Unweighted multijet event generation using factorisation-aware neural networks

Timo Janßen

in collaboration with K. Danziger, D. Maître, S. Schumann, F. Siegert, H. Truong

Institut für Theoretische Physik, Georg-August-Universität Göttingen

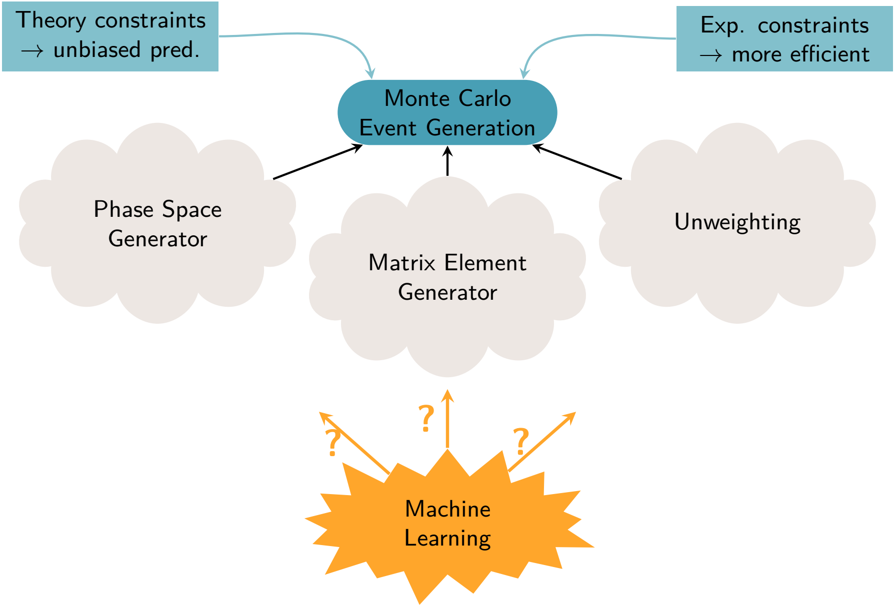
25th MCnet meeting 2023 @ CERN

GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN IN PUBLICA COMMODA SEIT 1737

