Swift-HEP Generators: Status Update

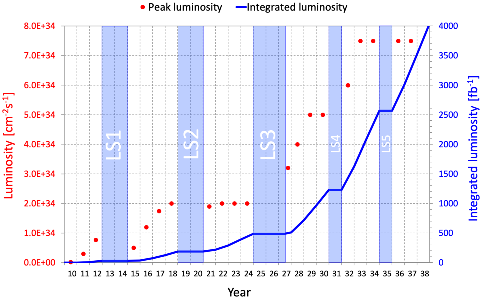
Enrico Bothmann (Göttingen), Andy Buckley (Glasgow),
Ilektra Christidi (UCL), Christian Gütschow (UCL),
Stefan Höche (FNAL), Max Knobbe (Göttingen),
Marek Schönherr (Durham)

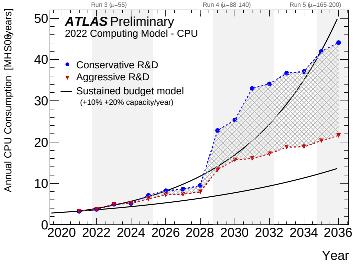
Swift-HEP workshop

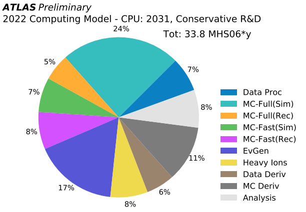
30 March 2023

Expected computing requirements

- latest update to the projected evolution of computing resources sees cost of event generation on par with detector simulation
- LHC measurements in danger of being limited by Monte Carlo statistics







[CERN-LHCC-2022-005]

Systematic profiling

- Most event generation CPU spent on multi-leg NLO calculations [JHEP 08 (2022) 089]
  - used for main Standard Model processes
  - relevant to measurements and searches alike
  - extremely large event sample sizes

Systematic profiling

- Most event generation CPU spent on multi-leg NLO calculations [JHEP 08 (2022) 089]
  - used for main Standard Model processes
  - relevant to measurements and searches alike
  - extremely large event sample sizes
- Study CPU performance of MEPS@NLO calculations for $e^+e^- + 0, 1, 2j$@NLO+$3, 4, 5j$@LO and $t\bar{t} + 0, 1j$@NLO+$2, 3, 4j$@LO with Sherpa 2.2.11, OpenLoops 2.1.2 and LHAPDF 6.2.3 using VTune 2021.7.1

Systematic profiling

- Most event generation CPU spent on multi-leg NLO calculations [JHEP 08 (2022) 089]
  - used for main Standard Model processes
  - relevant to measurements and searches alike
  - extremely large event sample sizes
- Study CPU performance of MEPS@NLO calculations for $e^+e^- + 0, 1, 2j$@NLO$+3, 4, 5j$@LO and $t\bar{t} + 0, 1j$@NLO$+2, 3, 4j$@LO with Sherpa 2.2.11, OpenLoops 2.1.2 and LHAPDF 6.2.3 using VTune 2021.7.1
- performance dependence on the number of multiweights studied using different setups:
  - baseline MEPS@NLO (no variations)
  - + $\mathrm{EW_{virt}}$ corrections
  - + 7-point variations of factorisation and renormalisation scales in matrix element and parton shower
  - + 100 (1000) NNPDF3.0nnlo replicas

Systematic profiling

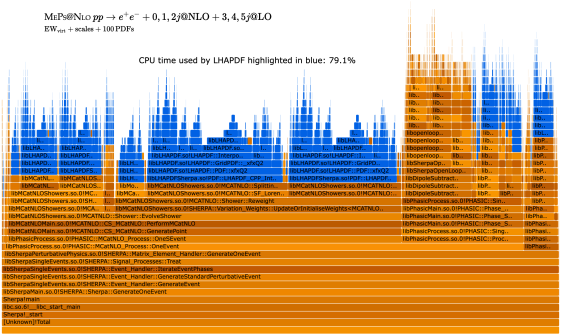
- Most event generation CPU spent on multi-leg NLO calculations [JHEP 08 (2022) 089]
  - used for main Standard Model processes
  - relevant to measurements and searches alike
  - extremely large event sample sizes
- Study CPU performance of MEPS@NLO calculations for $e^+e^- + 0, 1, 2j$@NLO$+3, 4, 5j$@LO and $t\bar{t} + 0, 1j$@NLO$+2, 3, 4j$@LO with Sherpa 2.2.11, OpenLoops 2.1.2 and LHAPDF 6.2.3 using VTune 2021.7.1
- performance dependence on the number of multiweights studied using different setups:
  - baseline MEPS@NLO (no variations)
  - + $\mathrm{EW}_{\mathrm{virt}}$ corrections
  - + 7-point variations of factorisation and renormalisation scales in matrix element and parton shower
  - + 100 (1000) NNPDF3.0nnlo replicas
- detailed write-up presented in [EPJC 82 (2022) 12]

