Physics Performance & Datasets

Louis Moureaux Patrick L.S. Connor

Universität Hamburg

CMS PO&DAS at DESY — October 2023

SPONSORED BY THE Federal Ministry of Education and Research

CLUSTER OF EXCELLENCE QUANTUM UNIVERSE

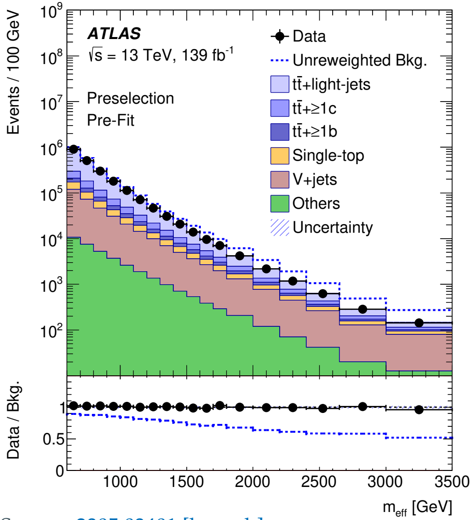
Overview of PPD

PPD — Physics Performance & Datasets

- **PdmV**
  - Analysis Recipes
  - Data Processing: Run 2, Run 3
  - MC Production: Index Page
  - Validation of CMSSW releases
- **AlCaDB**
  - Alignment, Calibration, Database
  - Global Tags
- **DQM**
  - Data Quality & Monitoring
  - Shifts, Golden JSON
- **PF**
  - Particle Flow
  - Physics Objects

In close collaboration with other groups in CMS!

ATLAS Analysis



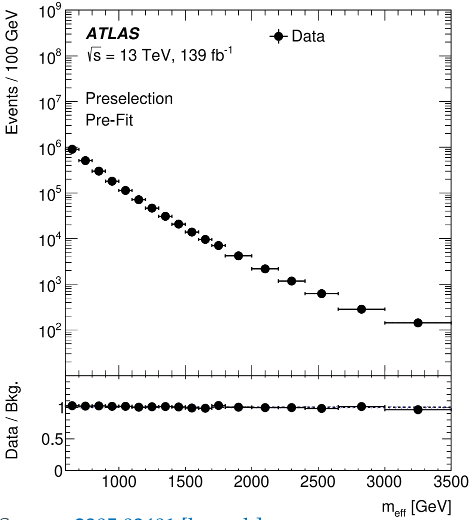
Source: 2305.03401 [hep-ph]

What do we need to reproduce this analysis in CMS?

Search for vector-like T quarks decaying into Ht or Zt in pp collisions at $\sqrt{s} = 13\,\text{TeV}$ *with the ATLAS detector*

Basic selection:

- One electron or muon
- At least three jets, one of which b-tagged
- MET $> 20\,\text{GeV}$



Source: 2305.03401 [hep-ph]

First Part

Data

Datasets

Events are stored within “datasets”, with a structured naming convention:

/MET/Run2016F-21Feb2020_UL2016_HIPM-v1/AOD

- MET → trigger family
- Run2016F → data-taking period
- 21Feb2020_UL2016_HIPM-v1 → processing
- AOD → file format

- Trigger categories: SingleMuon, MET, JetHT, ...
- Data-taking periods: every year split in A, B, C, ...
- Data processing encoded in a “global tag” (here 21Feb2020_UL2016_HIPM) and version number
- Many event storage formats exist: RAW/AOD/MINIAOD/NANOAOD/...

Finding Data

To find your data, follow the “ReReco UL” (=Run 2) or “Run3 DATA” links from:

