Experimental Feedback from

CMS

at MC4BSM17

Stephen Mrenna (Fermilab)
for the CMS Exotica group

May 11, 2017

Perspectives on This Talk

The Analyzer What tools do I need to set the best limit?

The Provider How do I use the tools to perform the model scan needed by the analysts?

The Developer How do I evolve my tool to enable the provider? What are the priorities?

Do the tools limit the physics?

My Outline and $0.02

- Mechanics and details of what we do.
- Results and comments on tuning.
- Evolution of software and computing models.

History of This Type of Talk

2006 Exhortation, LHC experimentalist generator wish list, precision on backgrounds, benchmarks

2007 Repeat

2008 No such talk

2009 Extreme corners of phase space, rare backgrounds, new code is hard – LHE is not, need specific models and new ideas

2010 No record of talk

2011 No workshop

2012 Fast detector simulations

2013 Simplified model scans, integrated infrastructure, Pythia8

2014 Results & input and interaction with MC and theory builders essential

2015 RunI results confirmed MC tools. Generating BSM like SM.

2016 Results, MC4BSM 2026

2026?: Schedule Reminder

- Lead-time to produce billions of fully simulated and reconstructed Monte Carlo events is long
  - Pythia-based production for Run 2 started in October 2014
  - LHE-based production started (late) in Feb. 2015
- Inertia from experiments to change things like tunes, pdf’s, etc (needs validation)
- Inertia from developers to change things (balancing problem)

- Understand **how** we use external generator programs in CMS at both a physics and technical level
- Better understanding of our problems and solutions (workflows) $\rightarrow$ more effective use of generators in CMS, suggestions for how we can improve

CMS Software Overview:CMSSW

- Modular C++ application for GEN, SIM, RECO, analysis
- Steered with python-based configuration files
- Consumes/produces root-based EDM files
- Links directly to many **externals** – externally maintained C, C++, fortran, or python software
- External versions tied to CMSSW release

CMS Production Overview: the Grid

- Python-based tools manage large-scale submission of CMSSW jobs to grid resources for central production of Monte Carlo, data processing, etc
- Input and output are assumed to be EDM files (with a few special cases)
- Similar mechanisms for analysis
- CMSSW software and externals available on worker nodes through **CVMFS** (distributes http-based read-only filesystem)

CMSSW: Event Generation

- Basic paradigm: A C++ module ⇒ linked external generator code ⇒ HepMC::GenEvent ⇒ EDM output
- Generator configuration controlled by CMSSW python configuration
- **Advantages:**
  - Uniform configuration and IO mechanism (production tools only have to deal with CMSSW)
  - No intermediate files needed (HepMC::GenEvent is passed along in memory to standard CMSSW/root IO mechanisms or directly to GEANT, which is also called from inside CMSSW)
- **Disadvantages:**
  - Each generator needs a dedicated interface and must be packaged as a CMSSW external
  - Initialization and event generation calls must be possible from within a C++ application
- Pythia, Herwig, Sherpa work nicely (some preference for C++)

Example CMSSW GEN Configuration Fragment

```
import FWCore.ParameterSet.Config as cms

from Configuration.Generator.Pythia8CommonSettings_cfi import *
from Configuration.Generator.Pythia8CUEP8M1Settings_cfi import *

generator = cms.EDFilter("Pythia8GeneratorFilter",
  maxEventsToPrint = cms.untracked.int32(1),
  pythiaPylistVerbosity = cms.untracked.int32(1),
  filterEfficiency = cms.untracked.double(1.0),
  pythiaHepMCVerbosity = cms.untracked.bool(False),
  comEnergy = cms.double(13000.0),

  crossSection = cms.untracked.double(1.92043e+07),

  PythiaParameters = cms.PSet(
    pythia8CommonSettingsBlock,
    pythia8CUEP8M1SettingsBlock,
    processParameters = cms.vstring(
      'HardQCD:all = on',
      'PhaseSpace:pTHatMin = 50  ',
      'PhaseSpace:pTHatMax = 80  ',
    ),
    parameterSets = cms.vstring('pythia8CommonSettings',
                                'pythia8CUEP8M1Settings',
                                'processParameters',
                               )
  )
)
```

