中国科学院高能物理研究所
Institute of High Energy Physics
Chinese Academy of Sciences

CMS Needs and Concerns for Physics Event Generators

Efe Yazgan
for the CMS Collaboration

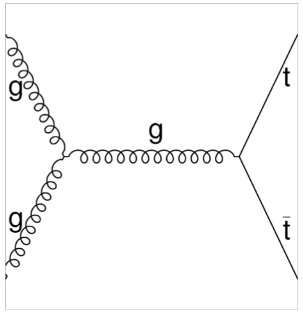
Physics Event Generator Computing Workshop,
26-27 November 2018, CERN

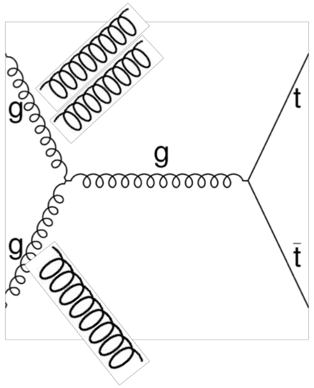
Event Modeling in CMS

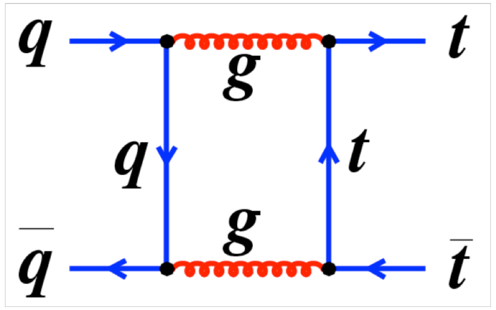
- Most measurements at hadron colliders rely on large scale Monte Carlo production.
  - Understanding and interpretation of data – test SM with more precise and complex calculations.
  - Many cases in which irreducible backgrounds extrapolated to signal phase-space regions for new physics searches through predictions using MC simulations.
- At the LHC, most events are accompanied by additional hard jets from initial or final state QCD radiation.
  - SM measurements
  - Many searches select or veto these extra jets.

NLO/multi-leg/merged MC generators
needed for high accuracy predictions for the LHC

Matrix Element Generation







- Multi-leg LO and NLO consistently matched to the parton shower
- LO: Most commonly used in CMS: MG5_aMC@NLO+Pythia8 with MLM matching
  - Most complex process up to 4 additional jets
- NLO:
  - Most commonly used in CMS: MG5_aMC+Pythia8 with FxFx merging
    - Most complex process up to 2 additional jets at NLO.
  - POWHEG: Commonly used (e.g. almost all top quark samples)
    - Most complex process: MINLO-NNLOPS (ggH->WW) w/ 0-jet at NNLO, 1-jet at NLO, 2-jet at LO and w/ finite quark mass effects.

CMS Software

- Modular C++ application used for event generation, detector simulation, reconstruction and analysis
- Steered with python-based configuration files
- Input/output: root-based EDM files
  - Store run-,lumi-section-, and event-level data
- Links directly to « externals »
  - Externally maintained fortran, python, C, C++, … codes (e.g. parton shower codes Pythia, Herwig, …)
  - External code versions locked to CMSSW release

CMS Central Monte Carlo Sample Production

- Python-based tools for submission of CMSSW jobs to grid resources
- Similar mechanism available for users to submit analysis jobs
- CMSSW + externals available on worker nodes through CVMFS
  - distributed disk system for providing code and libraries to interactive nodes and grid worldwide.

