Available on CMS information server

CMS CR -2025/041

The Compact Muon Solenoid Experiment

Conference Report

Mailing address: CMS CERN, CH-1211 GENEVA 23, Switzerland

14 February 2025 (v4, 28 February 2025)

Refining Jets for CMS Run 3 using Fast Simulation

Dorukhan Boncukcu, Acelya Deniz Gungordu, Samuel Bein, Kevin Pedro, Moritz Wolf, M. Ekim Vural, Altan Cakir on behalf of the CMS Collaboration

Abstract

As the LHC moves into its high-luminosity phase, the CMS experiment must handle more complex data collected at much higher rates. While the Geant4-based simulation application (FullSim) provides highly accurate simulation to complement real data, FullSim's intensive consumption of computing resources becomes an increasing liability as the rates increase, while faster tools offer an advantage. The fast MC production application (FastSim) delivers a complete simulation with a factor of 10 speedup over FullSim, but introduces inaccuracies in some observables. A specialized refinement method, Fast Perfekt, employs machine learning to improve the accuracy of FastSim. An initial report of this work focused on the refinement of jet flavor tagging observables. This article presents an update on the refinement, focusing on PUPPI jets with Run 3 data-taking conditions. Refinement is extended to include jet transverse momentum as well as its propagation to missing transverse momentum. A grid-based framework and real-time monitoring system have been developed to facilitate optimization and scaling of the refinement to a large number of target variables.

Presented at *CHEP2024 27th International Conference on Computing in High Energy and Nuclear Physics*

Refining Jets for CMS Run 3 using Fast Simulation

Açelya Deniz Güngördü[1,*], *Dorukhan* Boncukçu[1,**], *Kevin* Pedro[2,***], *Samuel* Bein[3,****], *Moritz* Wolf[4], *Altan* Çakır[1], and *Mirac Ekim* Vural[1] on behalf of the CMS Collaboration

[1]Istanbul Technical University, Science and Letters Faculty, Istanbul, Turkiye
[2]Fermi National Accelerator Laboratory, Batavia, IL, USA
[3]Université Catholique de Louvain, Louvain-la-Neuve, Belgium
[4]University of Hamburg, Institute of Experimental Physics, Hamburg, Germany

Abstract. As the LHC moves into its high-luminosity phase, the CMS experiment must handle more complex data collected at much higher rates. While the Geant4-based simulation application (FullSim) provides highly accurate simulation to complement real data, FullSim's intensive consumption of computing resources becomes an increasing liability as the rates increase, while faster tools offer an advantage. The fast MC production application (FastSim) delivers a complete simulation with a factor of 10 speedup over FullSim, but introduces inaccuracies in some observables. A specialized refinement method, Fast Perfekt, employs machine learning to improve the accuracy of FastSim. An initial report of this work focused on the refinement of jet flavor tagging observables. This article presents an update on the refinement, focusing on PUPPI jets with Run 3 data-taking conditions. Refinement is extended to include jet transverse momentum as well as its propagation to missing transverse momentum. A grid-based framework and real-time monitoring system have been developed to facilitate optimization and scaling of the refinement to a large number of target variables.

1 Introduction

The CMS experiment [1] at the CERN LHC is preparing to record an order of magnitude more data after introducing new highly complex detectors [2]. In the upcoming High-Luminosity LHC (HL-LHC) runs, efficiently processing and analyzing this large quantity of data will be essential for maintaining the experiment's scientific output and ensuring quality results [3], as will producing sufficiently large and accurate simulated data sets to complement the observed data.

The physics analysis pipeline relies on simulation. To address the increasing computational demands, the CMS experiment employs two distinct simulation chains, called FullSim and FastSim. FullSim [4, 5], based on the GEANT4 toolkit [6, 7], simulates the detector response to collision events at state-of-the-art accuracy, based on realistic transport of particles traversing and interacting with the detector. Although highly accurate, FullSim also has high

*e-mail: acelya.deniz.gungordu@cern.ch
**e-mail: dorukhan.boncukcu@cern.ch
***e-mail: pedrok@cern.ch
****e-mail: samuel.bein@cern.ch

computational intensity, which results in slower processing, posing a challenge for scalability in the high-luminosity era. In contrast, FastSim [8–10] implements a set of approximations that significantly reduce computation time, achieving speeds approximately ten times faster than FullSim. FastSim brings a significant advantage for statistically limited analyses. However, the acceleration comes at the cost of reduced accuracy in certain observables.