CMS Software: LHE Input

- CMS uses own LHE parser
- LHE file can be read as input to a CMSSW job and converted to C++ classes
- LHE information passed to generator using its hooks (e.g. Pythia8::LHAup)
- generator LHE parsers not used
- **Advantage:** Uniform hadronizer-independent storage and access to lhe information
- **Disadvantage:** We have to maintain our own LHE parser

LHE Input for Central Production

- ascii LHE input not ideal
  - metadata not automatically available in data management system, skipping of events is inefficient, etc
- Can handle privately produced LHE files
  - user files ⇒ eos ⇒ EDM files containing the LHE products ⇒ hadronization, simulation, etc)
- Disk space, file corruption, etc, are major issues when dealing with large sets of LHE files

Central production of LHE events

- Madgraph_aMC@NLO, POWHEG, etc don't fit our computing model
- LHE generator code hard to include as an external, since each process requires dedicated (and sometimes dynamically generated) libraries
- However, "externalLHEProducer" can read LHE files into EDM products

Solution: Gridpacks

- Pre-generated/compiled code, and with initial phase space integration results stored in a tarball
- Gridpacks are put in CVMFS and accessed by remote jobs (gridpack location is a configuration parameter of the externalLHEProducer module)
- Minimal and compact external input, and compressed EDM output make very large scale LHE production possible.
- CMS has produced over 30 billion LHE events (before matching) through this mechanism for the initial Run 2 campaign
- We maintain scripts for Madgraph_aMC@NLO (including NLO processes), POWHEG, JHUGen to produce gridpack tarballs based on the appropriate input cards

Gridpack Considerations

- Compiling code on batch workers is discouraged (should be possible to fully precompile everything)
- Long initialization time for event generation is discouraged
- Gridpack size is an issue ($>$ 500MB for the tarball or 5GB decompressed)
  - For Madgraph_aMC@NLO we use lzma compression with very large dictionaries because of large use of space from duplicated code in statically linked executables for each subprocess
- Gridpack generation step needs reliability and reasonable run-time "as the physicist waits" (we can use multi-core machines and/or condor/lsf batch queues to do the phase space integration, but does no good if process is bottle-necked by single-threaded steps, or individual long-running jobs)
- Can we recycle "wrong" grids (e.g. 600 GeV gluino for 1 TeV)

