Computational challenges for MC event generation

Computational challenges for MC event generation

Andy Buckley
School of Physics & Astronomy, University of Glasgow, Glasgow, G12 8QQ, UK
E-mail: andy.buckley@cern.ch

Abstract. The sophistication of fully exclusive MC event generation has grown at an extraordinary rate since the start of the LHC era, but has been mirrored by a similarly extraordinary rise in the CPU cost of state-of-the-art MC calculations. The reliance of experimental analyses on these calculations raises the disturbing spectre of MC computations being a leading limitation on the physics impact of the HL-LHC, with MC trends showing more signs of further cost-increases rather than the desired speed-ups. I review the methods and bottlenecks in MC computation, and areas where new computing architectures, machine-learning methods, and social structures may help to avert calamity.

MC event generation – i.e. simulation of the fundamental beam-particle scattering in a collider experiment, rather than the following interaction of its decay products with detector systems – is critical to the the LHC physics programme. Compared to previous collider analyses, the design and execution of ATLAS and CMS data analyses in particular have been deeply dependent on predictions of both background and signal distributions in measured observables [1]. This greater reliance has also brought unprecedented demands for precision and accuracy [1], with an associated computational cost.

In this contribution, I briefly review the computational bottlenecks in MC generation for the long-term LHC programme, and a variety of avenues by which they may be addressed. Many issues in this talk and article may be found discussed in full detail in the contributions to the 2018 HEP Software Foundation (HSF) MC workshop: https://indico.cern.ch/event/751693/timetable/.

1. MC generation overview

As "MC" is often used to cover all methods in event simulation using Monte Carlo methods, let me first clarify that the emphasis here is on fully differential "shower + hadronisation generator" (SHG) codes. In these, a quantum field theory matrix element (QFT ME) is sampled to provide parton-level events distributed as the differential cross-section

$$\mathrm{d}\sigma_{\mathrm{hard}}(p_1, \ldots, p_n|Q) \sim |\mathcal{M}(p_1, \ldots, p_n)|^2 \, \mathrm{d}\Phi(p_1, \ldots, p_n|Q)\,, \tag{1}$$

where the p_i are the outgoing leg momenta (we here ignore the coupled initial-state phase-space term which probes the parton density functions (PDFs)), Q is the renormalisation scale of the hard-process ME $\mathcal{M}$, and $\mathrm{d}\Phi$ is the phase-space density for the given configuration. The matrix

[1] The two concepts are distinct: precision reflects the formal order of a calculation (and hence technical sophistication) and accuracy the resulting uncertainty on predictions and the agreement with data.

element is composed of a finite number of fixed-order terms, expanded in the strong-force coupling α_s (and increasingly also in the electroweak couplings), i.e.

$$\mathrm{d}\sigma_{\mathrm{hard}} = \mathrm{d}\sigma^{\mathrm{LO}}_{\mathrm{hard}} + \alpha_s(Q)\,\mathrm{d}\sigma^{\mathrm{NLO}}_{\mathrm{hard}} + \alpha_s^2(Q)\,\mathrm{d}\sigma^{\mathrm{NNLO}}_{\mathrm{hard}} + \cdots . \tag{2}$$

The mod-squaring of the matrix element in eq. (1) mixes amplitudes of different orders in α_s, so that e.g. the $\mathrm{d}\sigma^{\mathrm{NLO}}$ cross-section correction includes both the square of a one-emission ("real") amplitude, and the cross-term composed of interference between a one-loop ("virtual") amplitude and the Born amplitude.

This few-body process is then systematically improved toward a realistic event as would be observed by an ideal collider experiment, by use of various approximate corrections which distinguish SHGs from ME-only generators. These most notably include "parton shower" (PS) QCD cascades, inclusion of multiple partonic interactions within the *pp* collision, and modelling of the non-perturbative hadronisation process by which coloured partons are resolved into a collection of primary hadrons. This scheme may be represented formally as

$$\mathrm{d}\sigma \sim \mathrm{d}\sigma_{\mathrm{hard}}(Q) \times \mathrm{PS}(Q \to \mu) \times \mathrm{Had}(\mu \to \Lambda) \times \ldots , \tag{3}$$

where the sequence $Q \to \mu \to \Lambda$ illustrates the evolution down in energy scale from the ME hard-process/factorisation scale Q, through the renormalisation (etc.) scales of the PS iterations, and eventually cut off by the dominance of non-perturbative physics at the QCD hadronisation scale, Λ. One of the dominant issues in SHG physics is immediately clear from eqs. (2) and (3): extra radiation can be generated both by fixed-order NLO, NNLO, etc. corrections *and* by the PS. The fixed-order ME computations include full QFT correlations and are important for describing the few dominant, widely-separated energy flows in collider events, while the PS "resums" large numbers of perturbative QCD emissions in the less well-separated collinear and soft (low-momentum) phase-space regions where most emission probability resides. We must be careful not to double-count these radiative corrections.

The last 20 years have brought a sea-change in the nature of SHG modelling precision. Back in 2000, when LHC operations were expected to start in around 5 years, the majority of exclusive event generation was still performed using Born-level matrix elements dressed by parton showers either from the Pythia [2] or Herwig [3] Fortran codes. These showers were both based on massless $1 \to 2$ parton splitting functions, with corresponding well-understood deficiencies in phase-space coverage, and the need for an *ad hoc* "reshuffling" scheme to restore Lorentz symmetry to the post-shower event. The shortcomings of the Born-level matrix element were also well-understood, and addressed in several cases (notably electroweak boson production processes) by "matrix element corrections" which added the $V + 1$ jet matrix element in a consistent way, without double-counting PS and ME contributions to the same phase-space: these paved the way for further increases in parton-showered ME precision. Soft-physics modelling was also in a state far separated from that today, with Pythia having laid the ground for an eikonal treatment of multiple scattering [4], followed by the similar Jimmy model[5] to replace Herwig's ageing UA5 parametrisation of the underlying event.

This state of affairs proved sufficient for analyses at HERA and similar colliders, but Run 2 of the Tevatron drove requirements for higher-precision matrix elements consistently interfaced to parton showers. With official first releases around 2002, the Alpgen and MadGraph [6, 7] programs were the pioneers for "multi-leg" tree-level corrections to the Born ME, introducing the cone-based MLM ME-PS matching scheme, followed by the developments of the CKKW matching scheme for first the Amegic ME generator and later Sherpa [8, 9]. The two issues here were again ones of double-counting phase-space population, both between different fixed-order processes ("merging") and between the fixed-order MEs and the resummed PS ("matching").

On the same timescale, the MC@NLO program [10] was the vanguard effort in interfacing one-loop QCD matrix elements (which for full consistency demand also the tree-level single real

emission matrix element) to parton showers[2]: this saw heavy use for a few processes, notably top-quark and Higgs physics, but a tight coupling between the NLO ME subtraction scheme and the PS algorithm delayed the "NLO explosion" of the LHC era until the realisation of the shower-independent POWHEG scheme [11, 12].