Initial profiling exercises

- first generator CPU profiling done by Tim Martin suggested per-event CPU dominated by LHAPDF

- graph shows PDF calls highlighted in blue (using LHAPDF 6.2.3)

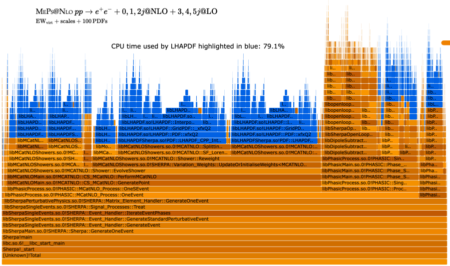
- maybe not completely surprising: multiweights originally not designed with hundreds of variations in mind [**EPJC 76 (2016) 11**]

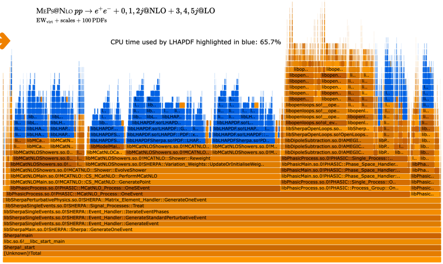


- explore two approaches in parallel: make LHAPDF faster and rework LHAPDF call strategy

Impact of new LHAPDF

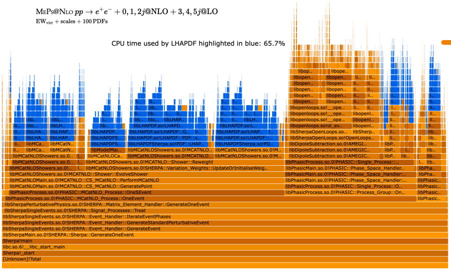
- ➜ ATLAS V+jets setup **overall 30% faster** using new LHAPDF release
  - ➜ switching from old ATLAS production default v6.2.3 to new v6.4.0 release



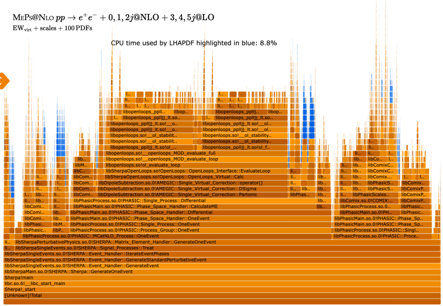


Internal restructuring in Sherpa 2.2.12: the pilot run

- ➜ perform the unweighting using a minimal setup and once an event is accepted, rewind RNG state and re-calculate accepted event using all the bells and whistles
- ➜ **achieves factor 5 speed improvement** for ATLAS setup (using LHAPDF 6.4.0 yields additional 6% speed-up)
- ➜ pilot run reduces CPU spent on evaluating PDFs to below 10%

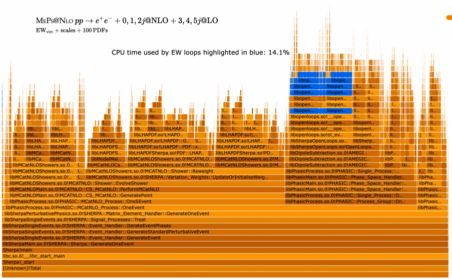


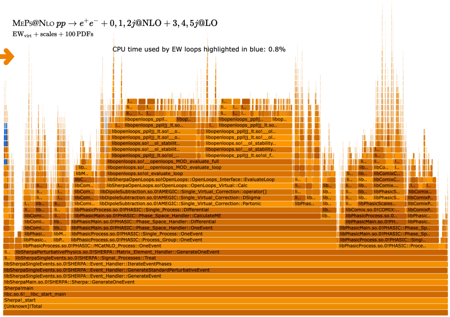
➜



Internal restructuring in Sherpa 2.2.12: the pilot run

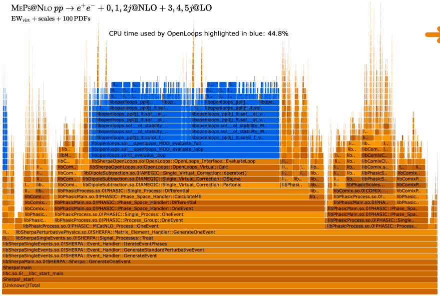
- CPU spent on calculating EW one-loop amplitudes going from 19% down to 0.8% when using the pilot run with the ATLAS V+jets setup
- nevertheless, $\sim$40% of the CPU still spent on calculating QCD loops