Parameter Scans

- Typical case: gluino/squark pair production (+0,1,2 jets LO) in MG5_aMC@NLO, decay in Pythia, steered by SLHA table
- Produce one gridpack eg. for each gluino mass
- Gridpack and pythia configuration+SLHA table for decays are randomly selected for each luminosity section ($\sim$ 200 events after matching)
- Resulting sample contains a mixture of all scan points
- High granularity of randomization ensures missing events from job failures are randomly and $\sim$ evenly distributed across scan points

```
process.generator = cms.EDFilter("Pythia8GeneratorFilter",
    RandomizedParameters = cms.VPSet( (cms.PSet(
        ConfigDescription = cms.string('T1ggLL_600_1_1'),
        ConfigWeight = cms.double(378.978723404),
        GridpackPath = cms.string('/cvmfs/cms.cern.ch/phys_generator/gridpacks/slc6_amd64_gcc481/13TeV/madgraph/V5_2.3.3/sus_sms/SMS
        PythiaParameters = cms.PSet(
            JetMatchingParameters = cms.vstring('JetMatching:setMad = off',
                'JetMatching:scheme = 1',
                'JetMatching:merge = on',
                'JetMatching:jetAlgorithm = 2',
                'JetMatching:etaJetMax = 5.',
                'JetMatching:coneRadius = 1.',
                'JetMatching:slowJetPower = 1',
                'JetMatching:qCut = 118',
                'JetMatching:nQmatch = 5',
                'JetMatching:nJetMax = 2',
                'JetMatching:doShowerKt = off',
                '6:m0 = 172.5',
                'Check:abortIfVeto = on'),
            parameterSets = cms.vstring('pythia8CommonSettings',
                'pythia8CUEP8M1Settings',
                'JetMatchingParameters'),
            pythia8CUEP8M1Settings = cms.vstring('Tune:pp 14',
                'Tune:ee 7',
                'MultipartonInteractions:pT0Ref=2.4024',
                'MultipartonInteractions:ecmPow=0.25208',
                'MultipartonInteractions:expPow=1.6'),
            pythia8CommonSettings = cms.vstring('Tune:preferLHAPDF = 2',
                'Main:timesAllowErrors = 10000',
                'Check:epTolErr = 0.01',
                'Beams:setProductionScalesFromLHEF = off',
                'SLHA:keepSM = on',
                'SLHA:minMassSM = 1000.',
                'ParticleDecays:limitTau0 = on',
                'ParticleDecays:tau0Max = 10',
                'ParticleDecays:allowPhotonRadiation = on',
                '1000021:tau0 = 1.000000e+00',
                'ParticleDecays:tau0Max = 1000.1',
                'LesHouches:setLifetime = 2',
                'RHadrons:allow = on')
        ),
```

```
SLHATableForPythia8 = cms.string('\nBLOCK MASS # Mass Spectrum\n
# PDG code mass particle\n 1000001 1.00000000E+05 # ~d_L\n
2000001 1.00000000E+05 # ~d_R\n 1000002 1.00000000E+05 # ~u_L\n
2000002 1.00000000E+05 # ~u_R\n 1000003 1.00000000E+05 # ~s_L\n
2000003 1.00000000E+05 # ~s_R\n 1000004 1.00000000E+05 # ~c_L\n
2000004 1.00000000E+05 # ~c_R\n 1000005 1.00000000E+05 # ~b_1\n
2000005 1.00000000E+05 # ~b_2\n 1000006 1.00000000E+05 # ~t_1\n
2000006 1.00000000E+05 # ~t_2\n 1000011 1.00000000E+05 # ~e_L\n
2000011 1.00000000E+05 # ~e_R\n 1000012 1.00000000E+05 # ~nu_eL\n
1000013 1.00000000E+05 # ~mu_L\n 2000013 1.00000000E+05 # ~mu_R\n
1000014 1.00000000E+05 # ~nu_muL\n 1000015 1.00000000E+05 # ~tau_1\n
2000015 1.00000000E+05 # ~tau_2\n 1000016 1.00000000E+05 # ~nu_tauL\n
1000021 6.000000e+02 # ~g\n 1000022 1.000000e+00 # ~chi_10\n 1000023
1.00000000E+05 # ~chi_20\n 1000025 1.00000000E+05 # ~chi_30\n 1000035
1.00000000E+05 # ~chi_40\n 1000024 1.00000000E+05 # ~chi_1+\n 1000037
1.00000000E+05 # ~chi_2+\n\n# DECAY TABLE\n# PDG Width\nDECAY 1000001
0.00000000E+00 # sdown_L decays\nDECAY 2000001 0.00000000E+00 #
sdown_R decays\nDECAY 1000002 0.00000000E+00 # sup_L decays\nDECAY
2000002 0.00000000E+00 # sup_R decays\nDECAY 1000003 0.00000000E+00 #
sstrange_L decays\nDECAY 2000003 0.00000000E+00 # sstrange_R
decays\nDECAY 1000004 0.00000000E+00 # scharm_L decays\nDECAY 2000004
0.00000000E+00 # scharm_R decays\nDECAY 1000005 0.00000000E+00 #
sbottom1 decays\nDECAY 2000005 0.00000000E+00 # sbottom2 decays\nDECAY
1000006 0.00000000E+00 # stop1 decays\nDECAY 2000006 0.00000000E+00 #
stop2 decays\n\nDECAY 1000011 0.00000000E+00 # selectron_L
decays\nDECAY 2000011 0.00000000E+00 # selectron_R decays\nDECAY
1000012 0.00000000E+00 # snu_elL decays\nDECAY 1000013 0.00000000E+00
# smuon_L decays\nDECAY 2000013 0.00000000E+00 # smuon_R decays\nDECAY
1000014 0.00000000E+00 # snu_muL decays\nDECAY 1000015 0.00000000E+00
# stau_1 decays\nDECAY 2000015 0.00000000E+00 # stau_2 decays\nDECAY
1000016 0.00000000E+00 # snu_tauL decays\n##### gluino decays - no
offshell decays needed\nDECAY 1000021 1.973270e-13 # gluino decays\n#
BR NDA ID1 ID2\n1.00000E+00 2 1000022 21\n\nDECAY 1000022
0.00000000E+00 # neutralino1 decays\nDECAY 1000023 0.00000000E+00 #
neutralino2 decays\nDECAY 1000024 0.00000000E+00 # chargino1+
decays\nDECAY 1000025 0.00000000E+00 # neutralino3 decays\nDECAY
1000035 0.00000000E+00 # neutralino4 decays\nDECAY 1000037
0.00000000E+00 # chargino2+ decays\n') ),
```