2. The rising CPU cost of MC

The availability of both single-emission NLO SHGs and multiple-emission LO SHGs, driven by increasing automation of numerical ME calculations, led to a revolution in experiment expectations of SHG simulation around the LHC startup. Since 2010, MC generation has gone from being a frequently trivial element of an experiment's CPU budget to, particularly in the cases of the core SM V+jets and top quark production processes, a substantial consumer in the range of 15–20% of experiment CPU budget. The driver of this CPU increase has been the availability of the complex multileg and NLO processes – by now inevitably combined into multiple-emission NLO generators e.g via the MEPS@NLO formalism [13, 14] implemented in SHERPA, and the FxFx [15, 16] one in MADGRAPH5 [17], with towers of shower-matched NLO subprocesses up to some number of final-state particles, followed by even more LO ME subprocesses. MC is no longer cheap.

This trend is in contrast to the normal direction of travel for CPU budgets at the LHC experiments because, while the intrinsic complexity of reconstruction and simulation have increased relatively slowly due to increasing pile-up rates (ameliorated by algorithmic and implementation improvements), the leaps in formal precision available as exclusive event simulations come at exponentially increasing cost in numerical calculation. This has been a particular issue for the ATLAS experiment which makes heavy use of the SHERPA event generator: as will be discussed, it is currently more demanding of CPU by comparison with MADGRAPH5_AMC@NLO which dominates CMS simulation budgets. The single-emission NLO POWHEG generator, and a myriad of other MC codes, are used by both experiments, but with much smaller CPU consequences – in this summary we will focus on the most expensive computations.

As shown in Figure 1, this trend cannot continue: the CPU requirements of the High-Luminosity LHC programme are already insufficient for Grid-based MC generation alone, even with optimistic assumptions about evolution of computational purchase power and the ability to obtain algorithmic speed-ups. A further step change in formal precision, e.g. from 1-loop NLO to 2-loop NNLO as the core-process standard for SM SHG generation, may come at such unacceptable cost that it is of academic interest only, priced out of the straitened WLCG budgets by a further exponential step in cost per event.

It is clearly unthinkable that the vast public investment, and the scientific and engineering achievements, of the long-term LHC programme be limited in physics impact by an inability to generate sufficient MC event statistics at the required precision. Something needs to change: in the rest of this article I will summarise the most promising candidates.

3. Challenges and innovations in matrix element phase-space integration

At the core of MC generation, and the most expensive part for high-multiplicity processes, is the integration – synonymous with asymptotic sampling – of the (squared) QFT matrix element over the allowed phase-space of incoming and outgoing legs' momentum configurations, cf. eq. (1).

This is intrinsically a complex problem because the squared matrix elements are extremely "peaky" functions: the majority of their probability density is located in the vicinity of kinematic

[2] Notation is often used loosely in this area: following unofficial convention we will here use "NLO" to mean matrix elements consistently squared at one-loop order, and "LO multileg" for tree-level corrections beyond the leading (Born) order.

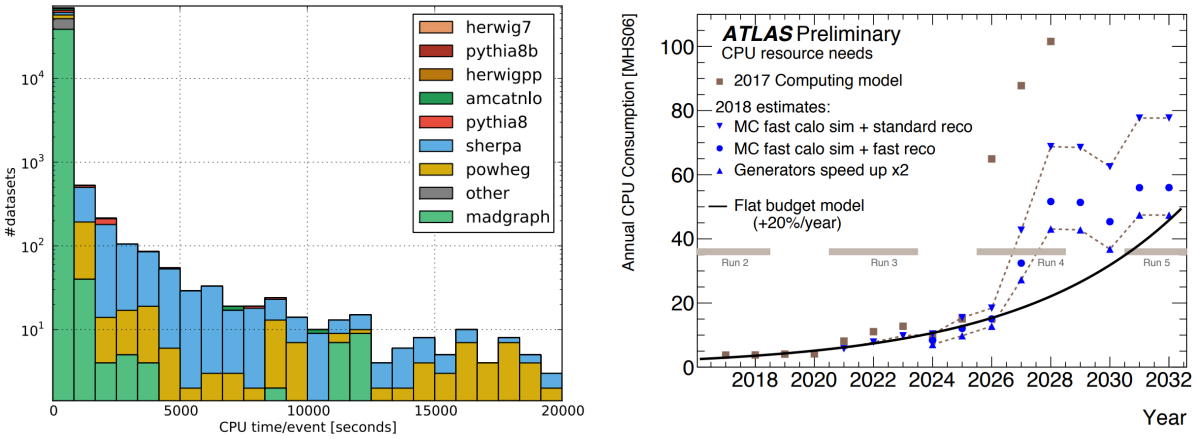

Figure 1. Left: 2015 ATLAS CPU time/event distribution and generator breakdown, plot from Josh McFayden. Right: ATLAS CPU demand vs. availability projection to HL-LHC operation.

divergences, and naïve "flat" sampling of the phase-space will produce abominable statistical convergence. Even with improved sampling proposals, such issues are seen in the form of a wide distribution of sample *weights* or equivalently a poor efficiency for generation of unweighted events – both major problems for LHC consumers of precision MC generators.

3.1. Sampling and event weights

Efficient MC generation requires efficient sampling of the peaky ME function over the partonic phase space. The ideal sampling would be to draw the samples themselves from the asymptotic ME distribution, i.e. knowing the answer before we began! In practice, of course, this is not known *a priori* and so approximate proposal distributions are needed, with standard sampling techniques used to recover the asymptotic distribution by discarding or re-weighting the samples from the approximate one. The aggregation of these imperfect phase-space mappings is the major cause of poor efficiencies in MC event sampling at high fixed orders in α_s.

The ideal, maximally efficient proposal distribution would by definition always have unit weights (the proposal density exactly matches the real one) but in practice the sample weights have a tail to lower values (the proposal included phase-space points which were uninteresting). Example weight distributions for SHERPA multi-leg W + jets event sampling are shown in Figure 2. Greater than unit weights are an even worse problem in principle: the proposal density underestimated the maximum, probably due to a failure in pre-sampling. Both problems feed into observables computed from the sampled events, as poor statistical convergence and as single-event spikes respectively.

In practice weighted events are often difficult to use for experimental purposes because, unlike for most phenomenology purposes, the bulk (or at least a comparable amount) of their CPU cost is still to come in the form of detector simulation. It is usually a better strategy to *unweight* events to obtain a (still unbiased) sample of high-weight events than to spend processing power expensively running Geant 4 for events with unrepresentative phase-space configurations. This sample rejection from broad distributions of weights leads to a further inefficiency which, in combination with already CPU-intensive computation of the ME value for each sample, explains the rocketing CPU cost of state-of-the-art SHG MC event samples. Some current processes can even take $\mathcal{O}$(24 hours) of CPU time per event!

