

Sherpa in CMS

**Piergiulio Lenzi, Sebastian Thuer, Philipp Millet
on behalf of CMS
Sherpa developer/user meeting
Dresden
January 7th, 2016**

Usage statistics

- We show comparisons to Sherpa mainly in our Standard Model
 - Few recent examples:
 - Z+jets/ γ +jets: J. High Energy Phys. 10 (2015) 128, Phys. Rev. D 88 (2013) 112009 ,
 - W+Jets, Z+jets: Phys. Lett. B 741 (2015) 12 , Phys. Rev. D 91 (2015) 052008 , Phys. Lett. B 722 (2013) 238-261

Usage statistics

- But this is not translated into a use of Sherpa as a mainstream generator for full simulated samples
 - It means it is not used for background estimation in searches
 - Despite significant push from generator group conveners over the years
- It is however used for a few exotic graviton signal sample (ADD) and for the corresponding background ($\gamma\gamma$ +jets)

Sherpa in CMSSW

- CMSSW release philosophy to ensure reproducibility:
 - We build everything we need as part of the release
 - From gcc onwards
 - The use of pre-compiled code is strongly discouraged, i.e. forbidden in production jobs
 - e.g. we are **unable to use Collier for loop induced processes**, a place where Sherpa is really miles ahead
- Sherpa is built as an external package
 - With dependencies on
 - Hepmc, lhpdf, blackhat, sqlite, fastjet, openloops, mcfm
- We have an interface code that instantiates sherpa and runs the event generation
 - Using our own random number engine

Sherpa sample production

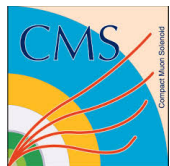
- We make a standalone run to get integration results and libraries compiled
- We pack everything in a tarball (called **sherpack**)
- We ship the sherpack via cvmfs
- Production jobs fetch the sherpack, untar it in the current folder and run sherpa through the interface
- **Without the pain of LHE files**

Then why is it not used?

- Inertia is for sure a player
- Related to inertia, the fact that we often get good results with other generators as well
 - Madgraph quickly closed the gap on MEPS@NLO
 - With a few but noticeable exceptions
 - Loop induced processes
- At some point we struggled keeping up with the Sherpa release pace, but this seems solved now
 - Both because the sherpa release pace has decreased and because we are more flexible building releases
- Some improvement is needed in the feedback loop from the authors
 - especially on technical issues

- Our Sherpa installation does not currently support MPI
- This can be changed, but
 - We struggled to find MPI clusters
 - We had one in Zurich but it is now not available anymore
- It requires a dedicated cluster (and a dedicated person running on that cluster to produce sherpacks for the collaboration)
- Madgraph uses LSF jobs
 - Easy to manage, can run at CERN or on any LSF cluster
 - Can easily run thousands of jobs at CERN and everyone in the collaboration can do it

- All generators in CMSSW are re-initialized every few hundred events
 - what we call a lumi-section, a concept that is also present in data files
- This is needed for replay features of our random number generator
 - i.e. if you want to replay events from a given random number generator state
- We store the random engine state per event
 - This is not enough though
 - If you want to replay event 124 you cannot simply start from the random state at 123 because the initialization process may throw random numbers
- We address this by re-init every few events and storing a random state for the init stage and one for the event
- Sherpa does not like re-init
 - Crash happens if you try to call `Sherpa::InitializeTheRun` more than once
 - Even if you destroy the Sherpa object and allocate a new one
 - I can provide test code to reproduce



Miscellanea III



- NJET, LJET keywords
 - VERY confusing
 - NJET is number of jets, LJET is the number of final state particles