Standard Configurations for CMS Monte Carlo

Run1

- Pythia6 standalone, Madgraph5+Pythia6 with/without MLM matching, POWHEG+Pythia6, little bit of Sherpa 1.x and Herwig6

Run2

- Pythia8 Standalone
  - Mainly for QCD, especially with additional generator level filters ("multiple hadronization" feature has been added)
  - some more exotic BSM
- POWHEG-BOX + Pythia8 (power showers with emission veto)
  - Mostly for Higgs signals, diboson production, and $t\bar{t}$ production
  - Starting MINLO and NNLOPS

Standard Configurations for CMS Monte Carlo

Run2 (cont)

- MG5_aMC@NLO + Pythia8
  - LO without matching for many exotic BSM signal samples
  - LO with MLM matching (up to 4 additional partons depending on process) used for boosted/multijet phase space for search backgrounds with W/Z/γ+jets, QCD multijet, $t\bar{t}$+jets, and for SUSY signal samples
  - NLO (without merging) used for a few complex processes where extra jets are either computationally expensive (ttW/Z/γ, *ttbb*, etc), or not possible with FXFX (γ+jets, dijets, VBF, etc)
  - NLO with FXFX merging (up to 2 additional partons at NLO) used for Z/W+jets, dibosons, $t\bar{t}$, some Higgs signals
  - CKKW/UMEPS/UNLOPS have technical problems with combining samples (solved now) and weights (maybe solved now also at least for LO CKKW?)

Additional Configurations for CMS Run 2 Monte Carlo

- Sherpa 2.X used for some samples (especially diphoton+jets)
- POWHEG+Herwig++ (wimpy showers) used for systematics vs Pythia8
- MG5_aMC@NLO+Herwig++ (no merging)
- Herwig++ standalone used for QCD and MinBias
- Herwig7 integration in progress
- JHUGen for anomalous Higgs spin/parity studies, H$\rightarrow$ VV decays
- Several other generators used for special samples

Standard Configurations for CMS Run2

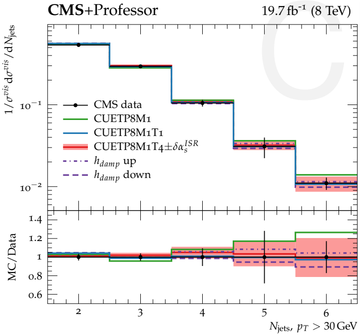
Pythia8 Tune

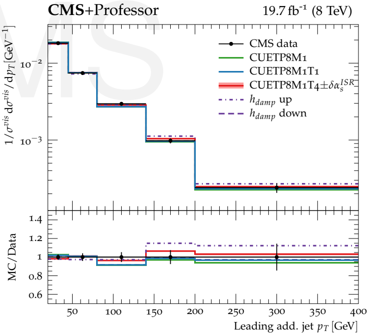
- So far have used Tune CUETP8M1 for most samples (arXiv:1512.00815)
  - Re-tuning of UE parameters on top of Monash, α_S and other shower parameters left untouched
  - In particular this means α_S=0.1365 used for both ISR and FSR in the shower, despite using 0.118 in the ME for NLO samples and 0.130 for LO samples
- Tuning of shower parameters in particular being revisited in future production

Pythia8 Shower Tuning with $t\bar{t}$ (CMS-PAS-TOP-16-021)

- Differences observed between generators, and sensitivity to shower α_s in $t\bar{t}$ production in kinematics/multiplicity of additional jets
- Retune PS ISR α_s and POWHEG hdamp using POWHEG+Pythia $t\bar{t}$ vs dilepton+jets data, yields (much) lower value of $\alpha_s^{ISR} = 0.1108$