https://pdmv-pages.web.cern.ch/

2022	/EGamma/Run2022C-v1/RAW 202 runs | 263872531 events	/EGamma/Run2022C-PromptReco-v1/AOD 202 runs PromptReco | AOD	263857026 0.99994124 -0.00005876	/EGamma/Run2022C-PromptReco-v1/MINIAOD 202 runs PromptReco | MiniAODv3	263857026 1.00000000	/EGamma/Run2022C-PromptNanoAODv10_v1-v1/NANOAOD 202 runs PromptNanoAODv10 | NanoAODv10
				/EGamma/Run2022C-10Dec2022-v1/MINIAOD 179 runs 10Dec2022 | MiniAODv3	263336263 0.99802634 -0.00197366	/EGamma/Run2022C-PromptNanoAODv10_v1-v1/NANOAOD 202 runs PromptNanoAODv10 | NanoAODv10
		/EGamma/Run2022C-10Dec2022-v1/AOD 179 runs 10Dec2022 | AOD	263470928 0.99847804 -0.00152196	/EGamma/Run2022C-PromptReco-v1/MINIAOD 202 runs PromptReco | MiniAODv3	263857026 1.00146543 +0.00146543	/EGamma/Run2022C-PromptNanoAODv10_v1-v1/NANOAOD 202 runs PromptNanoAODv10 | NanoAODv10
				/EGamma/Run2022C-10Dec2022-v1/MINIAOD 179 runs 10Dec2022 | MiniAODv3	263336263 0.99948888 -0.00051112	/EGamma/Run2022C-PromptNanoAODv10_v1-v1/NANOAOD 202 runs PromptNanoAODv10 | NanoAODv10
2022	/EGamma/Run2022D-v1/RAW 73 runs | 89293439 events	/EGamma/Run2022D-PromptReco-v2/AOD 48 runs PromptReco | AOD	56696615 0.63494715 -0.36505285	/EGamma/Run2022D-PromptReco-v2/MINIAOD 48 runs PromptReco | MiniAODv3	56696615 1.00000000	/EGamma/Run2022D-PromptNanoAODv10_v2-v1/NANOAOD 48 runs PromptNanoAODv10 | NanoAODv10
		/EGamma/Run2022D-PromptReco-v1/AOD 24 runs PromptReco | AOD	32596823 0.36505283 -0.63494717	/EGamma/Run2022D-PromptReco-v1/MINIAOD 24 runs PromptReco | MiniAODv3	32596823 1.00000000	/EGamma/Run2022D-PromptNanoAODv10_v1-v1/NANOAOD 24 runs PromptNanoAODv10 | NanoAODv10
				/EGamma/Run2022D-10Dec2022-v1/MINIAOD 69 runs 10Dec2022 | MiniAODv3	88530905 2.71593661 +1.71593661	/EGamma/Run2022D-PromptNanoAODv10_v1-v1/NANOAOD 24 runs PromptNanoAODv10 | NanoAODv10
		/EGamma/Run2022D-PromptReco-v3/AOD 1 runs PromptReco | AOD	1 0.00000001 -0.99999999	/EGamma/Run2022D-PromptReco-v3/MINIAOD 1 runs PromptReco | MiniAODv3	1 1.00000000	
2022	/JetMET/Run2022C-v1/RAW 138 runs | 170101289 events			/JetMET/Run2022C-PromptReco-v1/MINIAOD	170094593	/JetMET/Run2022C-PromptNanoAODv10-v1/NANOAOD 138 runs PromptNanoAODv10 | NanoAODv10
						-v1/NANOAOD Dv10
						1/NANOAOD Dv10
						/JetMET/Run2022C-PromptNanoAODv10_v1-v1/NANOAOD 138 runs PromptNanoAODv10 | NanoAODv10

Production settings can be found on the ReReco website.

Luminosity

Luminosity encodes how much data there is and is used to normalize predictions:

$$N_{\text{events}} = \mathcal{L} \cdot \sigma$$

luminosity → $\mathcal{L}$; cross section → σ

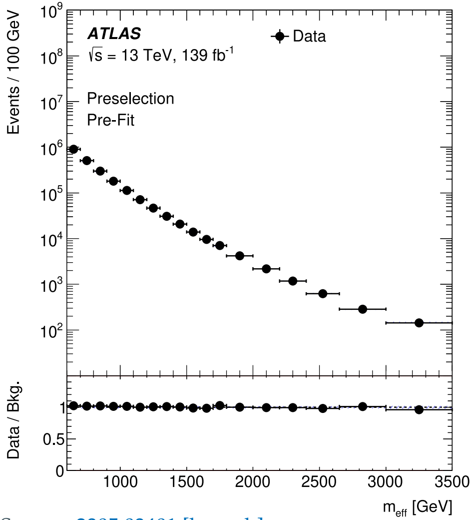
The Lumi POG measures it for everyone in CMS.

Results (and uncertainties) can be found in the TWiki (see lumi POG exercise).

How can one measure this?