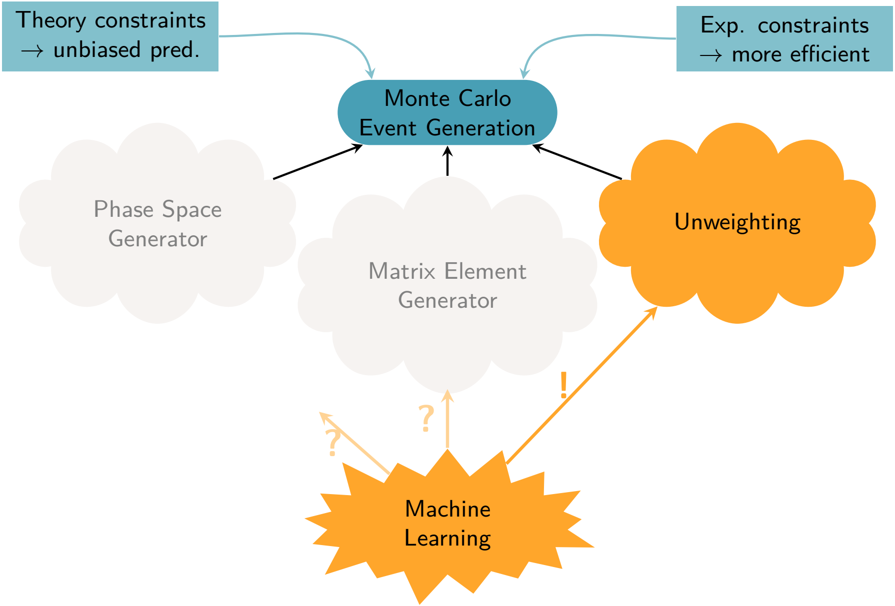
$[q,p]=i\hbar$

GEFÖRDERT VOM
Bundesministerium für Bildung und Forschung

Introduction

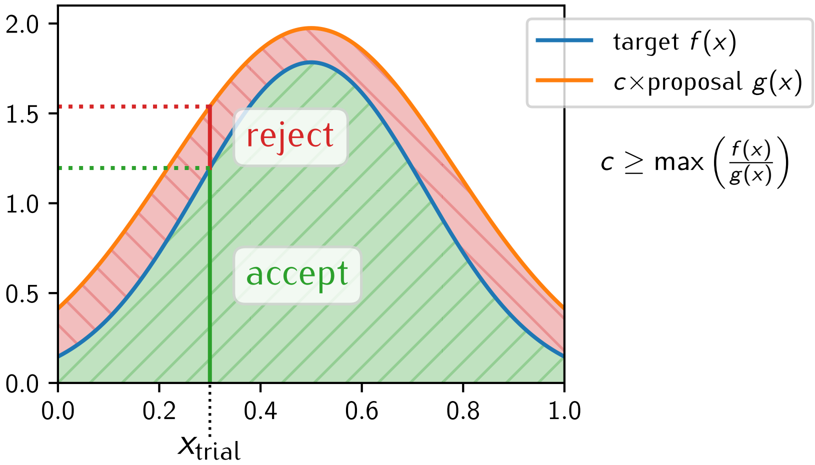


Introduction



How to generate unweighted events

rejection sampling (hit-or-miss):



Unweighting efficiency: $\epsilon = \frac{N_{\text{accepted}}}{N_{\text{trials}}} \approx \frac{1}{c}\left\langle \frac{f}{g} \right\rangle$

Basic Idea

K. Danziger, TJ, S. Schumann, F. Siegert
SciPost Phys. 12, 164 (2022)

- #Feynman diagrams grows quickly with #particles
  $\rightarrow$ high-multiplicity MEs are very expensive
- need to evaluate the ME for each trial event
  $\rightarrow$ small unweighting efficiency = bottleneck

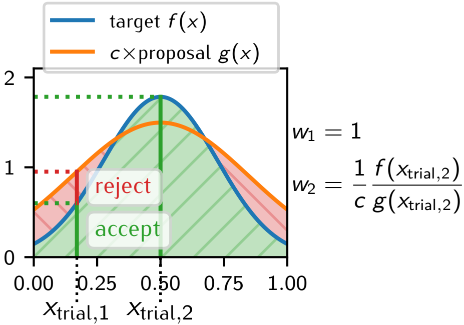
Idea:

- reduce event generation time by reducing the number of calls to the matrix element
  $\rightarrow$ use a fast & accurate surrogate
- correct all errors from the approximation in a 2nd unweighting step
  $\rightarrow$ method is unbiased by design

Interlude: Partial unweighting

- NN are suitable as highly accurate surrogates

  ... but can produce extreme outliers
- large-weight outliers diminish unweighting efficiency even when contribution to total XS is miniscule

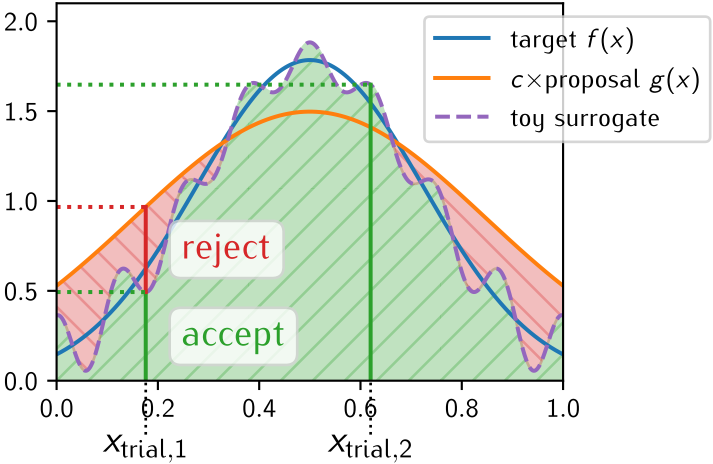


Partial Unweighting

- allow $\max(g) < \max(f)$
- some events get an overweight $\tilde{w} > 1$
- partial unweighting is the default in SHERPA (and other generators)
  - we don't know the global maximum
  - partial unweighting is much faster