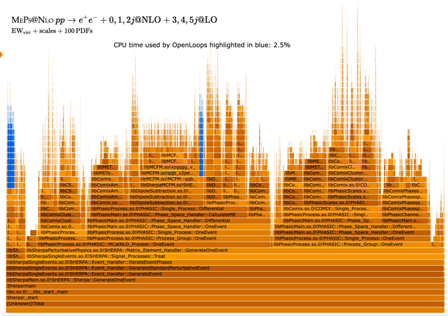




Analytic vs numerical QCD loop amplitudes

- employ analytic one-loop amplitudes (if available) in the pilot run using Sherpa-MCFM interface [EPJC 81 (2021) 12]
- yields **additional ∼35% speed improvement** for the V+jets setup





Full suite of improvements

➜ study the impact of different improvements sequentially:

Full suite of improvements

- study the impact of different improvements sequentially:
  - improved interpolation strategies in LHAPDF (6.2.3 → 6.4.0)

Full suite of improvements

- study the impact of different improvements sequentially:
  - improved interpolation strategies in LHAPDF (6.2.3 $\rightarrow$ 6.4.0)
  - replace full-colour spin-correlated S-MC@NLO algortihm with leading-colour spin-averaged $\langle LC \rangle$-MC@NLO (`NLO_CSS_PSMODE` 0 $\rightarrow$ 1)
    - this disables subleading colour corrections in the parton shower

Full suite of improvements

- study the impact of different improvements sequentially:
  - improved interpolation strategies in LHAPDF (6.2.3 $\rightarrow$ 6.4.0)
  - replace full-colour spin-correlated S-MC@NLO algortihm with leading-colour spin-averaged $\langle LC \rangle$-MC@NLO (`NLO_CSS_PSMODE` 0 $\rightarrow$ 1)
    - this disables subleading colour corrections in the parton shower
  - introduce pilot run in Sherpa (2.2.11 $\rightarrow$ 2.2.12)

Full suite of improvements

- study the impact of different improvements sequentially:
  - improved interpolation strategies in LHAPDF (6.2.3 $\rightarrow$ 6.4.0)
  - replace full-colour spin-correlated S-MC@NLO algortihm with leading-colour spin-averaged $\langle LC \rangle$-MC@NLO (`NLO_CSS_PSMODE` 0 $\rightarrow$ 1)
    - this disables subleading colour corrections in the parton shower
  - introduce pilot run in Sherpa (2.2.11 $\rightarrow$ 2.2.12)
  - defer leading-colour MC@NLO until after the unweighting (`NLO_CSS_PSMODE` 1 $\rightarrow$ 2)

Full suite of improvements

- study the impact of different improvements sequentially:
  - improved interpolation strategies in LHAPDF (6.2.3 $\rightarrow$ 6.4.0)
  - replace full-colour spin-correlated S-MC@NLO algortihm with leading-colour spin-averaged $\langle LC \rangle$-MC@NLO (`NLO_CSS_PSMODE` 0 $\rightarrow$ 1)
    - this disables subleading colour corrections in the parton shower
  - introduce pilot run in Sherpa (2.2.11 $\rightarrow$ 2.2.12)
  - defer leading-colour MC@NLO until after the unweighting (`NLO_CSS_PSMODE` 1 $\rightarrow$ 2)
  - use analytic one-loop amplitudes from MCFM in pilot run