Read this nice paper: LUM-17-003

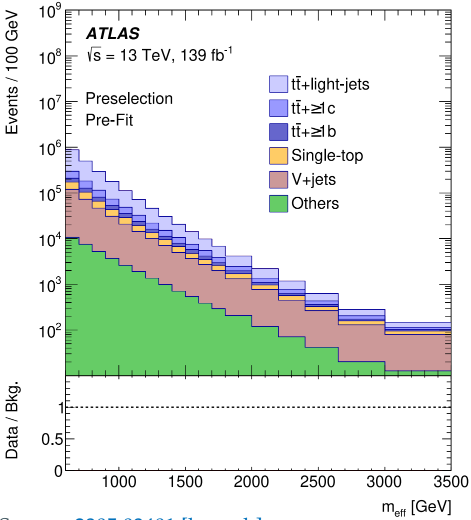
Exercise



Source: 2305.03401 [hep-ph]

Find your data samples!

- Using Run 2 "UltraLegacy" datasets
- Choose appropriate triggers for the analysis selection
- Focus on 2018 data
- Pick the latest NanoAOD datasets (v9)
- Check the luminosity for these data



Source: 2305.03401 [hep-ph]

Second Part

Simulation

Datasets

Like data, the name of datasets with simulated events follow a structured scheme:

```
/QCD_Pt-10to15_TuneCP5_Flat2018_13TeV_pythia8/RunIISummer20UL18MiniAOD
```

Datasets

Like data, the name of datasets with simulated events follow a structured scheme:

```
/QCD_Pt-10to15_TuneCP5_Flat2018_13TeV_pythia8/RunIISummer20UL18MiniAO
```

cuts

generation information

physics process

- Base physics process for the dataset (naming convention)
- Event preselection: decay modes, kinematic cuts
- Information about tools and settings used in the production

Datasets

Like data, the name of datasets with simulated events follow a structured scheme:

`pythia8/RunIISummer20UL18MiniAODv2-106X_upgrade2018_realistic_v16_L1v1-v2/MINIAO`

production campaign — global tag — version

- Production campaign: generation of Monte Carlo production
- Global tag for the simulated data-taking conditions
- Version number used in some cases

Datasets

Like data, the name of datasets with simulated events follow a structured scheme:

`upgrade2018_realistic_v16_L1v1-v2/MINIAODSIM`

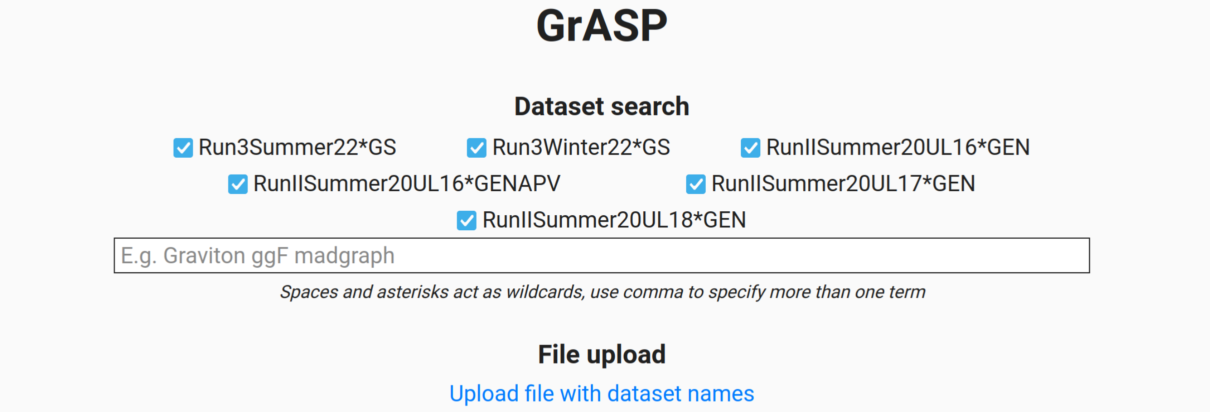
format

- Same event storage formats as data, but with the generated particles also saved: `AODSIM/MINIAOD/NANOAOD/`...