Surrogate unweighting

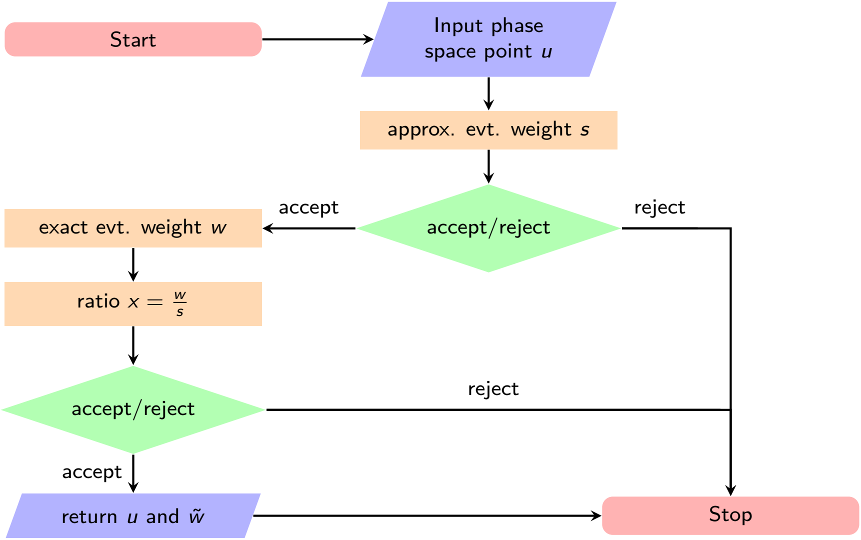
K. Danziger, TJ, S. Schumann, F. Siegert
SciPost Phys. 12, 164 (2022)



- surrogate should be fast and accurate
- have to correct for wrong accept/reject probabilities
  $\rightarrow$ 2nd unweighting against true target for all accepted points

Surrogate unweighting algorithm

K. Danziger, TJ, S. Schumann, F. Siegert
SciPost Phys. 12, 164 (2022)



Matrix element emulation

- gradient boosting machines for loop-induced amplitudes [F. Bishara, M. Montull: arXiv:1912.11055]
- NN for $e^+e^- \to$ jets [S. Badger, J. Bullock: JHEP 06 (2020) 114]
- NN for loop-induced amplitudes [J. Aylett-Bullock, S. Badger, R. Moodie: JHEP 08 (2021) 066]
- dipole model for $e^+e^- \to$ jets [D. H. Truong: JHEP 11 (2021) 066]
- learn ME$\times$PS for surrogate unweighting [K. Danziger, TJ, S. Schumann, F. Siegert: SciPost Phys. 12, 164 (2022)]
- Bayesian networks for loop amplitudes [S. Badger, A. Butter, M. Luchmann, S. Pitz, T. Plehn: arXiv:2206.14831]

Factorisation-aware matrix element emulation

D. Maître, H. Truong
JHEP 11 (2021) 066

soft/collinear factorisation properties

$$|\mathcal{M}_{n+1}|^2 \to |\mathcal{M}_n|^2 \otimes \mathbf{V}_{ijk}$$

[Catani, Seymour Nucl.Phys. B485 (1997) 291-419]

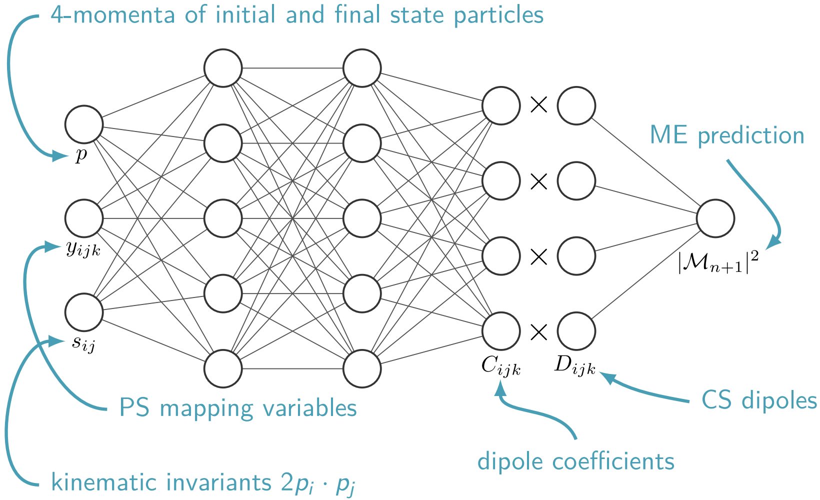
Ansatz

$$\langle|\mathcal{M}|^2\rangle = \sum_{\{ijk\}} C_{ijk} D_{ijk}$$

- $D_{ijk} = \langle V_{ijk} \rangle / s_{ij}$: spin-averaged Catani-Seymour dipoles divided by kinematic invariant
- C_{ijk}: coefficients fit by neural network