To increase FastSim's accuracy while maintaining its edge in throughput, a procedure for refining FastSim has been developed based on machine learning (ML): a regression neural network (NN) is trained to refine the FastSim outputs to be more similar to FullSim. A prototype, based on the Fast Perfekt method [11], was recently introduced [12], focusing on jet flavor observables. This proceeding presents an update on the effort to refine FastSim, extending to the jet momentum and its impact on the missing transverse momentum $p_{\mathrm{T}}^{\mathrm{miss}}$. This strategy aims to ensure that the CMS collaboration can effectively navigate the challenges of the HL-LHC.

2 Methodology

Following the methodology of Ref. [12], the training focuses on the simulation of gluino pair production in the framework of the simplified model T1tttt [13] generated using the PYTHIA 8.309 event generator [14] at leading order precision. Each gluino decays into a pair of top quarks and a neutralino, yielding a diverse range of final state objects, including leptons and jets of all flavors over a wide energy range, giving broad support for the phase space of most applications.

Identical sets of generated events are processed with FullSim and FastSim separately, and each generated jet is matched to its reconstructed FullSim and FastSim counterpart, resulting in "jet triplets". A summary of the jet reconstruction and selection is given in Table 1.

Table 1: Jet reconstruction and selection parameters for the training dataset.

Parameter	**Value**
Jet algorithm	anti-k_{T} PUPPI jets [15–18]
Distance parameter (R)	0.4
Jet/jet matching (ΔR)	< 0.2
Overlap removal (ΔR)	< 0.5
p_{T}	$> 15\,\mathrm{GeV}$
$\lvert\eta\rvert$	< 5.0

To preserve the association between reconstructed and generator-level jets, the generated jets, FullSim jets, and FastSim jets are required to be separated from neighboring jets (in the same collection) by $\Delta R > 0.5$. The resulting training sample comprises approximately 6×10^6 jet triplets.

2.1 Network training

The task of the NN is to modify the FastSim jet properties in order to render them as similar as possible to their more accurate FullSim counterparts. The NN takes as input $[\vec{x}_{\mathrm{Fast}}, \vec{g}]$ and outputs a refined vector $\vec{x}_{\mathrm{Refined\,Fast}}$; the concatenation of this output and the generator vector $[\vec{x}_{\mathrm{Refined\,Fast}}, \vec{g}]$ is compared to the target FullSim vector ($[\vec{x}_{\mathrm{Full}}, \vec{g}]$). As shown in Table 2, the process uses four key jet flavor tagging observables derived from the DeepJet algorithm [19]: B for generic b-tagging, and CvB, CvL, and QvG for discrimination of charm vs. bottom, charm vs. light, and quark vs. gluon, respectively. The refinement is guided by auxiliary parameters ($\vec{g}$) including generator-level p_{T}, η, and true hadron flavor information.

Parameter	Definition
Input: $\vec{x}^{T}_{\mathrm{Fast}}$	$\left(p_{\mathrm{T,Fast}}, \mathrm{B}_{\mathrm{Fast}}, \mathrm{CvB}_{\mathrm{Fast}}, \mathrm{CvL}_{\mathrm{Fast}}, \mathrm{QvG}_{\mathrm{Fast}}\right)^{T}$
Auxiliary Parameters ($\vec{g}$)	$p_{\mathrm{T,GEN}}, \eta_{\mathrm{GEN}}$, true hadron flavor (b, c, or uds/g)
Target: $\vec{x}^{T}_{\mathrm{Full}}$	$\left(p_{\mathrm{T,Full}}, \mathrm{B}_{\mathrm{Full}}, \mathrm{CvB}_{\mathrm{Full}}, \mathrm{CvL}_{\mathrm{Full}}, \mathrm{QvG}_{\mathrm{Full}}\right)^{T}$

Table 2: The FastSim input features, FullSim output targets, and auxiliary parameters.

