CMSDriver

1. “cmsDriver is a tool to create production-solid configuration files from minimal command line options.”

2. That isn’t to say it is obvious…

3. https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideCmsDriver

CMSDriver (example command)

cmsDriver.py TTbar_13TeV_TuneCUETP8M1_cfi --conditions auto:run2_mc -n 10 --eventcontent RAWSIM -s GEN,SIM --datatier GEN-SIM-RAW --fileout file:step1.root --no_exec

1. **Generator fragment**
2. **alignment and calibration conditions**
3. **number of events to produce**
4. **eventcontent to be saved**
5. **steps to be run in the sequence**
6. **metadata description of steps run for ProdAgent**
7. **fileout name**
8. **do not execute, just generate config file (run with cmsRun)**

CMSDriver (steps)

1. Available options (not exhaustive list): LHE,GEN,SIM,DIGI,L1,DIGI2RAW,HLT,RAW2DIGI,RECO,POSTRECO,DQM,ALCA,EI,PAT,VALIDATION,HARVESTING, NONE or ALL

2. Common to break up into 3 steps:

    1. GEN,SIM

    2. DIGI,L1,DIGI2RAW,HLT

    3. RAW2DIGI,L1Reco,RECO,PAT…

CMSDriver (example EventContent)

1. LHE: include pure LHE production
2. RAW , RECO, AOD: include reconstruction content
3. RAWSIM, RECOSIM, AODSIM: include reconstruction and simulation
4. MINIAOD, MINIAODSIM, RAWMINIAOD, RAWMINIAODSIM
5. GENRAW: slimmed-down version of RAWSIM for small transient disk size during MC production, contains Gen+Rawdata
6. PREMIX: extension of GENRAW with special Digi collection(s) for pre-mixing minbias events for pileup simulation. Raw2Digi step is done on this file.
7. PREMIXRAW extension of RAWSIM for output of second stage of PreMixing using the DataMixer.
8. RAWDEBUG(RAWSIM+ALL_SIM_INFO), RAWDEBUGHLT(RAWDEBUG+HLTDEBUG)
9. RAWSIMHLT (RAWSIM + HLTDEBUG)
10. RAWRECOSIMHLT, RAWRECODEBUGHLT
11. FEVT (RAW+RECO), FEVTSIM (RAWSIM+RECOSIM), FEVTDEBUG (FEVTSIM+ALL_SIM_INFO), FEVTDEBUGHLT (FEVTDEBUG+HLTDEBUG)

CMSDriver (Generator)

1. Can use other generator inputs as needed

2. LHE step available to generate LHE within CMSSW

3. Or, What if you need to use an LHE file?

    1. add in “--filein=file:ttbar_1.lhe"

    2. replace the gen-fragment with a hadronizer

cmsDriver.py Configuration/GenProduction/python/ Hadronizer_TuneCUETP8M1_13TeV_generic_LHE_pythia8_cff.py --filein=file:ttbar_1.lhe --conditions auto:run2_mc -n 10 --eventcontent RAWSIM -s GEN,SIM --datatier GEN-SIM-RAW-HLT --fileout file:step1.root --no_exec

CMSDriver (Where to find conditions and eras)

1. Era:
    1. Configuration/Eras/python
    2. https://github.com/cms-sw/cmssw/blob/master/Configuration/StandardSequences/python/Eras.py
    3. https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideCmsDriverEras
2. Conditions:
    1. Configuration/AlCa/python/autoCond.py
    2. https://github.com/cms-sw/cmssw/blob/master/Configuration/AlCa/python/autoCond.py

CMSDriver (where to find examples)

1. Production recipes: https://twiki.cern.ch/twiki/bin/view/CMS/PdmVMcCampaigns

2. Standard recipes: https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideFastSimulationExamples#Standard_recipes

3. https://cmssdt.cern.ch/SDT/html/showIB.html

    1. https://cms-sw.github.io/relvalLogDetail.html#slc6_amd64_gcc530;CMSSW_9_1_X_2017-04-24-1100

4. runTheMatrix.py:

    1. Great tool to get cmsdriver commands used for validation with up to date conditions.

    2. However, you should remove the validation steps, harvesting and ALCA steps when adapting commands if you aren't doing validation stuff.