Factorisation-aware matrix element emulation

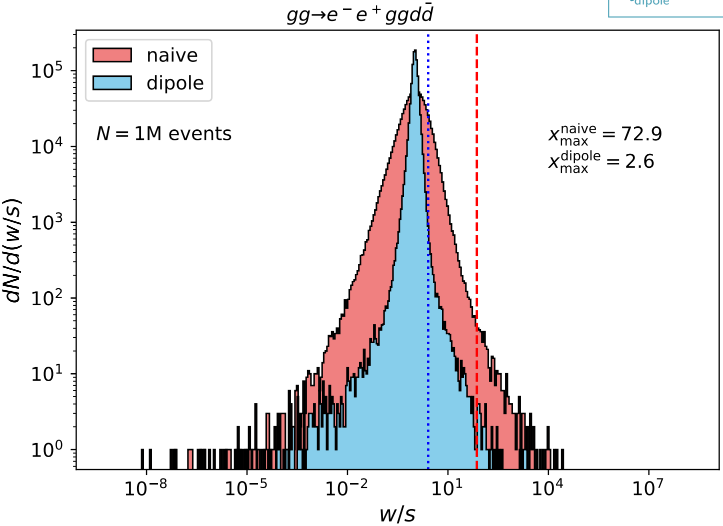
D. Maître, H. Truong
JHEP 11 (2021) 066



Factorisation-aware matrix element emulation

Comparison with naive (non-dipole) model for $Z + 4j$:

Comparison of eval time:
$\frac{t_{\text{AMEGIC}}}{t_{\text{dipole}}} \approx 388$



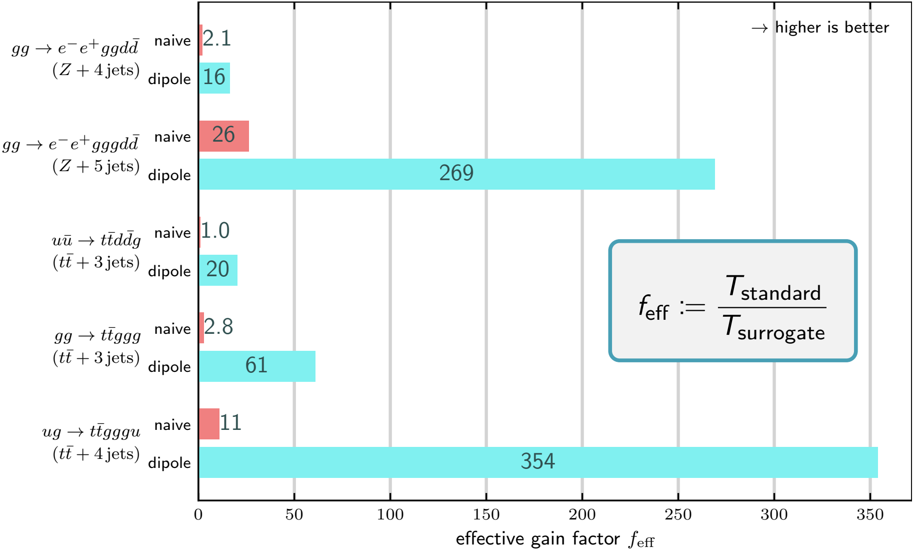
Implementation details

- constraint from experiment simulation workflow: CPU single threaded

  $\rightarrow$ no benefit from NN vectorisation capabilities
- for NN evaluation use ONNX Runtime with all possible optimisations
- two step unweighting implemented in SHERPA [Gleisberg et al. JHEP02(2009)007, Bothmann et al. SciPost Phys. 7, 034 (2019)]
- ME generator: AMEGIC [Krauss et al. JHEP 02 (2002) 044]
- we evaluate the performance for processes that are very important for the LHC: V+jets & $t\bar{t}$+jets