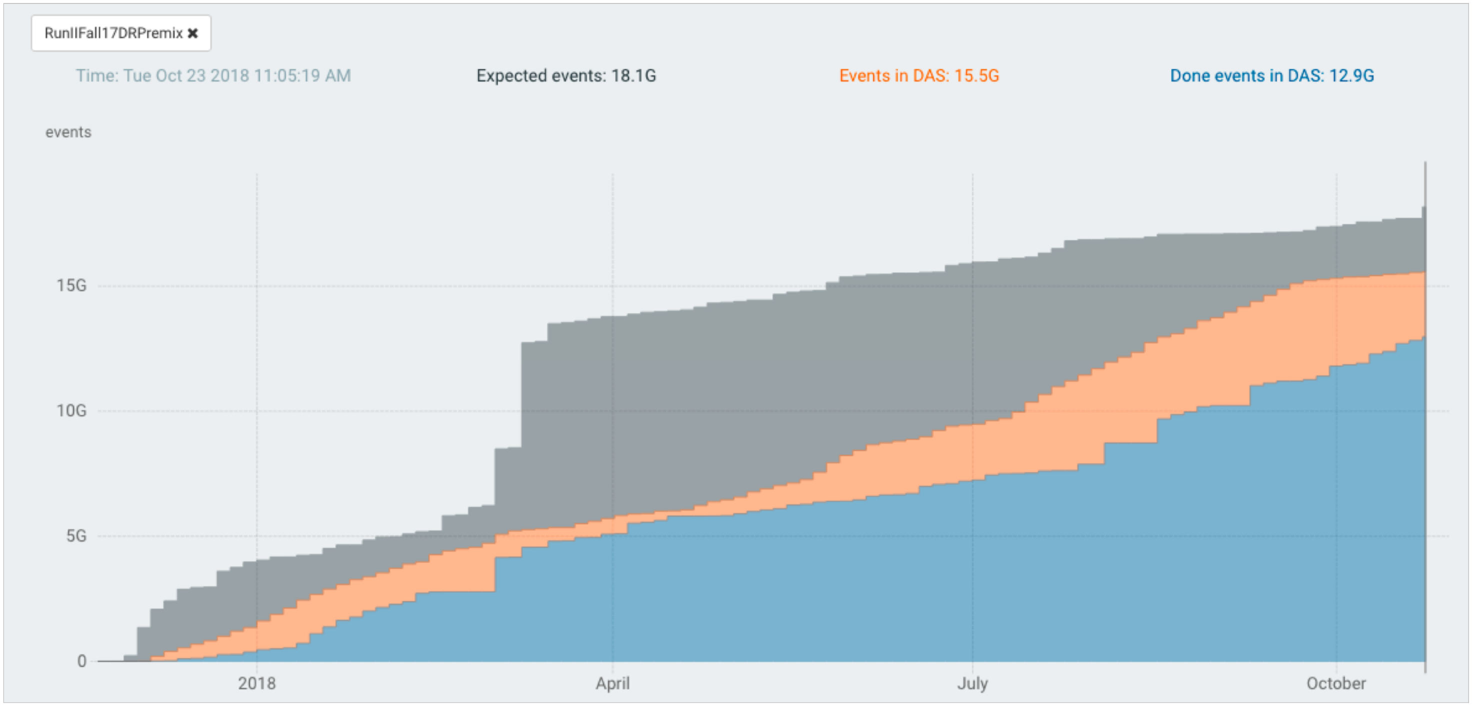
Central Production of LHE Events

- MG5_aMC, Powheg, … called from CMSSW through the externalLHEProducer module
- LHE generator code difficult to include as an external, since each process requires dedicated and sometimes dynamically generated libraries.
    - Solution: gridpacks

Gridpacks

- Pre-generated and compiled code with initial phase space integration results stored in a tarball (with fixed model/run parameters in the standard case).
- Contribution from each subprocess is calculated with high precision.
- The gridpack jobs *randomly include subprocesses based on their relative contributions* to the total cross section.
- Inputs to generated gridpack: Number of events and the random number seed.
- Placed in CVMFS and accessed by remote jobs
- Gridpack location – a parameter of the externalLHEProducer module
- Gridpacks produced in batch systems: cms-connect at Fermilab, and CERN condor now, ...
- In production, significant time spent in untarring the gridpacks
  - MG5_aMC O(100) x slower than Powheg (MG < Sherpack < Powheg)
  - May be less of a problem starting from MG5_aMC 2.6.3
- Number of threads in gridpack production is always 1

