Table of Contents

Fast Simulation User Guide

We welcome all suggestions to improved the FastSim documentation.Feel free to bug the people listed under SWGuideFastSimulation#Contacts.

Contacts

- **Convener**: Sam Bein
- **Hypernews forum**: https://hypernews.cern.ch/HyperNews/CMS/get/famos.html (hn-cms-famos@cernNOSPAMPLEASE.ch)

Links

- FastSim main page: FastSim (CMS only)
- FastSim examples: SWGuideFastSimulationExamples
- FastSim developer's guide: SWGuideFastSimulationDev (CMS only)
- FastSim FAQs: SWGuideFastSimFAQ

Standard recipe for the generation and simulation of ttbar events

See SWGuideFastSimulationExamples

Recommended CMSSW releases

releas cylce	use case	recommended releases
CMSSW_7_6_X	development	
CMSSW_7_5_X	development	-
CMSSW_7_4_X	run2 production	CMSSW_7_4_4 or later
CMSSW_7_3_X	-	-
CMSSW_7_2_X	-	-
CMSSW_7_1_X	-	-
CMSSW_7_0_X	CSA14	≥ CMSSW_7_0_3
CMSSW_5_3_X	recommended for analysis of 2012 data	≥ CMSSW_5_3_6
CMSSW_5_2_X	Summer12 campaign	≥ CMSSW_5_2_6
CMSSW_4_4_X	Fall11 campaign	≥ CMSSW_4_4_3_patch1
CMSSW_4_2_X	Summer11 campaign	≥ CMSSW_4_2_8_patch4

Relevant cmsDriver options

The `cmsDriver.py` command is documented here (outdated) and here (incomplete) and in `cmsDriver.py --help` . The next table lists and explains the options relevant for FastSim.

Example `cmsDriver.py` commands for the fast simulation are available on SWGuideFastSimulationExamples.

`GENFRAGMENT.py` (1st argument)	Define the "genfragment" that configures the event gen-content. For more info see below
`--fast`	tells cmsDriver.py to use FastSim (default is FullSim). This option is in place since `CMSSW_6_2_X`. For the old syntax, see examples.
`-s STEP1,STEP2,...`	**note**: in CMSSW_7_5_0_pre2 the step syntax changes, documentation is provided here: SWGuideFastSimulationSteps75X. Comma separated sequence of steps. Each step itself is a sequence of algorithms for event generation (GEN), detector simulation (SIM), reconstruction (RECO), ... The available steps and their definitions are listed here (outdated) and here (incomplete). Most purposes are covered by one of the following combinations of steps. **Commonly used combinations of steps** : • `-s GEN,SIM`, run generation and detector simulation • `-s GEN,SIM,RECO`, the above + event reconstruction • `-s GEN,SIM,RECO,HLT:GRun`, the above + HLT • `-s GEN,SIM,RECO,HLT:GRun,VALIDATION`, the above + production of validation plots (DQM) • `-s GEN,SIM,HLT:GRun`, run generation, detector simulation and HLT • `-s HARVESTING:validationHarvestingFS`, "Harvest" validation plots, see examples for usage. Harvesting turns the complicated DQMio files into plain root files) Note: for the old syntax, < `CMSSW_6_2_X`, see examples. **Selecting the HLT menu** The HLT menu is selected in the steps with the syntax HLT:<MENU>. Some examples for CMSSW_7_X: • `-s GEN,SIM,RECO,HLT:GRun --conditions=auto:startup_GRun` to use the development HTL menu • `-s GEN,FASTSIM,HLT:2013 --conditions=auto:startup_2013` to use the frozen HLT menu "2013" To find out which HLT menus are available for FastSim in your favourite release: each available HLT menu `XXX` has a corresponding file `HLTrigger/Configuration/python/HLT_X_Famos_cff.py`. Note that, in the examples above, there is a correspondence between the HLT menu name and the last part of the `--conditions` option.This correspondance is not obligatory, but usually it is recommended. The HLT depends on the conditions for e.g. subdetector calibrations, channel mappings, geometry, and, at higher-level for e.g. jet energy corrections etc. **WARNING related to VALIDATION step** It is currently not possible to run the VALIDATION step in a standalone manner. Please run validation together with GEN,SIM and RECO steps.
`--conditions=CONDITIONS`	Define the conditions, such as calibration, alignment and noise levels. See SWGuideFrontierConditions for available conditions.(Missing: most common examples. syntax: what's before the colon what's after it what is the

	meaning of "startup" and why is the trigger menu specified there)
`--pileup=PUSCENARIO`	define pileup scenario. All pileup scenarios available for FullSim are automatically available for FastSim.Pileup scenarios are defined in `Configuration/StandardSequences/python/Mixing.py`. For each available pileup option `X` there is a line `addMixingScenario("X","Y")`, where `Y` points to the configuration file or provides parameters that define the pileup scenario.==addMixingScenario("FS_X","Y")==, where `Y` points to the configuration file or provides parameters that define the pileup scenario.See PdmVPileUpDescription to find out which PU scenario is used in which production campaign.
`--eventcontent=EVENTCONTENT`	defines which information of the events will be written to the output file. see header of `Configuration/EventContent/python/EventContent_cff.py` for available options and their definition.
`--datatier=DATATIER`	A flag to specify metadata in your output file. Not much to care about.
`-n`	number of events to be produced/processed
`--no_exec`	only produce the CMSSW configuration file, do not run it.
`--filein=FILE`	read input files from FILE **examples** • `--filein=file:YourLHEFile.lhe --filetype=LHE` when using lhe files as input • `--filein=mcdb:ID` for LHE files in mcdb, where ID corresponds to the article ID • `--filein=file:YourEDMFile.root` read gen information from existing EDM files (**see bug warning**) Note that when using lhe files as input, your genfragment must be configured to perform showering and hadronization, depending on how your lhe files were generated.See option `GENFRAGMENT.py` for more info about genfragments. **bug warning:** reading from EDM files results in double vertex smearing, leading to a significant reduction of track reconstruction efficiencies. You can fix this as explained here TODO: how to read files from das
`--filetype=FILETYPE`	use this option when using an input file that is not in EDM format, see option `--filein` for use cases
`--geometry=GEOMETRY`	**NOTE: This option has no effect on FastSim**
`--customise=PACKAGE.FUNCTION`	Apply function FUNCTION from package PACKAGE at the end of the simulation configuration. E.g. `--customise=SLHCUpgradeSimulations.Configuration.postLS1Customs` configures FastSim for run2 (after 2014) by applying the function `postLS1Customs(process)` in the directory `SLHCUpgradeSimulations/Configuration/python`
`--magField=MAGFIELD`	Use magnetic field description MAGFIELD. The default description is the run1 description (before 2013). To load the run2 description (after 2014) use `--magField=38T_PostLS1`

-- LukasVanelderen - 27 Mar 2014

This topic: CMSPublic > SWGuideFastSimulation
Topic revision: r40 - 2020-11-12 - SezenSekmen