup mitigation algorithm originally developed in C++ (FastJet repository), in Julia.
- Initial runtime analysis shows that the Julia implementation of the Tiled jet clustering strategy achieves notable speed gains in cluster reconstruction across a wide range of pileup levels compared to the C++ implementation.
- All Julia code and examples developed for this study are publicly available in the JuliaHEP/JetReconstruction.jl GitHub repository.

Background

- Pileup refers to the phenomenon of multiple proton–proton collisions occurring simultaneously within the same bunch crossing at a hadron collider such as the LHC at CERN.
- In this study, pileup events are simulated by overlaying minimum bias events generated with Pythia 8.315, without detector simulation.
- **Pile-up levels at different future colliders:**
  - LHC Run 3 (Today): mu~60
  - HL-LHC Run 5: mu~200
  - 27 TeV HE-LHC: mu~460
  - 100 TeV FCC-hh: mu~1000





Why julia?

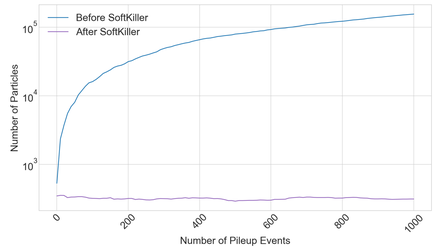
- Julia is a relatively new programming language designed for high-performance scientific computing.
- It combines the ease of use and expressiveness of high-level languages like Python with the performance of low-level languages like C++.
- Stewart et al. [1] have compared the N2Tiled jet finding strategy between C++ and Julia, finding that the latter implementation is ~6% more efficient and motivating further study of Julia for high-energy physics applications.

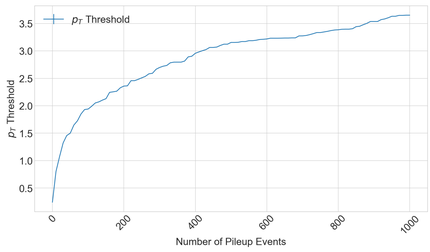
SoftKiller Performance

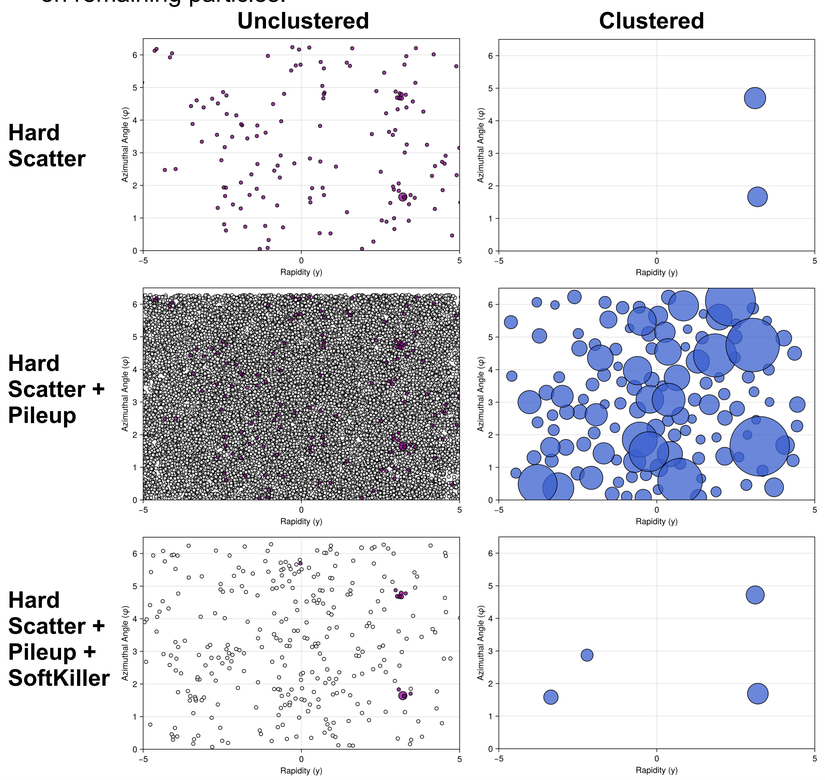
SoftKiller is a pile-up mitigation algorithm that removes low-momentum particles, improving the resolution and clarity of reconstructed high-energy jets.