Run II GEN Production



- 15 B (+ some other production campaigns ~ 20 B) in 8 months
  - GEN-SIM-DIGI-RECO ~85 sec/evt
  - 60k cores (~1/3 of the CMS production power)
- Multi-leg LO
  - up to ~10s/gen-evt
  - ~10% matching efficiency → 100s/full-sim-evt
- Multi-leg NLO
  - up to ~30s/gen-evt
  - ~30% matching efficiency → 100s/full-sim-evt
  - Large fraction of negative weights of up to ~40% → larger samples!

Run II:

→ GEN not stored for physics samples in disk. GEN-SIM re-produced whenever needed.

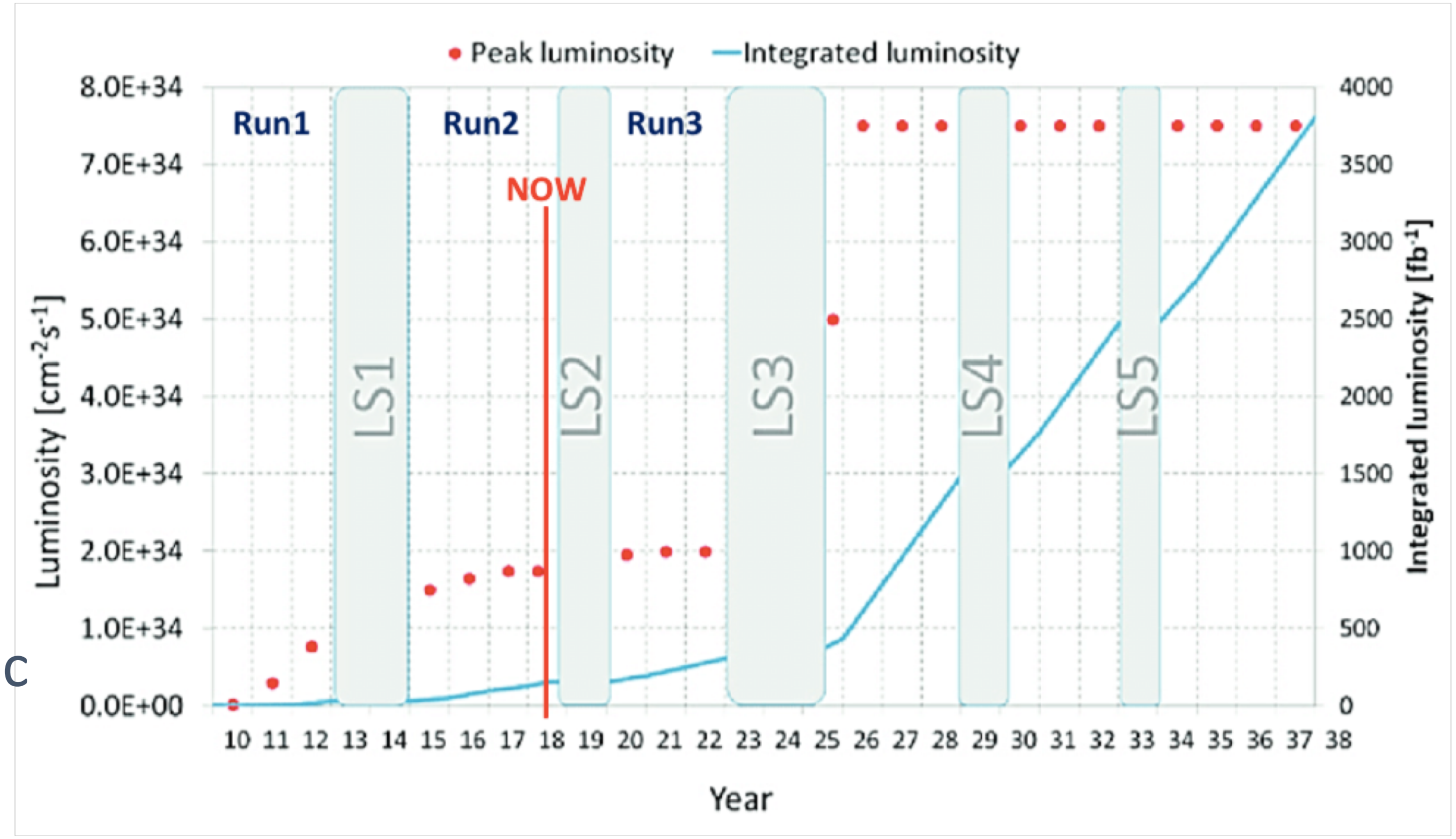
→ Generators ~1-10% of the total CPU

→ Variation due to LO, NLO, NNLO, complexity of the process, or different methods of calculation.

→ Most BSM samples at this point are simulated at LO.

Beyond Run II

- Generation will only be the 3rd CPU consumer after reconstruction and detector simulation, however
- much larger samples and disk space to match data statistics
- precision measurements; top mass, W mass, weak mixing angle, …
- larger alternative samples for systematic uncertainties
- precise differential distributions and tails of the phase space regions.
- more precise calculations: NLO, NNLO, and beyond depending on the process
  → negative weights
- NLO QCD x EWK corrections with high multiplicity final states, for both virtual and real contributions + parton shower



→ requires much larger samples, improved PDFs, …,

→ and RIVETized (or similar) data at the extremes of the phase-space regions to improve modelling

→ *To make it technically very easy, CMS provides particle-level objects in nano-aod – and simple to produce from MiniAod*

→ *GenJets w/ hadron-flavor info*

→ *Dressed leptons*

Use of event weights for Systematic Uncertainties

- Used since sometime in Matrix Elements for PDF and perturbative QCD scales
- Recent Pythia8/Herwig7 versions → weights for parton shower systematics
  - Used in 2017 CMS top quark samples (Pythia8)
  - Used in all 2018 CMS samples that use Pythia8 as shower.

Can never be calculated with weights?

Source	Handle	Weights	Variation	Note/Reference	Dedicated studies