Full suite of improvements

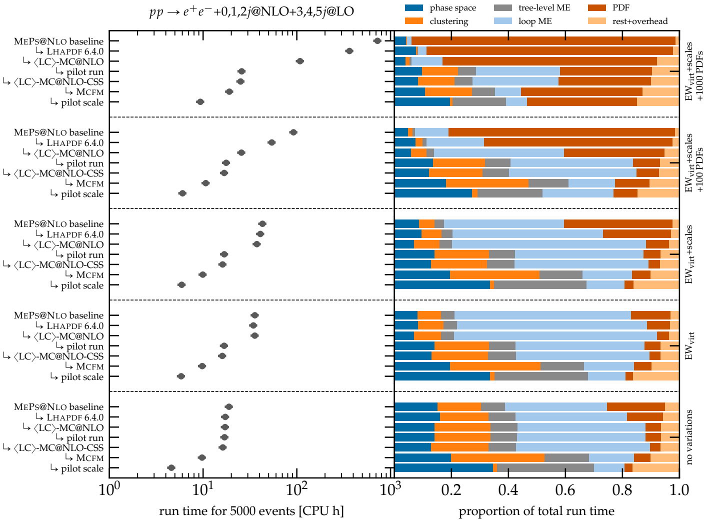
- study the impact of different improvements sequentially:
  - improved interpolation strategies in LHAPDF (6.2.3 $\to$ 6.4.0)
  - replace full-colour spin-correlated S-MC@NLO algortihm with leading-colour spin-averaged $\langle LC \rangle$-MC@NLO (`NLO_CSS_PSMODE` 0 $\to$ 1)
    - this disables subleading colour corrections in the parton shower
  - introduce pilot run in Sherpa (2.2.11 $\to$ 2.2.12)
  - defer leading-colour MC@NLO until after the unweighting (`NLO_CSS_PSMODE` 1 $\to$ 2)
  - use analytic one-loop amplitudes from MCFM in pilot run
  - use a simplified pilot scale for the unweighting

Full suite of improvements

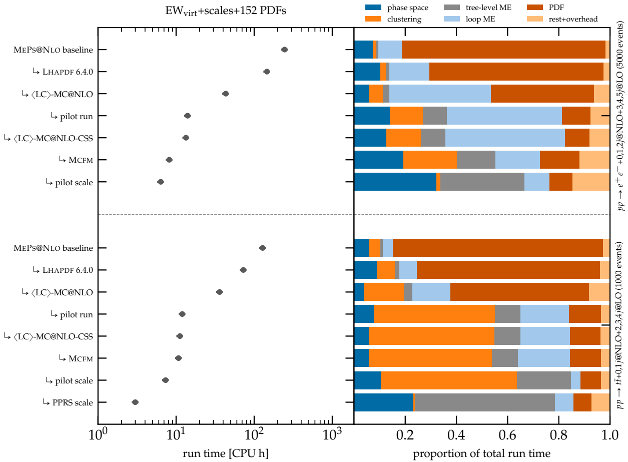
- study the impact of different improvements sequentially:
  - improved interpolation strategies in LHAPDF (6.2.3 $\rightarrow$ 6.4.0)
  - replace full-colour spin-correlated S-MC@NLO algortihm with leading-colour spin-averaged $\langle LC \rangle$-MC@NLO (`NLO_CSS_PSMODE` 0 $\rightarrow$ 1)
    - this disables subleading colour corrections in the parton shower
  - introduce pilot run in Sherpa (2.2.11 $\rightarrow$ 2.2.12)
  - defer leading-colour MC@NLO until after the unweighting (`NLO_CSS_PSMODE` 1 $\rightarrow$ 2)
  - use analytic one-loop amplitudes from MCFM in pilot run
  - use a simplified pilot scale for the unweighting

	$pp \rightarrow e^+e^-$ + jets			$pp \rightarrow t\bar{t}$ + jets		
	runtime [CPU h/5k events]			runtime [CPU h/1k events]		
setup variant	old	new	speed-up	old	new	speed-up
no variations	20 h	5 h	4×	15 h	8 h	2×
EW_{virt}	35 h	5 h	6×	20 h	8 h	2×
EW_{virt} +scales	45 h	5 h	7×	25 h	8 h	4×
EW_{virt} +scales+100 PDFs	90 h	5 h	15×	55 h	8 h	7×
EW_{virt} +scales+1000 PDFs	725 h	8 h	78×	440 h	9 h	51×

Breakdown of CPU budget in V+jets

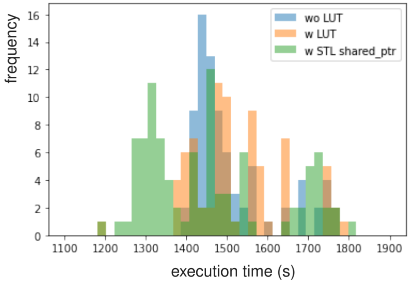


Case study: latest ATLAS baseline configuration



➜ CPU consumption **overall improved by factors of $\times$39 and $\times$43** for V+jets and $t\bar{t}$+jets