Finding Simulation

The main index for simulation is at:

https://cms-pdmv-prod.web.cern.ch/grasp/



Finding Simulation

The main index for simulation is at:

https://cms-pdmv-prod.web.cern.ch/grasp/

GrASP

Run 3 — **Dataset search** — **Run 2 UL**

- [x] Run3Summer22*GS
- [x] Run3Winter22*GS
- [x] RunIISummer20UL16*GEN
- [x] RunIISummer20UL16*GENAPV
- [x] RunIISummer20UL17*GEN
- [x] RunIISummer20UL18*GEN

E.g. Graviton ggF madgraph

Spaces and asterisks act as wildcards, use comma to specify more than one term

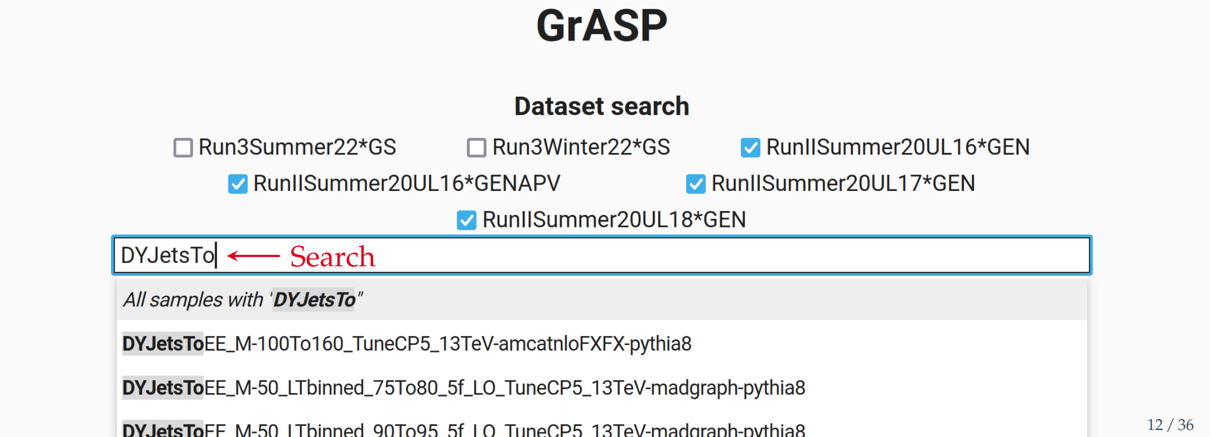
File upload

Upload file with dataset names

Finding Simulation

The main index for simulation is at:

`https://cms-pdmv-prod.web.cern.ch/grasp/`



Finding Simulation

The main index for simulation is at:

https://cms-pdmv-prod.web.cern.ch/grasp/

Samples in

mmer22*GS,Run3Winter22*GS,RunIISummer20UL16*GEN,RunIISummer20UL16*GENAPV,RunIISumr

Dataset **DYJetsToEE_M-50_massWgtFix_TuneCP5_13TeV-powhegMiNNLO-pythia8-photos**

Events: **All** 5M+ 10M+ 20M+ 50M+ | MiniAOD: **All** v1 v2 | NanoAOD: **All** v2 v9 | Download table: CSV XLS

Add or remove [ ] interested PWG or [ ] tag

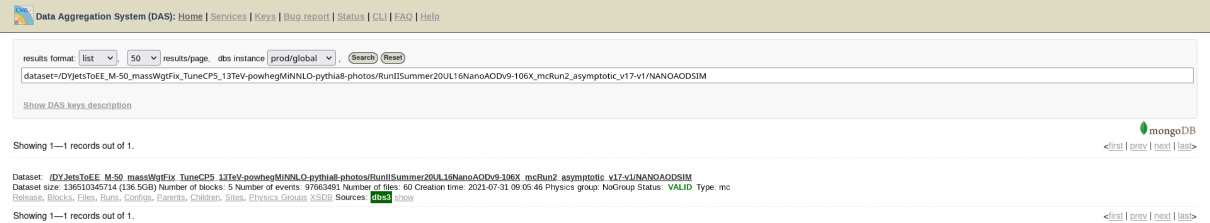
Pages: **1**

Short Name	Dataset Name	Root Request			MiniAOD Request	NanoAOD Request	Chained Request	☐
Type to search...	Type to search...	Tags	Int. PWGs	Type to search...	Type to search...	Type to search...	Type to search...	
DYJetsToEE NLO PH+P8	DYJetsToEE_M-50_massWgtFix_TuneCP5_13TeV-powhegMiNNLO-pythia8-photos	ZplusJets	SMP	McM pMp RunIISummer20UL16wmLHEGEN Events: 100M	McM pMp DAS v1 Events: 99.8M/100M	McM pMp DAS v2 Events: 99.71M/99.8M	Premix	☐
					McM pMp DAS v2 Events: 99.18M/99.79M	McM pMp DAS v9 Events: 97.66M/99.18M	Premix	☐
		ZplusJets	SMP	McM pMp RunIISummer20UL16wmLHEGENAPV Events: 120M	McM pMp DAS v1 Events: 119.76M/120M	McM pMp DAS v2 Events: 119.76M/119.76M	PremixAPV	☐
					McM pMp DAS v2 Events: 118.7M/119.7M		PremixAPV	☐
						McM pMp DAS v9 Events: 118.7M/118.7M	PremixAPV	☐
		ZplusJets	SMP	McM pMp RunIISummer20UL17wmLHEGEN	McM pMp DAS v1 Events: 258.89M/260M	McM pMp DAS v2 Events: 258.89M/258.89M	Premix	☐
					McM pMp DAS v2		Premix	☐