Shower scales	ISR scale (SpaceShower:renormMultFac) FSR scale (TimeShower:renormMultFac)	YES	0.5-2.0 0.5-2.0	FSR variations can be scaled down by √2 from LEP	TOP-15-011, TOP-16-021 TOP-17-13, TOP-17-015, …
ME-PS Matching	hdamp	No	hdamp=1.58m_t +0.66-0.59 m_t	see TOP-16-021	Starting scale variations for MG5_aMC@NLO
Soft QCD	UE parameters	No	UE tune up/down	See TOP-16-021 MPI & CR strength doesn't affect resonance decays	TOP-17-015 GEN-17-001
Color reconnection (odd clusters)	MPI based, QCD-inspired, gluon move	No	different models	CR affecting resonance decays	TOP-17-13, TOP-17-015
Fragmentation	momentum transfer from the b-quark to the B hadron: $x_b=p_T(B)/p_T(b\text{-jet})$	YES	Vary Bower-Lund parameter within uncertainties from LEP/SLD fits	see TOP-16-022 (re-weight x_b)	
Flavor response/ hadronization	Pythia vs Herwig	No	Vary the JES independently per flavour for light, g, c, b.		
Decay tables	B semi-leptonic BR	YES	vary semileptonic BR +0.77%/-0.45%	re-weight the fraction of semi-leptonic b jets by the PDG values (scale Λ_b to match PDG)	

What else can be calculated with weights?

MG5_aMC Bias weights for LO and NLO

O. Mattelaer, arXiv:1607.00763

- Uses an event sample generated with a certain model and associates the original events with a new sample corresponding to a different model with weights.
- *The method requires the original and the alternative model significantly contributes to the same phase space region.*
- Can be used (w/o performing full simulation)
  - to enhance the number of events in the desired phase-space region.
  - to directly test the effect of an alternative model (directly modifying the underlying matrix element)

doesn’t work well if it covers a large phase space:
→ Decreasing weights in a particular phase space region increases it in another region.
→ This is OK in some cases but when large and small weights, difficult to stitch, e.g. W and DY.
➔ Instead, use Njet, VpT binned, unbiased samples - more flexible to fill an insufficiently populated part of the phase space.