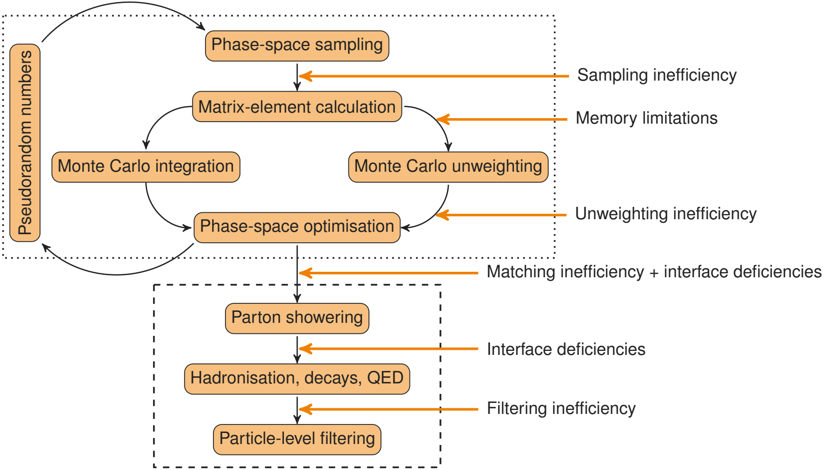
Ongoing COMIX studies



[Ilektra Christidi]

- no more ‘low hanging fruits’ at this point
- focus now shifting to COMIX (used for high-multiplicity LO legs)
- code memory bound, but no localised bottlenecks
  - implementation of a weight cache to avoid having to resolve virtual classes showed promising results at LO but tricky to generalise to NLO
  - not amenable to auto-vectorisation: could vectorise manually, but poor gain/pain ratio (processing time fragmented across too many different parts of the code)
  - memory cache misses checked but not too significant ($\sim$5% L1, $\sim$11% L2, $\sim$4% L3)
- targeted profiling of costliest functions
  - attempt to optimise memory access in `Vegas` interface showed little success
  - substitution of custom shared pointer with STL implementation looking more promising

Looking ahead

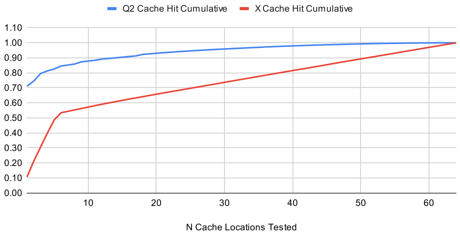


→ Lack of active development on infrastructure tools (LHE, HepMC, ...) set to become a major bottleneck going forward

Summary

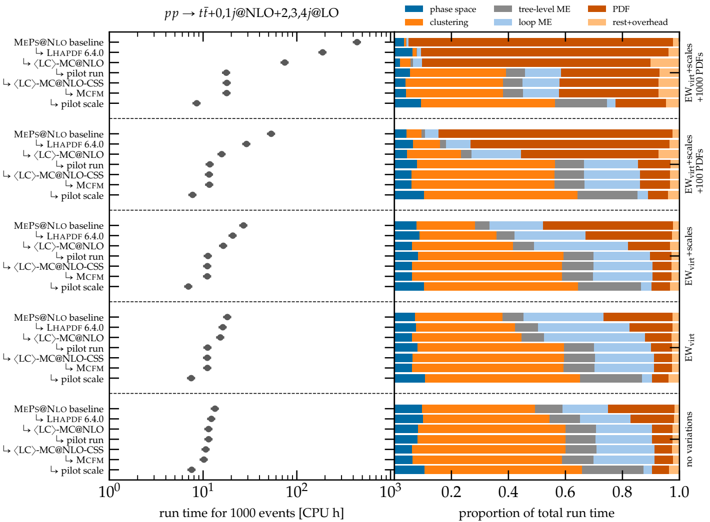
- overall factor 40 speed-up following dedicated profiling of ATLAS multi-leg NLO setups
  - latest LHAPDF release series brings major performance improvements with noticeable impact on overall event-generation run time
  - introduction of pilot run in Sherpa brings a factor 5 improvement
  - using analytic QCD loop amplitudes in the unweighting brings another factor 1.5
- achieves major factor-10 milestone set by HEP Software Foundation
- remaining processing time fragmented with no obvious bottleneck
  - auto-vectorisation doesn't seem to help much
  - switch to modern C++ utilities (e.g. smart pointers) appears more promising