DAS

Clicking the DAS links in Grasp will bring you to:

`https://cmsweb.cern.ch/das`



There you can find (also for data!):

- The full dataset name
- The list of files

DAS also has a command line (`dasgoclient`) and a search language for dataset-related queries.

Cross Sections

We also need to know the cross section for the physics process:

$$N_{\text{events}} = \mathcal{L} \cdot \sigma$$

luminosity ↑ ↑ cross section

This information can be found in different ways:

- For common **standard model processes**, accurate values are available on the TWiki for 13 TeV and 13.6 TeV (preliminary)
- **Higgs** cross sections are computed separately
- Most processes also have an entry in XSDB
- Custom calculations are performed for some publications

Finding Details

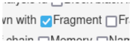
The dataset name contains basic information about the generators used:

`DYJetsToEE_M-50_massWgtFix_TuneCP5_13TeV-powhegMiNNLO-pythia8-photos`

> You should cite every generator tool used in your analysis.
> See CitationsForGenerators on the TWiki for guidance.

Sometimes you need more info (as in the second part of the exercise):

1. From Grasp, click on the **McM** link in the **Root Request** column
2. Click on **Select View** 

and tick the box left of **Fragment** 

3. A new column called **Fragment** appears on the right. 

Click on any of the two icons there.
4. You will see the CMSSW configuration used in production

Typical Example

```
import FWCore.ParameterSet.Config as cms

from Configuration.Generator.Pythia8CommonSettings_cfi import *
from Configuration.Generator.MCTunes2017.PythiaCP5Settings_cfi import *
from Configuration.Generator.Pythia8PowhegEmissionVetoSettings_cfi import *
from Configuration.Generator.PSweightsPythia.PythiaPSweightsSettings_cfi import *

externalLHEProducer = cms.EDProducer("ExternalLHEProducer",
    args = cms.vstring('/cvmfs/cms.cern.ch/phys_generator/gridpacks/slc6_amd64_gcc700/13TeV/powheg/Vj_NNLOPS/Zj_slc6_am
    nEvents = cms.untracked.uint32(5000),
    numberOfParameters = cms.uint32(1),
    outputFile = cms.string('cmsgrid_final.lhe'),
    scriptName = cms.FileInPath('GeneratorInterface/LHEInterface/data/run_generic_tarball_cvmfs.sh'),
    generateConcurrently = cms.untracked.bool(True),
)

generator = cms.EDFilter("Pythia8HadronizerFilter",
    maxEventsToPrint = cms.untracked.int32(1),
    pythiaPylistVerbosity = cms.untracked.int32(1),
    filterEfficiency = cms.untracked.double(1.0),
    pythiaHepMCVerbosity = cms.untracked.bool(False),
    comEnergy = cms.double(13000.),
    PythiaParameters = cms.PSet(
        pythia8CommonSettingsBlock,
        pythia8CP5SettingsBlock,
        pythia8PSweightsSettingsBlock,
        pythia8PowhegEmissionVetoSettingsBlock,
        processParameters = cms.vstring(
            'SpaceShower:pTmaxMatch = 1',
            'TimeShower:pTmaxMatch = 1',
            'ParticleDecays:allowPhotonRadiation = on',
            'TimeShower:QEDshowerByL = off',
            'BeamRemnants:hardKTOnlyLHE = on'
```