➔ ***Not much exercised in BSM processes yet***

Needed Technical Developments

- Understanding timing for each generation step in ATLAS and CMS (effort already started).
- Significant reduction of events with negative weights at NLO
- Faster production for samples with very low filter efficiencies
- Code transition to adapt and optimize for multi-threading, vectorization, GPUs, esp. To reduce memory consumption for merged setups with high number of jets
  - Survey of the codes to understand the best way to move to GPUs and using vectorized code.
- Currently testing multi-threaded event generation with MG5_aMC and Powheg or in general all MC using gridpacks using ExternalLHEProducer
  - Will start extensive tests with different MC configurations in our actual production system soon.
- Can running multiple instances in parallel work for all?
  - Pythia8 OK (w/o external decay package), MG5_aMC being tested, fixed order calculations (e.g. QCD NNLO+NNLL+EWK ttbar and with cuts)?, …

N.B.
→ Multi-threading may be needed for Run III.
→ Without GPUs we may still keep up with increased production needs beyond Run III (assuming Moore's law at ~+20%/year).

Needed Technical Developments

- Faster phase-space integration
  - Neural networks
    - ~100x (w.r.t. VEGAS) better precision for a toy problem with multi-dimensional non-factorizable integrals [*J. Bendavid arXiv:1707.00028]*
    - Unweighting efficiency for e+ e- →q qbar g: ~70% (MG5_aMC ~4%) *[M. D. Klimek, M. Perelstein arXiv:1810.11509]*
  - GPUs *[K. Hagiwara et al. arXiv:0908.4403, arXiv:0909.5257]*
    - cross section calculations ~100 times faster than CPUs
    - parallelizing VEGAS on GPUs. → ~50-60 times faster integration.
  - Is running parallel showers possible?

DNN on GPUs, we might expect ~10-5000 times faster integration depending on the process, perturbative order, and the complexity of the calculation.

→ Gridpacks may become obsolete?