Improving LHAPDF



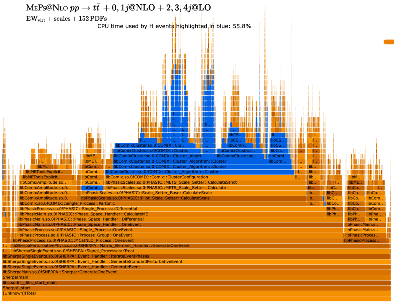
- first PDF-grid cache introduced in v6.3.0
  - rendered ineffective by PDF-call strategy used in Sherpa
  - nevertheless useful as case study
- follow-up release v6.4.0 with improved interpolation logic
  - revised cache implementation with improved memory layout (but well-matched call strategy in the generator still crucial)
  - pre-computation of shared coefficients of the interpolation polynomial along (x, Q^2) grid lines
  - results in factor 3 speed-up for single flavour computations
  - can achieve factor 10 speed-up when combining with multi-flavour caching

Breakdown of CPU budget in $t\bar{t}$+jets

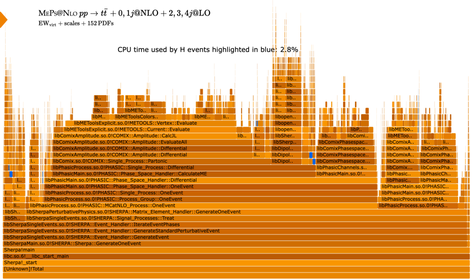


Cluster-independent scale definition

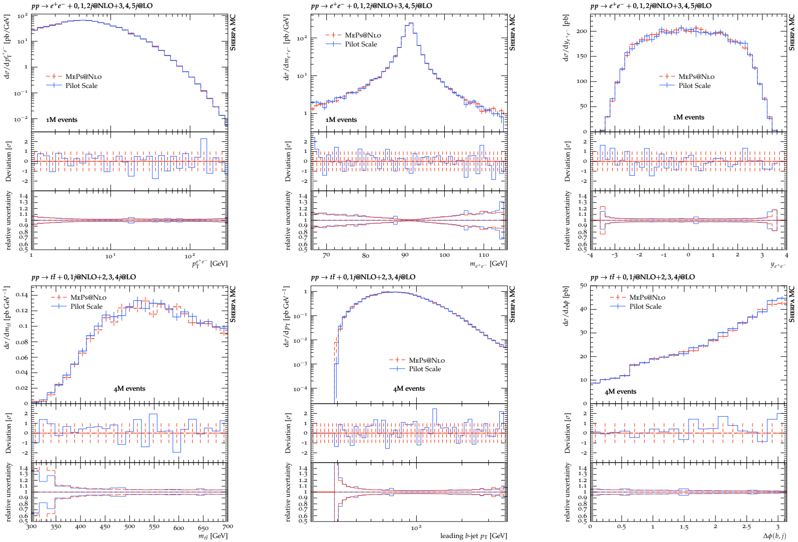
- ➜ employ clustering-independent scale definition ($H_T'/2$) for $\mathcal{H}$-events in $t\bar{t}$+jets (already used in V+jets baseline setup)
- ➜ yields **additional factor 2 speed-up** of the overall run time



➜

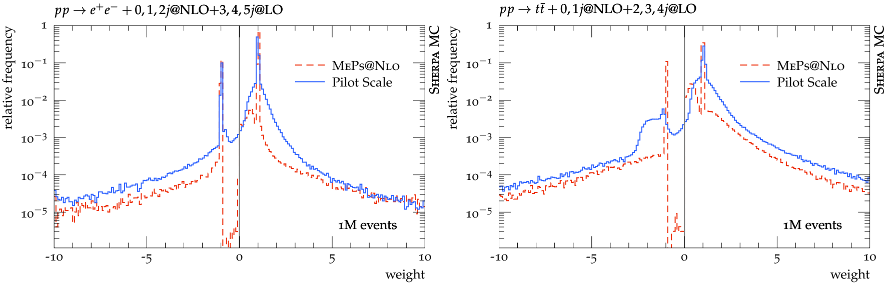


Comparison of MEPS@NLO vs Pilot Scale strategy



Weight distribution for pilot scale

➜ weight distributions for partially unweighted events after matching and merging:



➜ second unweighting would reduce the efficieny by less than factor 2 for large N_{events}