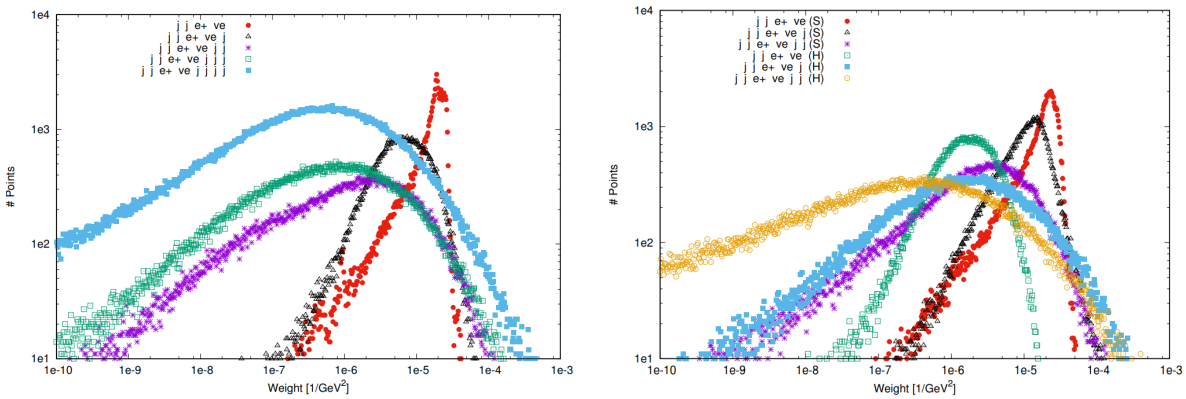

Figure 2. Sherpa event weight distributions for LO (left) and NLO (right, decomposed as MC@NLO soft/hard event classes) for different $Wjj + n$-jets ME multiplicities. The broadening of distributions toward lower weights with higher multiplicity, particularly at NLO, reflects the increasing difficulty of approximating the ME with current sampling proposal distributions. Plots from Marek Schoenherr.

3.2. Sampling improvements

The now-standard approach to phase-space integrals is transformation of the integral to reflect the naïve singularity structure of the cross-section, which typically diverges for soft and collinear configurations. This pattern continues into higher-order LO phase-space, with probability density increasing as legs' momenta become either very small, or parallel to another leg. The "multi-channel" decomposition of the integral into the weighted sum of its pole structures was a key step in automation of matrix element calculations [7], but its imperfections become evident at high multiplicities (where the combinatorics of divergences are formidable), and due to further modifications of the divergences by loop amplitudes.

A particular limitation here is the widespread use of the VEGAS adaptive integration/sampling algorithm for ME/PS integration [18]. This allocates dynamically resized bins in each integration parameter independently, with many smaller bins allocated to regions of (dynamically discovered) probability density. VEGAS gives good performance for separable distributions (provided the factorization is identified in advance), but is unable to cope efficiently with non-factorisable integrands. This issue has been realised for some time, but has not yet received mainstream attention in MC codes – perhaps for reasons to be discussed in Section 7. The obvious step up in sophistication is the multidimensional adaptive binning scheme of FOAM [19], which uses nested hypercubes in place of single-dimensional bins, cf. Figure 3 and hence has greater purchase on moderate-dimensional non-factorisable integrands. But still the scaling of cross-section integration is demanding.

An exciting prototype study [20] has applied machine learning (ML) to this problem, built on the identification of first boosted decision trees (BDTs) and latterly deep neural networks (DNNs) as generalisations of hypercube binning. This approach has not yet been applied in anger to MC integration, as significant technical work is required to rework the integration machinery of a real-world generator code, but the preliminary study's use of a standard "Camel" awkward function in 4 and 9 parameter dimensions produced remarkable gains in variance reduction for fixed sample number, relative to both VEGAS and FOAM, as shown in Table 1. This pathfinding work raises the possibility of ML-assisted integration coming to the rescue in the battle of MC precision vs. CPU budget.

A study with similar motivations was performed in a Masters thesis [21], this time employing several ML techniques in an attempt to specifically learn the integration phase-space of the

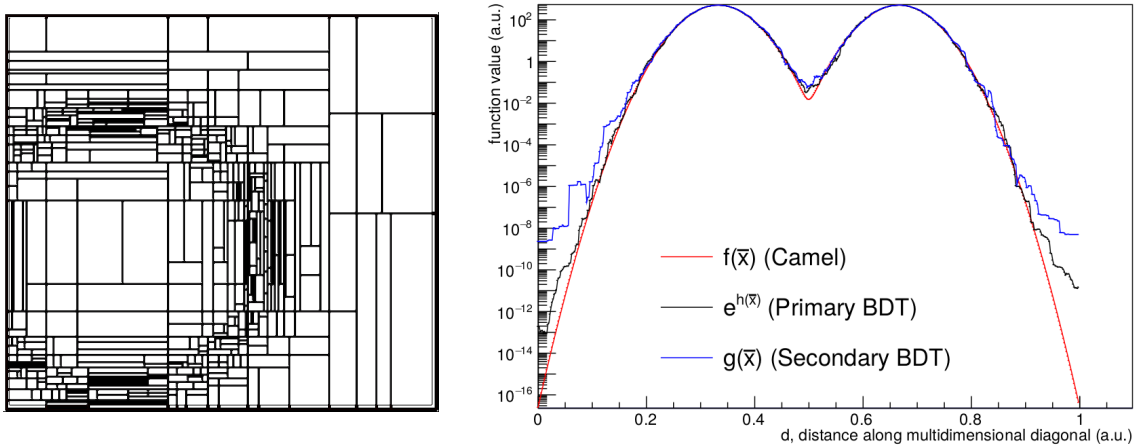

Figure 3. Left: an illustration of 2D FOAM for improved sampling of a non-factorisable 2D ring distribution. Right: convergence of the multidimensional "Camel" function using the BDT-integration method (deviations from optimal visible on tails and between the humps).

Algorithm	Number of evaluations	$\sigma_w/\langle w\rangle$
4D Camel function		
VEGAS	300k	2.820
FOAM	3.8M	0.319
Generative BDT (staged)	300k	0.077
Generative DNN (staged)	294k	0.030
9D Camel function		
VEGAS	1.5M	19.000
FOAM	–	–
Generative BDT (staged)	3.2M	0.310
Generative DNN (staged)	294k	0.081

Table 1. Performance of the best BDT and DNN models for integrator improvement, compared to VEGAS and FOAM (where feasible) on 4- and 9-dimensional Camel test functions.

$Z + q\bar{q}gg$ process. This, again preliminary, study illustrates the importance of domain knowledge to application of ML machinery, with a systematic study of three distinct bases in event-kinematics features, exploiting knowledge of Lorentz invariance and other physical principles in attempts to find the optimum parametrisation. Again there were indications of significant gains in performance, with a factor of 750 speed-up between sampling from the ML model as compared to standard multi-channel sampling of the full matrix element... but at the cost of a $\times 3$ mis-modelling of the weight distribution. This is too inaccurate to use the ML itself as a fast MC generator, but compared to the weight distributions in Figure 2 it seems again that ML techniques may have important future roles to play in efficient generation of ME-integrator proposal densities.

3.3. NLO subtraction and negative weights

In addition to the problem of broad weight distributions, and the resulting unweighting inefficiencies, the consistent treatment of loop amplitudes in NLO MC generation additionally introduces problematic rates of *negative weights*.