Generator Tools

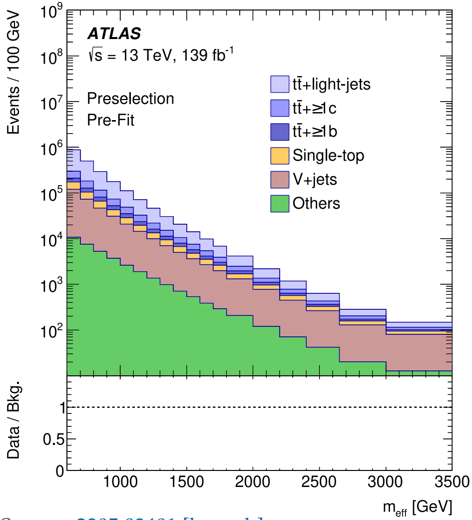
Typical steps for event generation:

- **PDF sampling:** get partons from colliding protons
  PDF sets: NNPDF 3.1, ...
- **Hard process:** generate particles following Feynman diagrams
  Generators: MadGraph5_aMC@NLO, the POWHEG BOX, Pythia, Herwig, Sherpa, MCFM, ...
  Accuracy: LO, NLO, NNLO, ...
- **Parton shower:** improve realism with extra radiation and hadronization
  Tools: Pythia, Herwig, Tauola, Photos, ...
  Merging & matching: MLM, FxFx, ...
  Tunes: CP5, CH3, ...

Want to know more?

Start from the Generators group TWiki and the great generator tutorials

Exercise



Source: 2305.03401 [hep-ph]

Find your signal samples

- Using Run 2 “UltraLegacy” datasets
- Focus on 2018 campaigns
- Pick the latest NanoAOD datasets (v9) for background processes
- Understand their properties: cross section to use, generators used and their settings, citations to add to the paper, ...
- Do we have samples for the signal?

Tips

There are often multiple fitting choices

Prefer “Premix” samples with more stat, higher accuracy, or leptonic decays

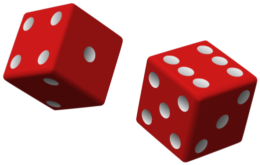
Search help:

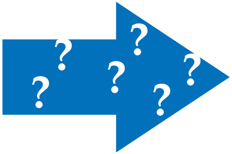
- $t\bar{t}$ = `TTTo*`
- Single-top = `ST_s-channel*`, `ST_t-channel_(anti)top*`
- $V + \text{jets}$ = `DYJets*`, `WJets*`
- Others = `TTZ*`, `TTW*`, `TTH*`, `TTTT*`, `ZZ*`, `WZ*`, `WW*`, ...

Third Part

Event Generation

Monte Carlo Production





Choose Collection:

Jets

Edit table:

Idx	Filtered	pt	eta	phi
0	1	116.2	-0.208	2.337
1	1	72.5	0.722	-0.816
2	1	58.6	-0.293	-0.504
3	1	22.2	0.930	-0.296
4	1	20.8	3.566	2.525
5	1	20.5	3.390	-1.231
6	0	19.0	-1.850	1.585
7	0	18.7	-4.444	0.876

How does CMS produce Monte Carlo events?

What we need

Things we need for accurately simulated datasets:

- Physics process (incl. showering and hadronization)

- Pileup
- The detector itself (many subsystems!)
- Detector readout (digitization)
- Triggers (L1 and HLT)
- Event reconstruction (with alignment)
- Calibrations (global tag)
- At least 3 file formats (AOD, MiniAOD, NanoAOD)

What we need

Things we need for accurately simulated datasets:

- Physics process (incl. showering and hadronization)
  $\rightarrow$ Pythia/MadGraph/Powheg...
- Pileup $\rightarrow$ Event mixing: classical or premix
- The detector itself (many subsystems!) $\rightarrow$ GEANT4 simulation
- Detector readout (digitization) $\rightarrow$ CMSSW code
- Triggers (L1 and HLT)
- Event reconstruction (with alignment)
- Calibrations (global tag)
- At least 3 file formats (AOD, MiniAOD, NanoAOD)

Organization

Main steps for a typical Monte Carlo production:

1. **GEN:** Run the MC generator
2. **SIM:** Simulate the detector
3. **DIGI:** Add pileup, merge hits, simulate detector output
4. **HLT:** Simulate the L1 and run the HLT
5. **RECO:** Run reconstruction and prepare AOD
6. **MiniAOD:** Prepare MiniAOD, reasonably fast
7. **NanoAOD:** Prepare NanoAOD, runs very fast

Unified tool to run all this: **cmsDriver.py**

Generator Fragments: What to produce

The physics process is described in a “Generator Fragment”: a short CMSSW config snippet that sets up the generator tool.