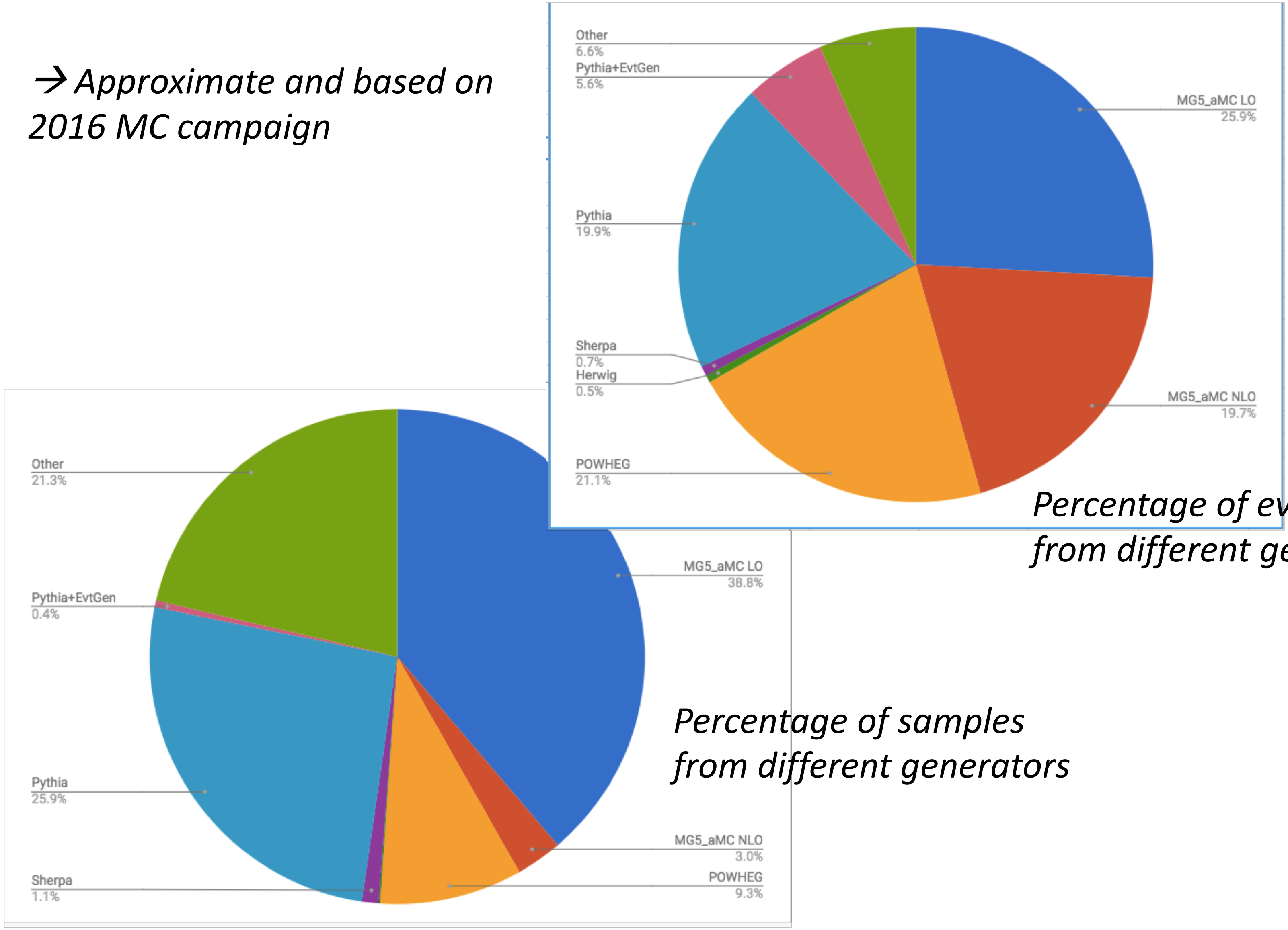
Others

- Use common generator level events between experiments? → x~2 for free event production.
- Find a common approach for MC collaborations for the details of the implementations?
- Can physicists be supported for MC (support) positions?