These arise because both the real and virtual $\mathrm{d}\sigma$ corrections have infrared divergences, in the soft and collinear limits of the real and loop momenta. In a tree-level calculation, the fixed-order real divergence is typically hidden by a phase-space cut, with the intention that approximate resummation of multiple emissions via a parton shower will consistently fill the divergent phase-space (this is the role of "LO matching"), but full NLO cross-section normalisation requires consistent combination of both fixed-order terms (as well as the PS again – "NLO matching"). Following the Block-Nordsieck, YFS, and KLN theorems [22, 23, 24, 25], in the indistinguishable limit of an unresolveable real emission, the two opposing fixed-order singularities cancel, leaving a finite residual. As infinities cannot be directly worked with in numerical codes, this is canonically implemented via *subtraction* of the opposing divergence structure $\mathrm{d}\sigma_\mathrm{S}$ from both the real and virtual terms, making them independently finite:

$$\mathrm{d}\sigma = \mathrm{d}\sigma_\mathrm{B} + \mathrm{d}\sigma_\mathrm{NLO} \tag{4}$$

$$= \mathrm{d}\sigma_\mathrm{B} + (\mathrm{d}\sigma_\mathrm{V} + \int \mathrm{d}\Phi_1\, \mathrm{d}\sigma_\mathrm{S}) + (\mathrm{d}\sigma_\mathrm{R} - \mathrm{d}\sigma_\mathrm{S})\ . \tag{5}$$

Of course, these singular phase-space regions are the same ones whose difficulty in mapping led to ME integration inefficiencies, and the same problem – only more-so – plagues NLO calculations, with the new issue that insufficient sample densities can produce negative cross-section estimates via the subtraction scheme. Negative weights essentially count double toward generation inefficiencies because not only do they not add statistical convergence but they actively subtract it, cf. the variance formula $\sigma^2 = \langle \sum w^2 \rangle - \langle \sum w \rangle^2$ in which the first term always increases but the second will be reduced by a mix of positive and negative weights. The MC@NLO subtraction formalism (which includes PS splittings in the subtraction) naturally induces negative-weight fractions of around 25%, corresponding to an effective halving of the statistical power of an event sample.

The effect of higher complexity and negative weights at NLO were illustrated by Valentin Hirschi to this workshop, in his demonstration of generation CPU for addition of 1 and 2 extra gluons to the $d\bar{d} \to ZZ$ process: at LO this scales rapidly from 7 µs to 35 µs to 220 µs, and are respectively made slower by factors of 10^2, 10^3, and 10^4 at one-loop order.

In the absence of a revolutionary new formalism for NLO calculations, the best hope is again that ML or other improvements in phase-space mapping will reduce the incidence of negative weights... or, as will be discussed in Section 7, to perhaps entirely reconsider the LHC experimental demand for SHG NLO events in favour of less sophisticated, but pragmatic, approximations.

4. Matrix-element merging

So far we have focused on ME phase-space integration as a leading bottleneck for high-multiplicity process generation. But in particular for the Sherpa event generator, the combination of samples from different-multiplicity matrix-element terms also adds a very significant computational cost. The concept behind merging is again (as is almost everything in pQCD MC) to use the highest-accuracy calculations in appropriate phase-space regions – i.e. the most sophisticated matrix-elements possible for configurations with hard, widely separated partons, and less sophisticated ones enhanced by the PS in soft and collinear regions where multiple emissions dominate – and to avoid double-counting in the process.

This is achieved by slicing the phase-space into orthogonal regions in which each kind of effect dominates – an unphysical distinction, but a necessary one in a world of finite intellectual and computational resources. Since QFT singularities again drive the relative dominance of fixed-order vs resummation effects, the CKKW merging procedure in particular uses a modification of the k_t jet clustering scheme (adapted to be aware of Sherpa's PS splitting functions) to delimit

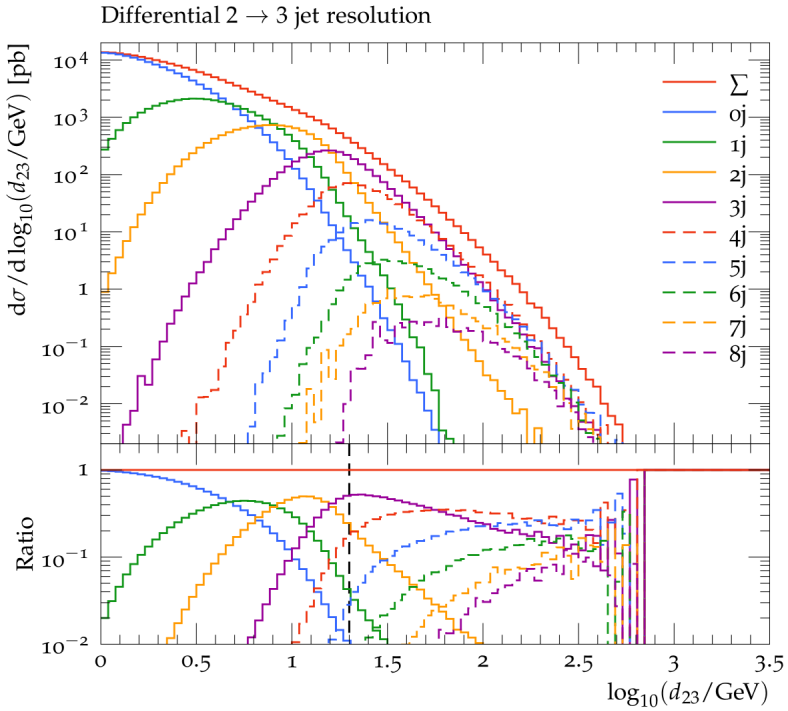

Figure 4. Illustration of ME merging in Sherpa, showing the contributions of up to 8 additional ME partons above the Born level, summing to an inclusive distribution. The smooth suppressions of the higher parton multiplicities at low splitting scales d_{23} are introduced by the CKKW phase-space slicing.

these phase-space slices, illustrated by Figure 4. But this algorithm is itself CPU-expensive! The rapid CPU scaling of Sherpa's components in LO and NLO W production, with increasing numbers of ME jets, is shown in Figure 5.

In fact, direct process-to-process comparisons of Sherpa to MadGraph5 indicate a factor of $\times 4$ performance difference, which can be reduced by more than a factor of 2 simply by using a simple scale-based calculation in place of the formally more accurate CKKW clustering, with minimal observable effect. This process-specific approximation, and similar insights for other bulk SM background samples used by the LHC experiments, is the basis for much of ATLAS' hoped-for MC speed-up by a factor of two in the HL-LHC era: not all MC improvements must come from brute-force calculation, but also judicious use of approximations.

5. Machine learning in ME/PS matching

Judicious approximations bring us to a last consideration in the physics of state-of-the-art SHG generation. I wish to highlight for once not a performance concern, but an example of how ML techniques may allow "missing" physics to be filled in without explicit calculation.

The same issues of double-counting and of using jet clustering to obtain formally more precise scale estimations are found in not just ME merging but also the matching of fixed-order MEs to multiple-emission parton showers. The Minlo (multi-scale improved NLO) method is an approach to improve the Powheg NLO matching scheme by discovery of an event-specific ME emission scale and matching the fixed-order real emission to the Sudakov form factor for resummed QCD emissions at the same scale.