Example:

```
import FWCore.ParameterSet.Config as cms
from Configuration.Generator.Pythia8CommonSettings_cfi import *
from Configuration.Generator.MCTunesRun3ECM13p6TeV.PythiaCP5Settings_cfi import *

generator = cms.EDFilter("Pythia8ConcurrentGeneratorFilter",
                         comEnergy = cms.double(13600.0),     #  <--- Energy
                         filterEfficiency = cms.untracked.double(1), maxEventsToPrint = cms.untracked.int32(1),
                         pythiaHepMCVerbosity = cms.untracked.bool(False),
                         pythiaPylistVerbosity = cms.untracked.int32(1),
                         PythiaParameters = cms.PSet(
        pythia8CommonSettingsBlock,
        pythia8CP5SettingsBlock,                        #  <--- Pythia tune (here CP5)
        processParameters = cms.vstring(
            'WeakSingleBoson:ffbar2gmZ = on',           #  <--- Pythia config: generate q + qbar -> Z
            '23:onMode = off',                          #  <--- Disable all Z decays
            '23:onIfAny = 11',                          #  <--- Enable Z -> e e
            '23:onIfAny = 13',                          #  <--- Enable Z -> mu mu
            '23:onIfAny = 15',                          #  <--- Enable Z -> tau tau
            '23:mMin = 50.'),                           #  <--- Minimum mass of 50GeV
        parameterSets = cms.vstring('pythia8CommonSettings', 'pythia8CP5Settings', 'processParameters')))

ProductionFilterSequence = cms.Sequence(generator)  #  <--- Define the CMSSW Sequence to run
```

cmsDriver: How to produce

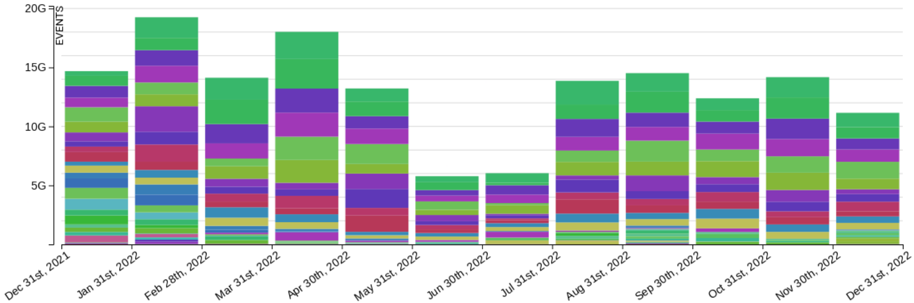
`cmsDriver.py` takes the config fragment and produces a complete CMSSW configuration that you can run with `cmsRun`.

Example:

```
cmsDriver.py Configuration/GenProduction/python/TSG-Run3Summer22GS-00003-fragment.py
    -n 100
    --no_exec
    --python_filename theCMSSWConfigurationFile_cfg.py
    --fileout file:TSG-Run3Summer22GS-00003.root
    --mc
    --step GEN,SIM
    --datatier GEN-SIM
    --eventcontent RAWSIM
    --era Run3
    --beamspot Realistic25ns13p6TeVEarly2022Collision
    --conditions 124X_mcRun3_2022_realistic_v12
    --geometry DB:Extended
```

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

Central Production



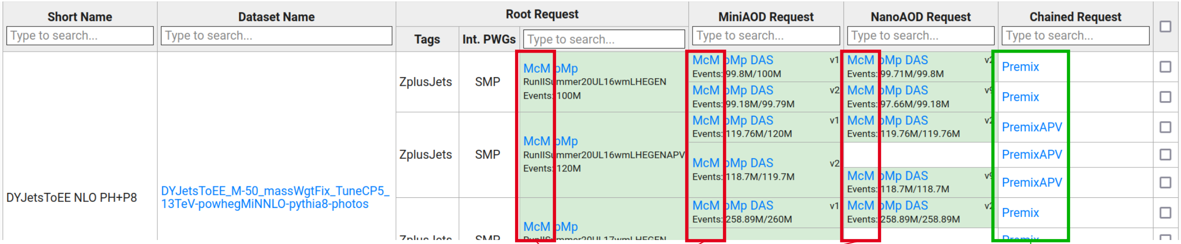
MC production in 2022. Source: `https://cms-pdmv.cern.ch/home/`

CMS produces billions of Monte Carlo events per month...