Algorithm:

1. Grid partition: Divide detector rapidity–azimuth plane into equal cells.
2. Compute cell sums: Sum transverse momentum (p_T) per cell.
3. Thresholding: Find median cell p_T to use as cutoff.
4. Soft removal: Discard particles with p_T below cutoff.
5. Clustering: Run standard anti-k_T on remaining particles.







Software

- The Julia implementation was developed using the JuliaHEP/JetReconstruction.jl package, which includes the SoftKiller algorithm and jet clustering tools.
- Event data in HepMC3 format was processed via HepMC3.jl, with command-line configurability provided by ArgParse.jl.
- The full source code and environment configuration are available at: **github.com/JuliaHEP/JetReconstruction.jl**

Profiling

Benchmark Setup:

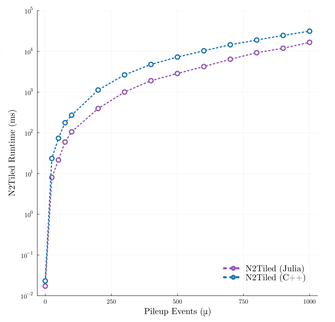
- Each data point is averaged over 100 trials.
- Benchmarks run on the Brown CMS Tier-3 cluster (Intel® Xeon® Gold 5318Y CPU @ 2.10GHz).

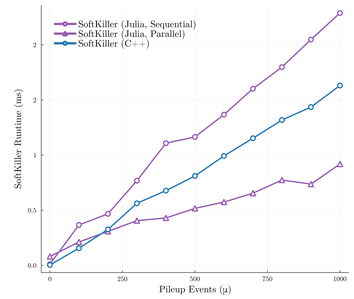
N2Tiled Jet Clustering Performance:

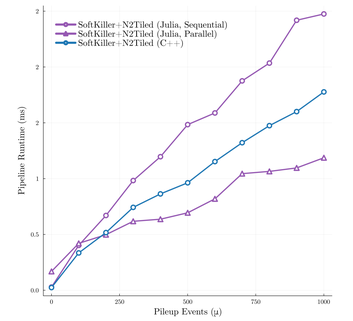
- Julia is ~25% faster at μ = 0 (0.017 ms vs. 0.023 ms).
- At μ = 1000 (154k+ particles), Julia is ~50% faster (16.6 s vs. 31.2 s).
- Results are consistent and promising for high-pileup levels (μC, EIC, FCC-hh).

SoftKiller (SK) Performance and Jet Clustering:

- Baseline C++ implementation is faster than Julia's single-thread version.
- Easy multithreading capabilities in julia have potential to improve performance, e.g. multithreading particle-to-tile assignment loop in softkiller can surpass C++ performance with minor software changes.
- Jet clustering with SK shows linear runtime scaling as pileup increases since SK reduces particle multiplicity to a constant output, avoiding the quadratic cost seen in unmitigated clustering.







References

[1] Stewart, G.A. et al. (2024) Polyglot jet finding, *EPJ Web Conf.* 295 (2024) 05017, proceedings of CHEP 2024. https://arxiv.org/abs/2309.17309
[2] Cacciari, M., Salam, G.P. and Soyez, G. (2016) SoftKiller, a particle-level pileup removal method, *Eur.Phys.J.C* 75 (2015) 2, 59. https://arxiv.org/abs/1407.0408
[3] Cacciari, M., Salam, G.P. and Soyez, G. (2011) Fastjet User Manual, *Eur.Phys.J.C* 72 (2012) 1896. https://arxiv.org/abs/1111.6097

Thanks to the Brown University Undergraduate Teaching and Research Awards (UTRA) program for supporting this work during Spring 2025!