In Reference [26], this approach is taken in the matching of the NLO single-top + jet (STJ)

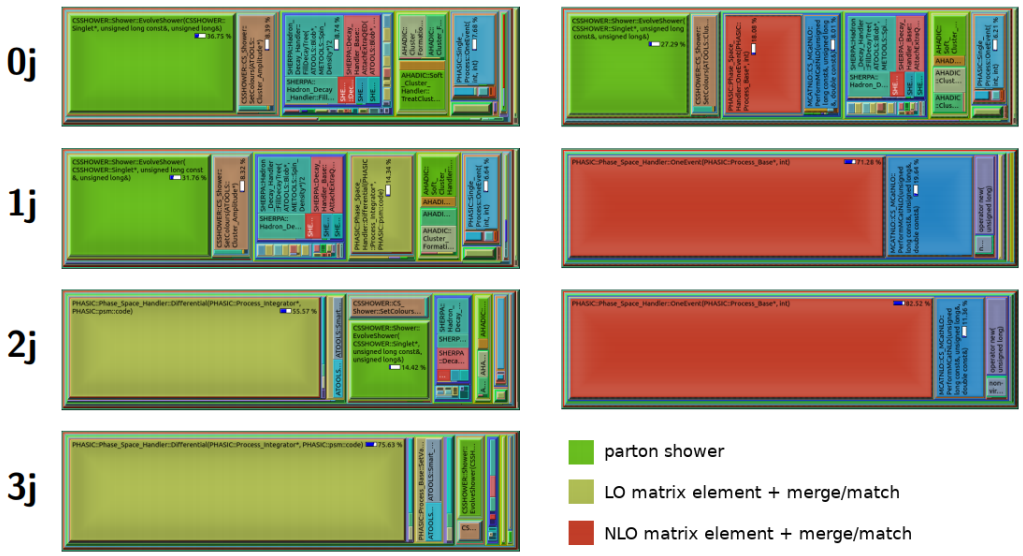

Figure 5. Valgrind CPU maps of Sherpa $W + n$-jet event generation for LO (left) and NLO (right) event generation. The rapid dominance of ME sampling and merging with increasing multiplicity, especially at NLO, is clearly visible. Plots from Marek Schoenherr.

process to a parton shower. The MINLO matching cross-section is given by

$$\mathrm{d}\sigma_{\mathcal{M}} = \Delta(y_{12}) \left[\mathrm{d}\sigma^{\mathrm{STJ}}_{\mathrm{NLO}} - \Delta(y_{12})|_{\alpha_s} \mathrm{d}\sigma^{\mathrm{STJ}}_{\mathrm{LO}}\right] , \tag{6}$$

where $\mathrm{d}\sigma^{\mathrm{STJ}}_{\mathrm{LO/NLO}}$ are the LO and NLO STJ cross-sections, and $\Delta(y_{12})$ and $\Delta(y_{12})|_{\alpha_s}$ are respectively the Sudakov form factor and its α_s term, evaluated at the k_t-clustering scale $\sqrt{y_{12}}$. As intended, this method replaces the fixed-order divergence of the STJ cross-section (as the extra jet becomes unresolved) with a smooth Sudakov suppression, giving STJ predictions accurate at NLO+NLL (next-to-leading logarithm resummation). But there is a remaining defect: this unresolved-jet limit can be understood as the ST limit of the STJ process, but the MINLO construction is not automatically accurate at NLO+NLL for ST. To do so, resummation terms at NNLL would need to be included in the Sudakov factors $\Delta(y_{12})$ in eq. (6). Calculation of such terms is in general a major exercise, which has not yet been automated to the same extent as fixed-order NLO calculations.

This is where machine learning comes in: ML techniques can in principle construct the missing Sudakov function by demanding boundary-condition consistency between the STJ and ST process limits. To achieve this the authors of Ref. [26] propose an "STJ*" ansatz of modifying the exponent of $\Delta(y_{12})$ by an $\mathcal{O}(1)$ function of the Born kinematics, $\mathcal{A}_2(\Phi)$,

$$\ln \delta\Delta(y_{12}) = -2 \int_{y_{12}}^{Q^2_{bt}} \frac{\mathrm{d}q^2}{q^2} \left(\frac{\alpha_s}{2\pi}\right)^2 \mathcal{A}_2(\Phi) \ln \frac{Q^2_{bt}}{q^2} , \tag{7}$$

such that the ST differential cross-section is given by

$$\frac{\mathrm{d}\sigma^{\mathrm{ST}}_{\mathrm{NLO}}}{\mathrm{d}\Phi} = \int \mathrm{d}y_{12} \, \frac{\mathrm{d}\sigma_{\mathcal{M}}}{\mathrm{d}\Phi \, \mathrm{d}y_{12}} \, \delta\Delta(y_{12}) . \tag{8}$$

The estimation of $\mathcal{A}_2(\Phi)$ then corresponds to a deconvolution of the STJ cross-section with $\delta\Delta(y_{12})$ over the Born phase space, and was performed using an optimised neural net architecture

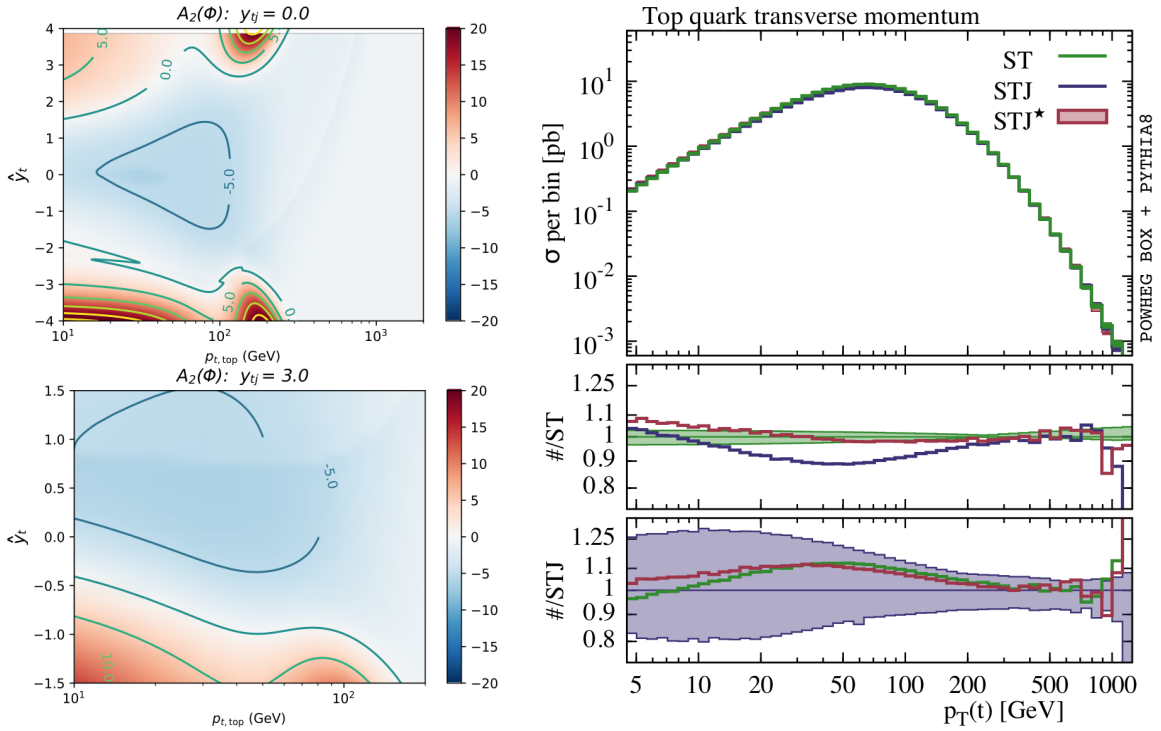

Figure 6. Left: a slice of the ML-approximated $\mathcal{A}_2(\Phi)$ function. Right: top-quark transverse momentum, comparing standard MINLO ST and STJ results with the ML-improved "STJ*" calculation. STJ* is, as intended, simultaneously consistent with both the ST and STJ processes.