2.2 Network architecture

The refinement network is a ResNet-like NN [20] that estimates the residual differences between FastSim and FullSim, and adds these residuals to the FastSim. The architecture comprises three main components, as illustrated in Fig. 1: a pre-processing layer, a series of four residual blocks, and a post-processing layer, forming an end-to-end refinement pipeline.

The input layer accepts the FastSim feature vector ($\vec{x}_{\mathrm{Fast}}$) and auxiliary parameters ($\vec{g}$), as depicted on the left side of Fig. 1. These inputs undergo pre-processing to normalize and prepare the data for the network's core processing stages. The backbone of the architecture consists of four consecutive residual blocks, each incorporating skip connections to maintain gradient flow and prevent information loss during training.

A single loss function, the maximum mean discrepancy (MMD), is minimized with respect to the network parameters, comparing ensembles of jets to accommodate independent stochasticity in both simulation chains. This is computed over batch sizes of 2048. This loss is combined with an additional constraint that maintains unitarity among the flavor observables using the modified differential method of multipliers.

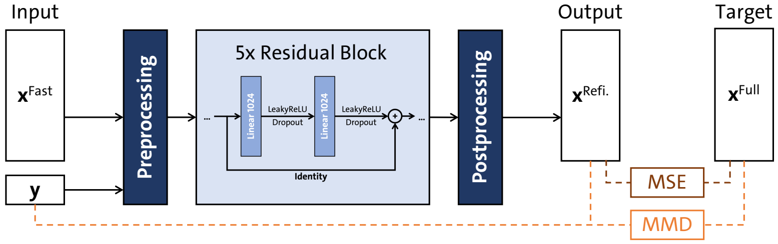

Figure 1: The architecture of the refinement network, taken from [11]. The diagram shows the complete processing pipeline from FastSim inputs through residual blocks to refined outputs, including the dual loss function implementation. In the presented application, the MSE is not used.

3 Results for jet flavor and momentum

Simultaneous refinement of jet flavor and jet p_{T} has been performed and the results are shown in Fig. 2 (upper left). The refined FastSim exhibits clear improvement in modeling. Good refined modeling of jet p_{T} is seen throughout the range spanning 30 GeV to 1 TeV, with a marginal improvement seen outside these ranges. Where FastSim's overly broad jet energy resolution shifts the p_{T} spectrum to higher values, the refinement brings the spectrum into alignment with FullSim. The results of the b-tagging discriminant are shown in Fig. 2 (upper right), where pronounced discrepancies in FastSim's modeling are mostly mitigated by the refinement. The FastSim test entry represents an independent sample of refined FatsSim jets, included as a check of overtraining. The impact of the refinement on correlations among the refined and auxiliary variables are studied in terms of Pearson coefficients. These coefficients are displayed in Fig. 2 (lower), along with residuals of the correlations with respect to those present in FullSim. Correlations among the refined FastSim variables are much closer to those of FullSim, with the largest improvement observed the correlation between p_{T} and the quark-gluon discriminator. A small amount of mismodeling remains after refinement, but the Fast Perfekt approach provides sufficient accuracy for typical analyses.