        1. Details on twiki for both full and fastsim: https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideFastSimulationExamples#Standard_recipes

    3. runTheMatrix.py -n lists workflows

    4. e.g. runTheMatrix.py -l 1325 --dryRun will give you the cmsDriver commands for a relval TTbar workflow

CMSDriver (runTheMatrix example)

Output to: runTheMatrix.py -l 1325 --dryrun

1. cmsDriver.py TTbar_13TeV_TuneCUETP8M1_cfi --conditions auto:run2_mc -n 10 --era Run2_2016 --eventcontent FEVTDEBUG ~~--relval 9000,50~~ -s GEN,SIM --datatier GEN-SIM --beamspot Realistic50ns13TeVCollision --fileout file:step1.root

2. cmsDriver.py step2 --conditions auto:run2_mc -s DIGI~~:pdigi_valid~~,L1,DIGI2RAW,HLT:@relval2016 --datatier GEN-SIM-DIGI-RAW-HLTDEBUG -n 10 --era Run2_2016 --eventcontent FEVTDEBUGHLT --filein file:step1.root --fileout file:step2.root

3. cmsDriver.py step3 --runUnscheduled --conditions auto:run2_mc -s RAW2DIGI,L1Reco,RECO,~~EI~~,PAT~~,VALIDATION:@standardValidation+@miniAODValidation,DQM:@standardDQM+@miniAODDQM~~ --datatier GEN-SIM-RECO,MINIAODSIM,~~DQMIO~~ -n 10 --era Run2_2016 --eventcontent RECOSIM,MINIAODSIM,~~DQM~~ --filein file:step2.root --fileout file:step3.root

4. ~~cmsDriver.py step4 --filetype DQM --conditions auto:run2_mc --mc -s HARVESTING:@standardValidation+@standardDQM+@miniAODValidation+@miniAODDQM --era Run2_2016 -n 100 --filein file:step3.root --fileout file:step4.root~~

5. ~~cmsDriver.py step5 --conditions auto:run2_mc --filein file:step3.root -s ALCA:TkAlMuonIsolated+TkAlMinBias+MuAlOverlaps+EcalESAlign+EcalTrg --datatier ALCARECO -n 1000 --era Run2_2016 --eventcontent ALCARECO --filein file:step3.root --fileout file:step5.root~~

CMSDriver (runTheMatrix example)

stripped down output from: runTheMatrix.py -l 1325 --dryRun

1. cmsDriver.py TTbar_13TeV_TuneCUETP8M1_cfi --conditions auto:run2_mc -n 10 --era Run2_2016 --eventcontent FEVTDEBUG -s GEN,SIM --datatier GEN-SIM --beamspot Realistic50ns13TeVCollision --filein file:step1.root

2. cmsDriver.py step2 --conditions auto:run2_mc -s DIGI,L1,DIGI2RAW,HLT:@relval2016 --datatier GEN-SIM-DIGI-RAW-HLTDEBUG -n 10 --era Run2_2016 --eventcontent FEVTDEBUGHLT --filein file:step1.root --fileout file:step2.root

3. cmsDriver.py step3 --runUnscheduled --conditions auto:run2_mc -s RAW2DIGI,L1Reco,RECO,PAT --datatier GEN-SIM-RECO,MINIAODSIM -n 10 --era Run2_2016 --eventcontent RECOSIM,MINIAODSIM --filein file:step2.root --fileout file:step3.root

Consider using a smaller eventcontent

CMSDriver (help)

1. For more flags and help:

    1. cmsDriver.py --help

    2. runTheMatrix.py --help

CMSDriver (FastSim)

What if you want to use FastSim?

1. Use --fast flag but need to change the command as well

2. Lots of up-to-date information and workflows here: https://twiki.cern.ch/twiki/bin/view/CMSPublic/SWGuideFastSimulationExamples

CMSDriver (checking and debugging)

1. Output of cmsDriver.py outputs a python config file, for example named “config.py”. This means you can:
    1. Check for syntax errors, etc.:
       python config.py
    2. Interact with the module, process, sequence objects created in the file:
       python -i config.py
    3. Dump the entire config to see all the settings:
       edmConfigDump config.py

CMSDriver (threading)

1. cmsDriver -t lets you specify threading.
    1. do not abuse this on interactive nodes!