with loss function

$$\mathcal{L} = \sum_{i}^{N_{\mathrm{bins}}} \left[\sum_{j}^{N_{\mathrm{evt}}} w_{i,j}^{\mathrm{ST}} - \sum_{k}^{N'_{\mathrm{evt}}} w_{i,k}^{\mathrm{STJ}} \mathrm{e}^{\tilde{\mathcal{A}}_2(\Phi_i)\mathcal{G}_2} \right] \tag{9}$$

where the i-sum is over discrete bins in the phase-space Φ, the j- and k-sums are over events in ST and STJ event samples, $\tilde{\mathcal{A}}_2(\Phi_i)$ is the trial function, and $\mathcal{G}_2$ is a Φ-independent term from NNLL resummation. A slice of the $\mathcal{A}_2(\Phi)$ function and the resulting improved-MINLO STJ prediction of top-quark p_T are shown in Figure 6. Such uses of new non-parametric methods for improvement of matched NLO calculations are an exciting new direction which we may expect to see more of in future.

6. New computing architectures

Computing architectures have long been of subleading interest to SHG MC developers, since the event sampling is embarrassingly parallel: just use distinct random-number generator seeds for each run, and linear scaling is trivial. But the rise of high-multiplicity matrix element calculations, in particular at NLO, has changed the game: large memory requirements require memory sharing between logical cores, and complex phase-space sampling demands *coordinated* adaptive sampling between many compute nodes. In addition, the availability of architecture developments like vectorization (in CPU, GPU & tensor processing) may also offer benefits that cannot responsibly be ignored but whose exploitation will require significant technical effort.

The first trend being driven by rising computational costs is deconstruction of the neat black-box operation of the C++-era SHG codes [27, 28, 9]. SHERPA in particular is a sophisticated combination of a matrix-element assembler, sampler, matching & merging algorithm, parton

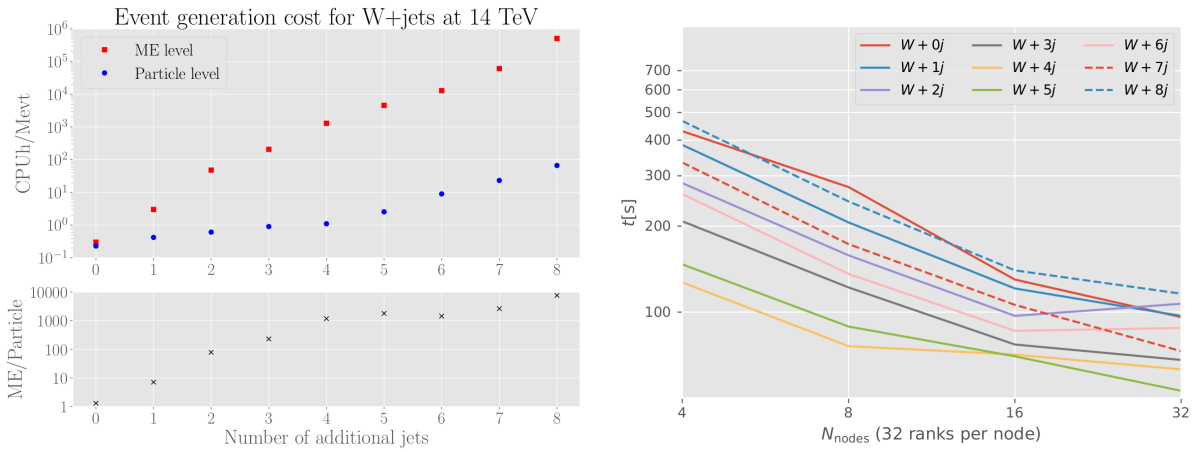

Figure 7. Left: relative scaling of ME-level and Particle-level (rest of SHG) CPU cost with ME sub-process multiplicity: ME is exponentially more expensive at high multiplicity, driving use of HPC for high-multiplicity ME sampling. Right: time-scaling of ME integrations with the degree of MPI parallelism. Plots from Holger Schulz.

showers, electroweak corrections, MPI, and hadron-decay modelling, all in one package and run through one command. The huge cost of ME integration long ago led to a decoupled initial scan mode to map the phase space for efficient use in event generation (the so-called "gridpack" mode, following MadGraph5_aMC@NLO nomenclature), but Sherpa still attempts to handle all n-leg ME generation, merging, matching, showering, etc. inside a single generation run – with the result that it runs as fast as its slowest sub-process. This, in addition to the CPU cost of CKKW clustering discussed earlier, largely explains the high CPU bill of Sherpa-heavy ATLAS as compared to MadGraph5-dominated CMS.

This deficiency, and the increasing availability of high-performance computing (HPC) facilities with multi-node coordination via the message passing interface (MPI), has driven a project to factorise Sherpa's generation strategy so that the number-crunching capacity of the big facilities is used to generate each parton-multiplicity ME event sample separately, for later merging and downstream physics-processing. This has involved developing a new, partonic event format based on the HDF5 technology, which supports parallel read and write, as opposed to the write-locked plain text LHE format which is the current standard [29]. As shown in Figure 7, the CPU scaling of ME generation is exponential in parton multiplicity, while the shower & matching elements scale more respectably: we may hence be entering Run 3 of the LHC with a new hybrid strategy for precision event generation, in which high-multiplicity partonic event samples are generated on HPC resources, and lower-multiplicities and downstream generator components run on WLCG Grid resources as now. The potential is very high: Figure 7 illustrates the HPC scaling of various $W + n$ jets MEs, up to an extraordinary $n = 8$: with this system, generating and analysing 100M events at up to 8-jet order has been performed in 25 minutes!