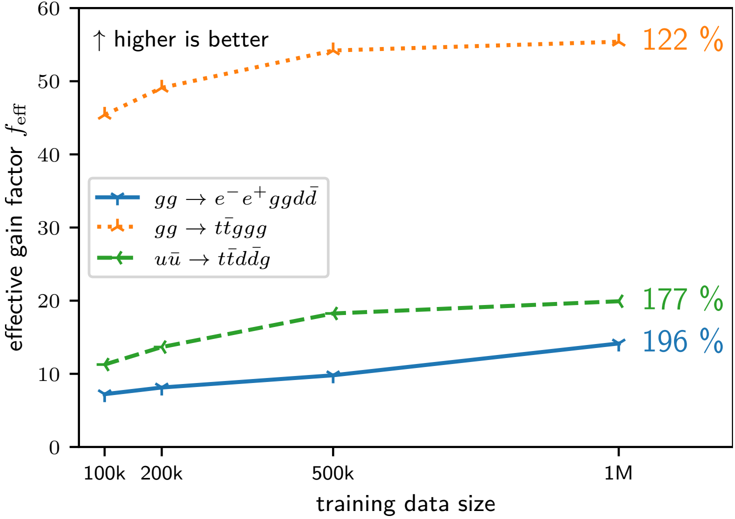
Results: effective gain factors for LHC multi-jet processes

Using 1M training events:



TJ, D. Maître, S. Schumann, F. Siegert, H. Truong: arXiv:2301.13562

Results: effect of training size variation



Colour sampling

- realistic use case: multi-jet merged calculations @LHC
- most promising part: highest multiplicity LO amplitudes
- at high multiplicity we prefer colour-sampling → see Max's talk

naive ansatz

- use the same (colour-summed) dipole model and augment it with colour assignments
- let the NN figure out the rest
- difficulty: with Comix [Gleisberg & Hoeche JHEP12 (2008) 039]: $T(w_{\mathrm{PS}}) \approx \mathrm{T}(\mathrm{w}_{\mathrm{ME}})$

  → train on full event weight ($w_{\mathrm{ME}} \times w_{\mathrm{PS}}$)

Result:

→ significant drop in performance, no gains

→ further work necessary

Summary

- generic method to speed up unweighting with surrogates
- premises: costly integrand & low unweighting efficiency
- dipole model very accurate for colour-summed MEs

  $\rightarrow$ incl. hadronic initial states & massive quarks
- large gain factors for unweighting of colour-summed MEs

  $\rightarrow$ can enable colour-summing for higher multiplicities

Outlook

- improve gains for **colour-sampled** MEs by using better suited models
- use dipole model for **other applications**
- emulation of **loop amplitudes**

Summary

- generic method to speed up unweighting with surrogates
- premises: costly integrand & low unweighting efficiency
- dipole model very accurate for colour-summed MEs

  $\rightarrow$ incl. hadronic initial states & massive quarks
- large gain factors for unweighting of colour-summed MEs

  $\rightarrow$ can enable colour-summing for higher multiplicities

Outlook

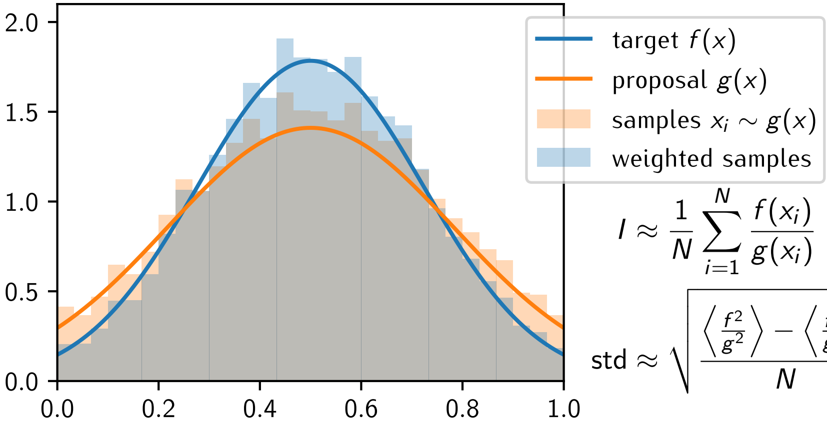
- improve gains for **colour-sampled** MEs by using better suited models
- use dipole model for **other applications**
- emulation of **loop amplitudes**

Questions?