Additional Slides

Standard Setups for CMS Monte Carlo at Run II

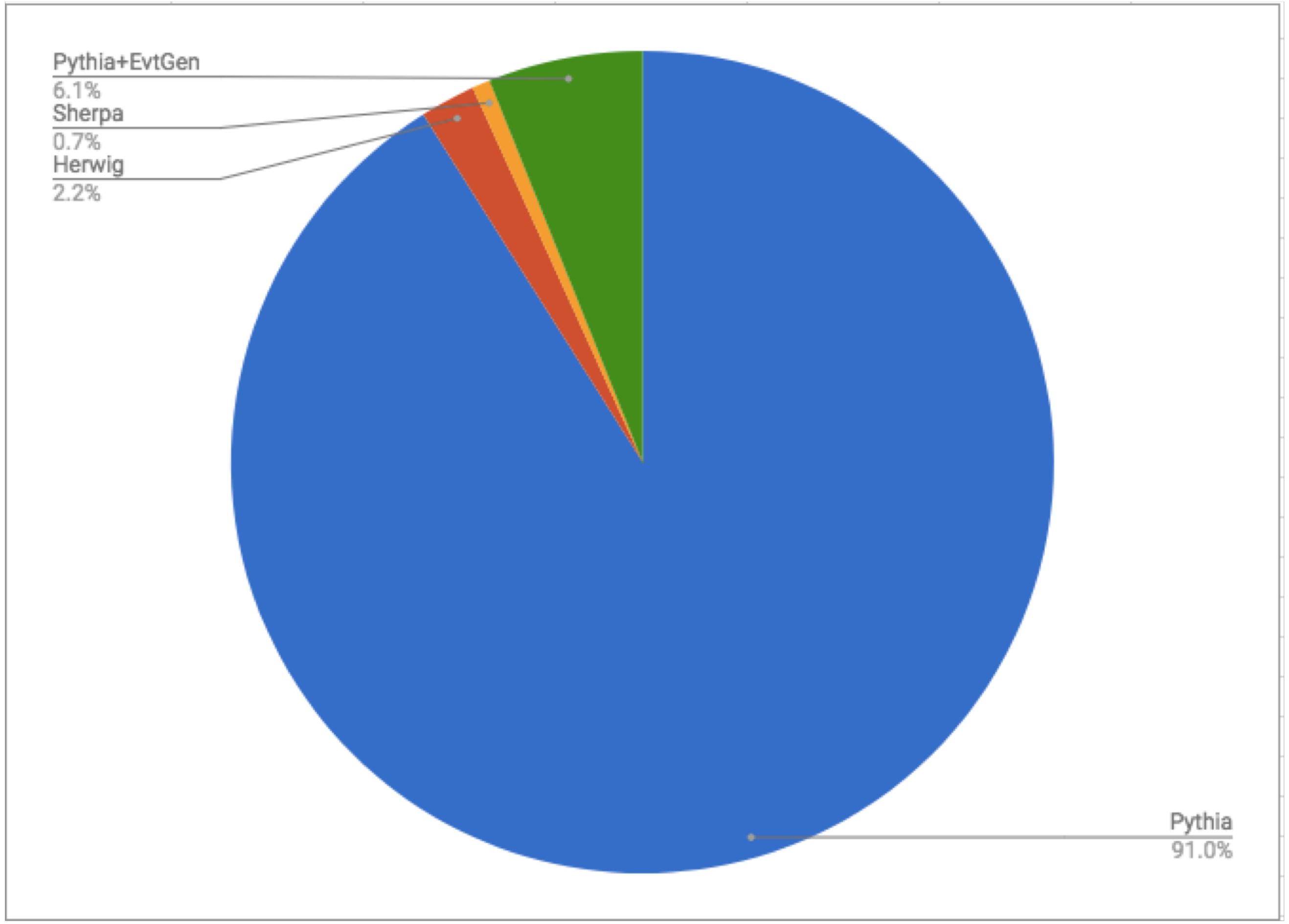
→ *Approximate and based on 2016 MC campaign*



Percentage of events from different generators

Percentage of samples from different generators

Standard Setups for CMS Monte Carlo at Run II – parton shower



→ *Percentage of events from different generators*

→ *Approximate and based on 2016 MC campaign*

Needed Technical Developments

- Faster phase space integration
  - Current: MC integration w/ importance sampling; VEGAS, FOAM
  - Boosted Decision Trees or Deep Neural networks
    - BDT significant improvement over VEGAS but slightly worste than DNN
    - DNN: with much less function evaluations, up to ~4x (w.r.t. FOAM) and ~100x (w.r.t. VEGAS) better precision for a toy problem with multi-dimensional non-factorizable integrals.
      - Additional improvements may come due to the flexibility of loss functions, network architecture, and minimization.
    - NN applied to integrable processes
      - Higher unweight efficiency
      - DNN do not require a choice of coordinates → may work even better at higher orders and in more complex calculations.
      - Next steps: interface the algorithm to MG5_aMC, parton showering.

J. Bendavid
arXiv:1707.00028

M. D. Klimek, M. Perelstein
arXiv:1810.11509

Unweighting efficiency	Scalar→1+2+3	3body decay w/ two resonances	e+ e- →q qbar g
NN	75%	54%	65-75%
MG5	6%	6%	4%

Needed Technical Developments

- Faster phase space integration
  - GPUs is shown to do cross section calculations ~100 times faster than CPUs
  - Phase space integration on GPUs: parallelizing VEGAS on GPUs. → ~50-60 times faster integration.