- Not running `cmsDriver` by hand!
- Main tool for central production: the McM website
- Need samples? Ask your group's Monte Carlo contact

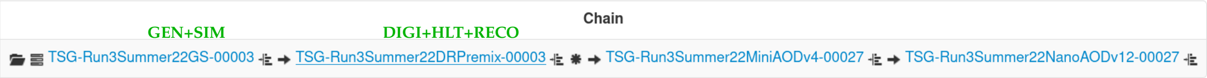
Links to McM

To enter McM, start from Grasp:



Individual steps ("Requests")

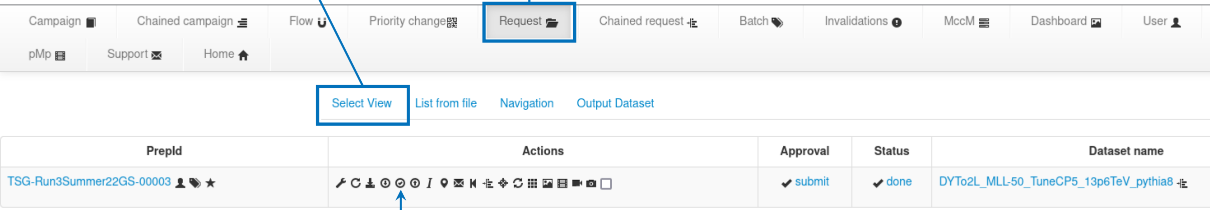
"Chain" of **all requests** for the dataset



McM Request Page

All McM pages look the same, check that you are in the right section

Add or remove columns



Get script with `cmsDriver` commands

You will see fewer buttons depending on your permissions.

Exercise

Produce 200 NanoAOD v12 events looking like the central dataset

`TTto2L2Nu_HT-500_NJet-7_TuneCP5_13p6TeV_powheg-pythia8`

without the cuts from the central sample.

This production uses `CMSSW_12_4_11_patch3` and `CMSSW_13_0_13`.
You need a grid proxy: `voms-proxy-init -voms cms`

Outline :

- Find the dataset in Grasp
- Go to the chained request in McM
- Open each request, get the `cmsDriver` command, run them in sequence
- There are 4 requests and 5 `cmsDrivers`

More info in the next slides!

Detailed Instructions: Root Request

For the root request (the first one), you need to:

1. Find the `cmsDriver` command
2. Download the fragment to the right location (input of the `cmsDriver` command)
3. Modify the fragment to **remove filters**
4. Do `scram b`
5. Run `cmsDriver` (set the seed and number of events, add `--nThreads 8` for speed)
6. Have a look at the produced configuration
7. Use `cmsRun` to produce the events
8. Check the produced file (use `edmDumpEventContent`)

Detailed Instructions: Other Requests

For the second request (DIGI-RECO with Premix pileup):

1. There are two `cmsDriver` commands
2. Set the number of events, add `--nThreads 8`
3. Check the produced files

Third request (`MiniAOD`):

1. **Warning: different CMSSW version!** You need to set the following environment variable for it to work at DESY:
   `export SITECONFIG_PATH=/cvmfs/cms.cern.ch/SITECONF/local/`
2. The command in McM is set to read the central CMS dataset: change to `--filein file:yourAODFile.root`
3. Set the number of events, add `--nThreads 8`
4. Check the produced file

Final request (`NanoAOD`):

1. Change `NANOEDMAODSIM` to `NANOAODSIM` to get flat ntuples
2. Set the number of events, add `--nThreads 8`
3. Plot a few distributions (jet p_T, b tagging score...)

Fourth Part

FastSim

FastSim

There is another simulation workflow in CMS: FastSim

- Lower fidelity but faster than full simulation
- Most variables simulated, notable exception: triggers
- For FastSim, a single request does everything up to AOD
- Example request: PPD-Run3Summer22FSwmLHEPremix-00001
- Add `--fast` to MiniAOD and NanoAOD `cmsDriver` commands

Exercise

Produce 1000 NanoAOD v10 events for the same sample as before, but **using FastSim**. Compare the outputs...

Concluding Remarks

- Tour of CMS information websites about samples
- Intro to the tools used for central production
- Full and fast simulation

Be part of it, join PdmV as a Monte Carlo contact!

Credits

- Original PPD exercise by Salvatore Rappoccio
- Heavily modified with Buğra Bilin for the CMSDAS 2023 at CERN
- Comments by past and current PPD conveners
- Dice picture: Stephen Silver via Wikimedia Commons (public domain)