Backup

How to generate weighted events

importance sampling:



$$I \approx \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{g(x_i)}$$

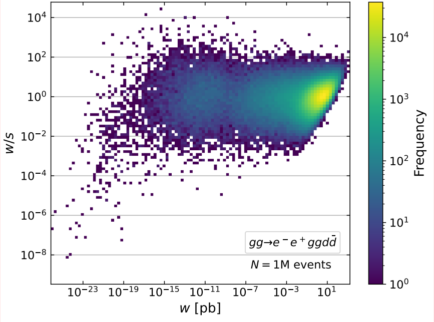
$$\text{std} \approx \sqrt{\frac{\left\langle \frac{f^2}{g^2} \right\rangle - \left\langle \frac{f}{g} \right\rangle^2}{N}}$$

HEP example: Breit-Wigner distribution for resonances

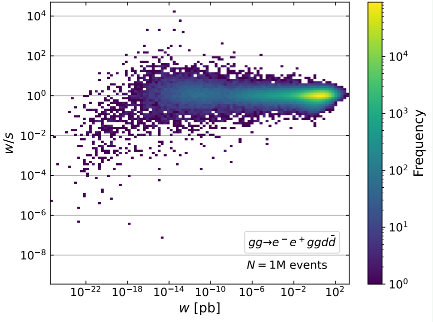
Factorisation-aware matrix element emulation

Comparison with naive (non-dipole) model:

naive

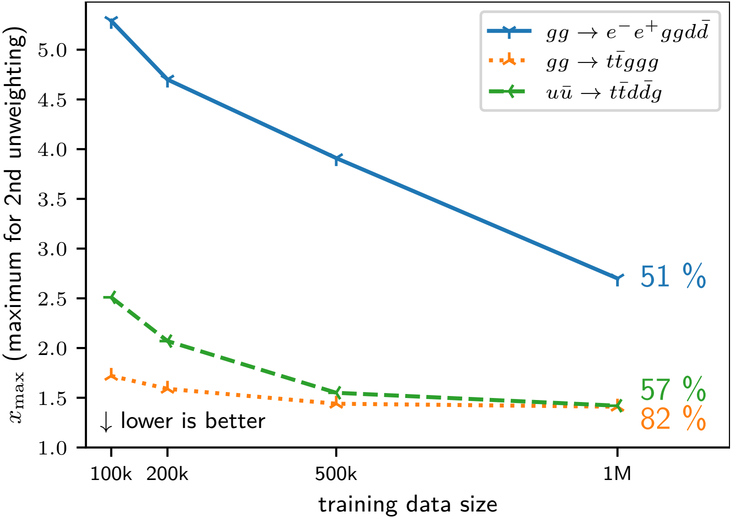


dipole



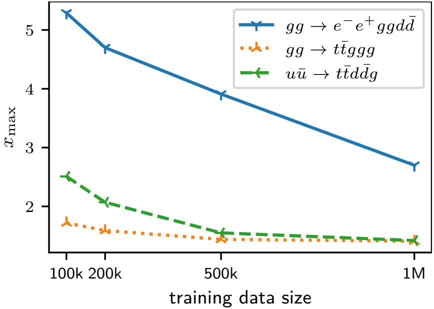
Factorisation-aware matrix element emulation

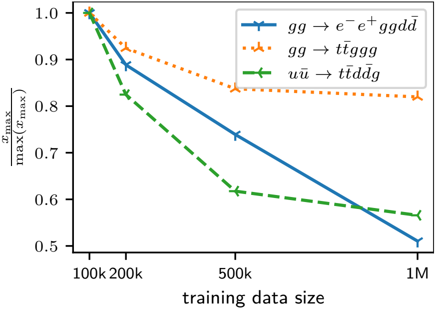
Effect of training size variation:



Factorisation-aware matrix element emulation

Effect of training size variation:





Results: effect of training size variation