A final mention of GPU technology is worthwhile, even though this technology has not yet delivered more than prototype demonstrators in event generation. MadGraph5 investigations of LO ME sampling with GPUs began in 2009[3], and all SM processes have been supported in that mode since 2013, but there is little production use so far. It remains to be seen whether GPU or conventional HPC resources offer more potential for LHC simulation campaigns: effort will naturally follow available facilities to some degree. An intriguing angle is the increasing

[3] NLO generation on GPUs is currently unfeasible since the large memory size of NLO matrix elements exceeds that available on GPU devices.

sophistication and "auto-GPUing" of vector processing tasks via machine-learning frameworks such as TensorFlow: these could remove the language barrier that discourages MC authors from focusing on uncertain uptake of GPU processing, by having the code automatically generated. This would also be an appealing route to incorporation of ML-assisted integration strategies. The ML/data-science world being heavily oriented towards Python, it seems that MadGraph5 is particularly well-positioned to explore and benefit from these new technologies.

7. Low-hanging fruit?

Our final duty is to step back briefly from the technicalities, both mathematical and technological, of high-precision MC generation, and consider whether there are simpler strategies that might also achieve LHC aims.

There is an increasing realisation that NLO precision may not always give better accuracy and that sometimes the optimal solution for experimental purposes is both more approximate and much cheaper computationally. To understand this, we must appreciate that SHG MC events are used in two ways by collider experiments:

(i) for tests of state-of-the-art theory against experiment data;

(ii) as the best available emulation of how events are distributed and reconstructed.

The latter is in fact more common: when estimating the backgrounds to new physics searches, or deriving the detector response for use in observable unfolding, a high formal precision gives confidence in a simulated event sample, but we would also be happy with any large, accurate event sample that fell out of the sky, regardless of origin. In practice, even the highest-precision MC often gets re-weighted either differentially or in normalisation to better match data in control regions. This reality in which formal precision is not something essential and sacrosanct – except of course when explicitly testing theory – offers us a strategic viewpoint different to that assumed thus far: can we achieve experimental accuracy via less formally satisfying (and expensive) ansaetz?

Given, in particular, the large per-event cost premium of high-multiplicity NLO over its tree-level LO equivalent, this is not a stupid question: there may be more physics value in using the available CPU for higher event statistics than for higher-n N^{n}LO and N^{n}LL computations. It is also useful to note that the headline achievement of NLO-matched SHG simulation is not actually used for most major SM samples: the total cross-section, so carefully preserved by matching algorithms, is frequently obviated by normalising the SHG sample to a state-of-the-art parton-level calculation at e.g. NNLO+NNLL order. Indeed, as differential calculations at such orders and in directions such as beyond-leading-colour PS become available, such re-weightings are also becoming differential – with great care. Making multidimensional re-weightings of this type efficient is again an area where ML methods may have a role to play: when approximations can add value, ML is our best hope of systematically making better approximations. NLO matching is not rendered pointless by this strategy, since the preservation of NLO-accurate observables[4] gives better stability against theory uncertainties such as renormalisation and factorisation scale variations. But this is a lesser benefit, and perhaps not important for many use-cases: a realistic appreciation of how SHG MC will be *used* at the HL-LHC is crucial in the calculus of estimating physics demands, rather than assuming that "more ns" is always the best strategy.

[4] Note that not all observables from an "NLO"-branded MC generator are NLO-accurate. The classic counterexample is the boson (or leading-jet) p_T in inclusive Higgs or vector boson production: the LO process gives a trivial zero-p_T, and it is this $2 \to 1$ amplitude that is stabilised by the addition of the NLO loop term; at "NLO" the boson p_T becomes non-trivial, but only at LO accuracy due to the unstabilised real-emission term. Further observables such as the second-jet p_T, will only be accurate to LL since they don't even exist in the fixed-order process and are produced by the PS.

Even less technical, but potentially crucial, are "social" initiatives, such as better coordination between experiments. A great deal of MC generation CPU is effectively duplicated, with ATLAS and CMS generating essentially equivalent MC events. Little or no experiment intellectual property resides in these pre-detector MC samples, and so the science case for keeping them private is not strong. The evolution of data archival and distribution machinery through CERN, cf. the EOS, CVMFS, and Zenodo systems [30, 31, 32, 33], means that sharing of events, at least those expensively generated at parton level, is a very low-hanging fruit indeed for improving each experiment's ratio of physics impact to CPU expenditure. While naïve suggestions of a factor of two gain here are overblown (as mentioned, ATLAS and CMS have quite large differences in generator preference), the simple availability of more variety can only be a good thing. It is unlikely that such sharing would significantly reduce any cross-checking via MC variation, since a greater variety of inputs intrinsically reduced biases, and the availability of common pre-detector samples should assist comparison of the experiments' measurements in slightly different phase-spaces. The development of the new HDF5 HPC partonic-event format may end up being a major player in this, along with the required engineering work to factorise previously monolithic generators into re-runnable processing stages. Even such a conceptually easy plan requires real effort to turn it into reality, but efforts coordinated by the HSF suggest that openness and sharing of MC-generated data is now on the agenda to an unprecedented extent.

Lurking in the background is also a big question of incentives and institutional structures, raised in policy statements by the HSF [34] and MCnet [35]. Particle physics internationally has a dichotomy of funding and career streams for "experimental" vs. "theoretical" personnel, and SHG MC development sits uncomfortably on the divide – essential to experiments without itself being experimental physics, but containing a significant amount of engineering and data-comparison work that is not always valued by theory panels. SHG MC authors operate in the theory environment and respond to that community's pressures – publishing of identifiably theoretical papers, and an emphasis on formal innovation over computational performance. This has been characterised [36] as an incentive to make MC codes that run just fast enough to write the theory paper, but no faster! This is all understandable, if not optimal, but means that if LHC experiments want improvements in MC generation performance – as is all but mandated by the HL-LHC physics demands and computing budget – they will need to contribute directly to the technical development and performance optimisation of MC codes. This includes "community codes" like the LHAPDF parton density and the HepMC event record libraries [37, 38], neither of which has dedicated funding: significant CPU is wasted in these "trivial" codes, for lack of developer person-power. Discussions along these lines are in an early stage and need to be taken seriously: to get the best out of our massive experimental facilities, a collegiate and two-way relationship is needed between the experiments and the MC theorists whose work we rely on.

8. Conclusions

This has been a whistle-stop tour through the current state of fully-exclusive MC event generation, primarily from the perspective of current and future LHC experiment simulation campaigns, and their increasingly tight CPU budgets. As we have seen, the main computational costs of MC generation are the efficient sampling and merging of high-multiplicity partonic matrix elements, a problem which becomes exponentially worse when the additional divergence structures of loop-level matrix elements are introduced. Several promising strategies to ameliorate this problem have been shown: use of machine learning both for generic and process-specific improvements to ME samplers, and factorisation of the ME calculation part for efficient use of MPI HPC facilities. In addition we have seen that ML techniques may be used for non-parametric estimation of calculation elements currently beyond *a priori* calculation. It is important also to not lose sight of the physics in such calculations and how it fits into the experiments' usage patterns: more collaboration, both between the experiments and between experiments and MC authors, and

judicious use of "less precise" MC samples have potential to return dividends as significant as the technical acrobatics. As the LHC Run 3 and HL eras approach, we may hope to see more of all these approaches in experiment production campaigns.