Njets (p_T >30 GeV) | Lead jet p_T

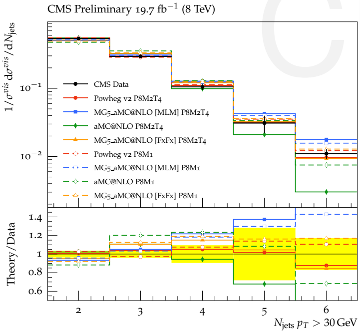




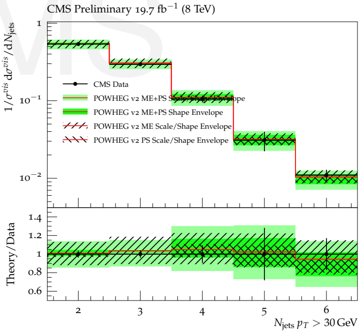
Pythia8 Shower Tuning with $t\bar{t}$ (CMS-PAS-TOP-16-021)

- Smaller α_s^{ISR} somewhat favoured for MG5_aMC@NLO+Pythia8 with FXFX merging, but uncertainties are large
- More systematic cross comparison of total uncertainty between generators would be useful (GEN-17-001 in prep)
- Impact of shower starting scale in mc@NLO configurations?

Generator Comparison



POWHEG+Py8 Uncertainties



Evolution and Revolution in Computing

- MG use of directories as a database is a bottleneck. Unpacking the tarball does not scale (overcome by NOT unpacking?)
- Codes will necessarily need to be multi-threaded or suffer a performance penalty
- Finding applications for GPUs would be nice (also, CUDA code can be readily adapted to OpenMP)

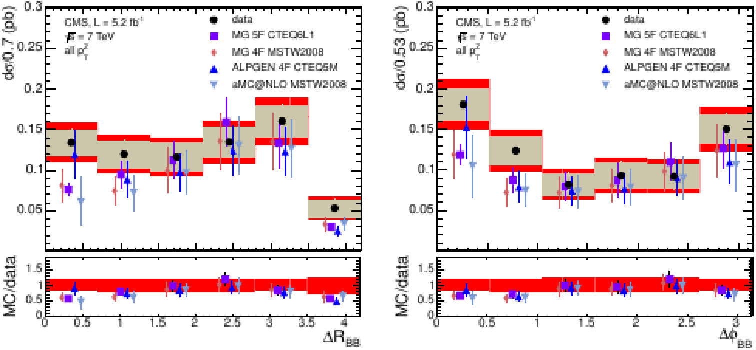
- Phase space integration is a bottle neck (gridpacks are not made on the grid – but testing HTCondor, CMS Connect, MPI, NERSC): New algorithims or multi-threading?
- Models can be passed between tools. Can runtime configurations (process, cuts, scales, ...)?
- Machine Learning (and custom FPGAs) is all the rage. Is there any overlap? Is making a VEGAS grid similar to fitting the sigmoid functions between NN layers?

More, Better, Faster Tuning

- families of tunes corresponding to “an error band”
- non–global tunes focused on “physics windows” – can indicate whether certain parameters are reasonably universal
- include PDF families
- MG+Pythia (Pythia alone will fit deficiencies in its perturbative model with parameters in the non-perturbative models)

- improve the tuning procedure – different surrogate models, eye to parallel processing, adaptive fits, ...
- tuning without unfolding – regions that are more interesting to searches for new physics, including extreme regions of phase space and regions that are sensitive to the detector response

Potentially useful Folded data



Conclusions & \$0.02

- *The mechanics and details of what we do.* Will this scale? Could it be done better?
- *Comments on tuning.* Will this scale? Could it be done better?
- *Evolution of software and computing models.* LHE codes are not written for how the experiments use them. There is inertia to change.

BACKUP

Standard Configurations for CMS Run2: PDF's

- Standardized on NNPDF30+uncertainties so far for most samples, using appropriate LO, NLO, 4fs and 5fs variations
- Additional weights for variations of central pdf+α_S included for $\sim$all POWHEG and MG5_aMC@NLO samples
- Additional weights for alternate pdfs+uncertainties included for POWHEG, and LO MG5_aMC@NLO (was not previously possible at NLO, will be included in future productions)
- (Pythia8-only samples used NNPDF23LO1 and no variations)
- Central and alternate PDF sets will be updated for 2017 production