K. Hagiwara et al. arXiv:0908.4403, arXiv:0909.5257

J. Kanzaki arXiv:1010.2107

$$u\bar{d} \to W^+(\to \mu^+\nu_\mu) + n - gluons$$

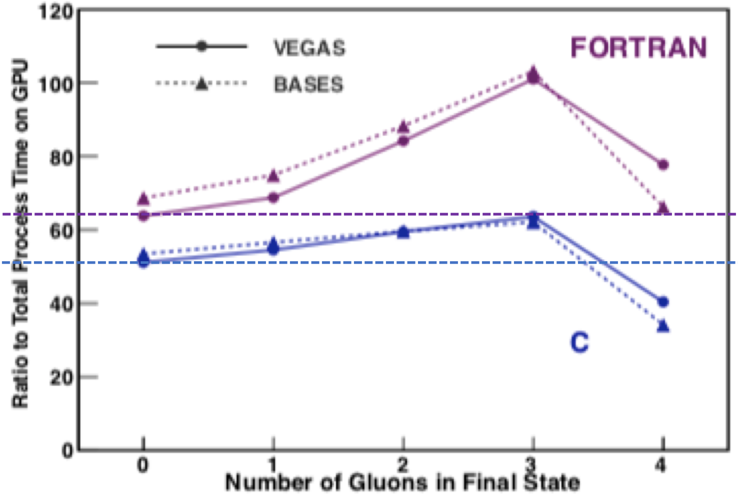
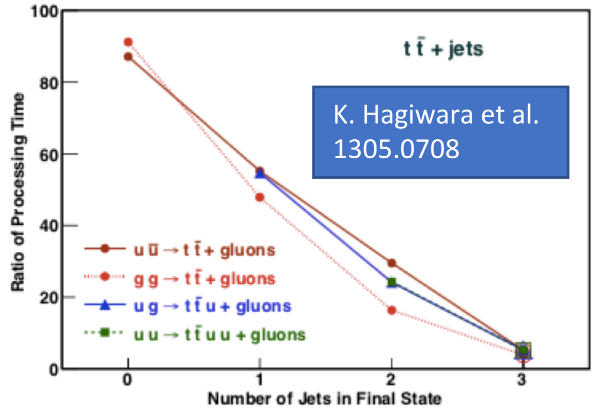


Fig. 2. Process time ratios of FORTRAN and C programs to the corresponding GPU program

When
program b
memories.
programs |
execution
the 4 gluo
on GPU tl



K. Hagiwara et al. 1305.0708

Fig. 6 Ratio of BASES process time (CPU/GPU) for $t\bar{t}+n$-jet production with $t\to b\ell^+\nu_\ell$ and $\bar{t}\to\bar{b}\ell^-\overline{\nu_\ell}$ ($\ell=e,\mu$) for $m_t=175$GeV and Br$(t\to b\ell^+\nu_\ell)=0.216$ in pp collisions at $\sqrt{s}=14$TeV. Event selection cuts are given by Eqs. (8a)-(8c), (10a)-(10b) and (11a)-(11b) and the parton distributions of CTEQ6L1 [14] at the factorization scale of $Q=p^{\mathrm{cut}}_{\mathrm{T,jet}}=20$GeV is used, except for $n=0$ for which the factorization scale is chosen as $Q=m_t$. The strong coupling constants are set as $\alpha_s^{2+n}=\alpha_s(m_t)^2_{\mathrm{LO}}\,\alpha_s(p^{\mathrm{cut}}_{\mathrm{T,jet}})^n_{\mathrm{LO}}$ with $\alpha_s(m_t)_{\mathrm{LO}}=0.108$ and $\alpha_s(20\,\mathrm{GeV})_{\mathrm{LO}}=0.171$.

DNN on GPUs, we might expect ~10-5000 times faster integration depending on the process, perturbative order, and the complexity of the calculation.