Acknowledgements
Many thanks to Stefan Hoeche, Holger Schulz, Keith Hamilton, Marek Schoenherr, Frank Siegert, Josh McFayden, Valentin Hirschi, Andrea Valassi and more, particularly in the MCnet and HSF communities. This work has received funding from the European Union's Horizon 2020 research and innovation programme as part of the Marie Sklodowska-Curie Innovative Training Network MCnetITN3 (grant agreement no. 722104), and the Royal Society via University Research Fellowship grant no UF160548.

References

[1] Andy Buckley et al. General-purpose event generators for LHC physics. *Phys. Rept.*, 504:145–233, 2011.
[2] Torbjorn Sjostrand, Stephen Mrenna, and Peter Z. Skands. PYTHIA 6.4 Physics and Manual. *JHEP*, 05:026, 2006.
[3] G. Corcella, I. G. Knowles, G. Marchesini, S. Moretti, K. Odagiri, P. Richardson, M. H. Seymour, and B. R. Webber. HERWIG 6: An Event generator for hadron emission reactions with interfering gluons (including supersymmetric processes). *JHEP*, 01:010, 2001.
[4] Torbjorn Sjostrand and Maria van Zijl. A Multiple Interaction Model for the Event Structure in Hadron Collisions. *Phys. Rev.*, D36:2019, 1987.
[5] J. M. Butterworth, Jeffrey R. Forshaw, and M. H. Seymour. Multiparton interactions in photoproduction at HERA. *Z. Phys.*, C72:637–646, 1996.
[6] Michelangelo L. Mangano, Mauro Moretti, Fulvio Piccinini, Roberto Pittau, and Antonio D. Polosa. ALPGEN, a generator for hard multiparton processes in hadronic collisions. *JHEP*, 07:001, 2003.
[7] Fabio Maltoni and Tim Stelzer. MadEvent: Automatic event generation with MadGraph. *JHEP*, 02:027, 2003.
[8] F. Krauss, R. Kuhn, and G. Soff. AMEGIC++ 1.0: A Matrix element generator in C++. *JHEP*, 02:044, 2002.
[9] T. Gleisberg, Stefan. Hoeche, F. Krauss, M. Schonherr, S. Schumann, F. Siegert, and J. Winter. Event generation with SHERPA 1.1. *JHEP*, 02:007, 2009.
[10] Stefano Frixione, Paolo Nason, and Bryan R. Webber. Matching NLO QCD and parton showers in heavy flavor production. *JHEP*, 08:007, 2003.
[11] Stefano Frixione, Paolo Nason, and Carlo Oleari. Matching NLO QCD computations with Parton Shower simulations: the POWHEG method. *JHEP*, 11:070, 2007.
[12] Simone Alioli, Paolo Nason, Carlo Oleari, and Emanuele Re. A general framework for implementing NLO calculations in shower Monte Carlo programs: the POWHEG BOX. *JHEP*, 06:043, 2010.
[13] Zoltan Nagy and Davison E. Soper. Matching parton showers to NLO computations. *JHEP*, 10:024, 2005.
[14] Thomas Gehrmann, Stefan Hoche, Frank Krauss, Marek Schonherr, and Frank Siegert. NLO QCD matrix elements + parton showers in e^+e^- —¿ hadrons. *JHEP*, 01:144, 2013.
[15] Rikkert Frederix and Stefano Frixione. Merging meets matching in MC@NLO. *JHEP*, 12:061, 2012.
[16] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H. S. Shao, T. Stelzer, P. Torrielli, and M. Zaro. The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations. *JHEP*, 07:079, 2014.
[17] Johan Alwall, Michel Herquet, Fabio Maltoni, Olivier Mattelaer, and Tim Stelzer. MadGraph 5 : Going Beyond. *JHEP*, 06:128, 2011.
[18] G Peter Lepage. A new algorithm for adaptive multidimensional integration. *Journal of Computational Physics*, 27(2):192 – 203, 1978.
[19] S. Jadach. Foam: A General purpose cellular Monte Carlo event generator. *Comput. Phys. Commun.*, 152:55–100, 2003.
[20] Joshua Bendavid. Efficient Monte Carlo Integration Using Boosted Decision Trees and Generative Deep Neural Networks. 2017.
[21] Johannes Krause. Efficiency Improvements Using Machine Learning in Event Generators for the LHC. 2015. MSc thesis, Technische Universität Dresden.
[22] Felix Bloch and Arnold Nordsieck. Note on the radiation field of the electron. *Physical Review*, 52(2):54, 1937.

[23] DR Yennie, Steven C Frautschi, and H Suura. The infrared divergence phenomena and high-energy processes. *Annals of Physics*, 13(3):379–452, 1961.
[24] Toichiro Kinoshita. Mass singularities of feynman amplitudes. *Journal of Mathematical Physics*, 3(4):650–677, 1962.
[25] Tsung-Dao Lee and Michael Nauenberg. Degenerate systems and mass singularities. *Physical Review*, 133(6B):B1549, 1964.
[26] Stefano Carrazza, Rikkert Frederix, Keith Hamilton, and Giulia Zanderighi. MINLO t-channel single-top plus jet. *JHEP*, 09:108, 2018.
[27] Torbjorn Sjostrand, Stephen Mrenna, and Peter Z. Skands. A Brief Introduction to PYTHIA 8.1. *Comput. Phys. Commun.*, 178:852–867, 2008.
[28] M. Bahr et al. Herwig++ Physics and Manual. *Eur. Phys. J.*, C58:639–707, 2008.
[29] Johan Alwall et al. A Standard format for Les Houches event files. *Comput. Phys. Commun.*, 176:300–304, 2007.
[30] Luca Mascetti, H Gonzalez Labrador, M Lamanna, JT Mościcki, and AJ Peters. Cernbox+ eos: end-user storage for science. In *Journal of Physics: Conference Series*, volume 664, page 062037. IOP Publishing, 2015.
[31] Andreas J Peters and Lukasz Janyst. Exabyte scale storage at cern. In *Journal of Physics: Conference Series*, volume 331, page 052015. IOP Publishing, 2011.
[32] Predrag Buncic, C Aguado Sanchez, Jakob Blomer, Leandro Franco, Artem Harutyunian, Pere Mato, and Yushu Yao. Cernvm–a virtual software appliance for lhc applications. In *Journal of Physics: Conference Series*, volume 219, page 042003. IOP Publishing, 2010.
[33] Megan Potter and Tim Smith. Making code citable with zenodo and github. *Software Sustainibility Institute*, 2015.
[34] Johannes Albrecht et al. A Roadmap for HEP Software and Computing R&D for the 2020s. *Comput. Softw. Big Sci.*, 3(1):7, 2019.
[35] Simone Alioli et al. Monte Carlo event generators for high energy particle physics event simulation. 2019.
[36] M. C. Theorist. Private Communication, 2018.
[37] Andy Buckley, James Ferrando, Stephen Lloyd, Karl Nordström, Ben Page, Martin Rüfenacht, Marek Schönherr, and Graeme Watt. LHAPDF6: parton density access in the LHC precision era. *Eur. Phys. J.*, C75:132, 2015.
[38] Matt Dobbs and Jorgen Beck Hansen. The HepMC C++ Monte Carlo event record for High Energy Physics. *Comput. Phys. Commun.*, 134:41–46, 2001.