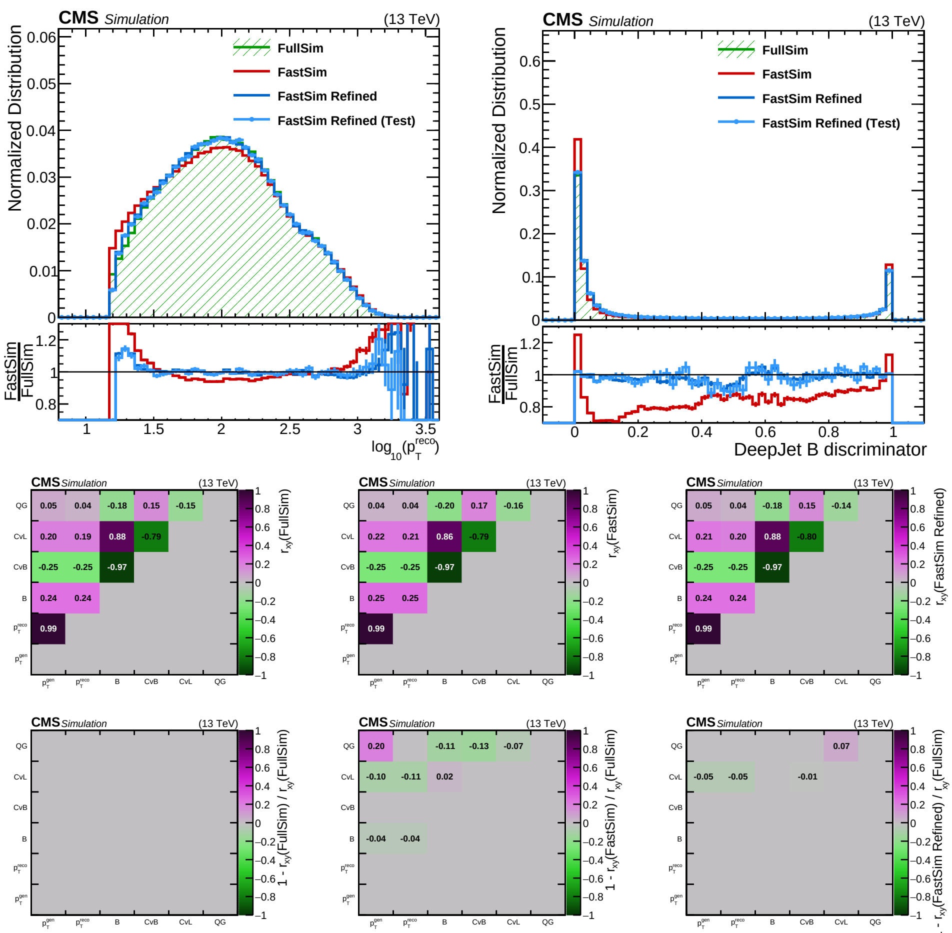

Figure 2: Upper left: Comparison of refined and unrefined jet p_{T} to FullSim. Upper right: Comparison of refined and unrefined DeepJet b discriminator to FullSim. Lower 6 plots: Correlation matrices for FullSim (left), FastSim (center), and refined FasSim (right), with the corresponding residual differences to FullSim shown below.

4 Propagation of corrections to missing momentum

The impact of refinement on the missing transverse momentum is also studied. The so-called hard $p_{\mathrm{T}}^{\mathrm{miss}}$ is used, defined as the magnitude of the negative vector sum of the p_{T} of jets within an event that satisfy $p_{\mathrm{T}} \geq 30\,\mathrm{GeV}$ and $|\eta| \leq 5$.

The effect of refinement on the jet p_{T} is propagated as a correction to the $p_{\mathrm{T}}^{\mathrm{miss}}$ [21]. This correction amounts to replacing the contribution of each jet's momentum to the $p_{\mathrm{T}}^{\mathrm{miss}}$ with that of the refined jet.

The result of these corrections is shown in Fig. 3. The distribution of the refined FastSim hard $p_{\mathrm{T}}^{\mathrm{miss}}$ aligns closely with that of FullSim, while the original FastSim deviates slightly in the lowest and highest ranges. The event-by-event difference between the FastSim and FullSim hard $p_{\mathrm{T}}^{\mathrm{miss}}$ before and after refinement, $\Delta \mathrm{Hard}\, p_{\mathrm{T}}^{\mathrm{miss}}$ is shown for all events, as well as its profile in bins of FullSim hard $p_{\mathrm{T}}^{\mathrm{miss}}$. The differences corresponding to refined FastSim are narrower and more centered around zero than for the unrefined FastSim, indicating an event-by-event improvement in FastSim modeling. A comparison of per-jet p_T differences to FullSim between FatSim and refined FastSim is also shown, indicating improved modeling of FastSim after refinement, particularly on the lower tail of the jet response.

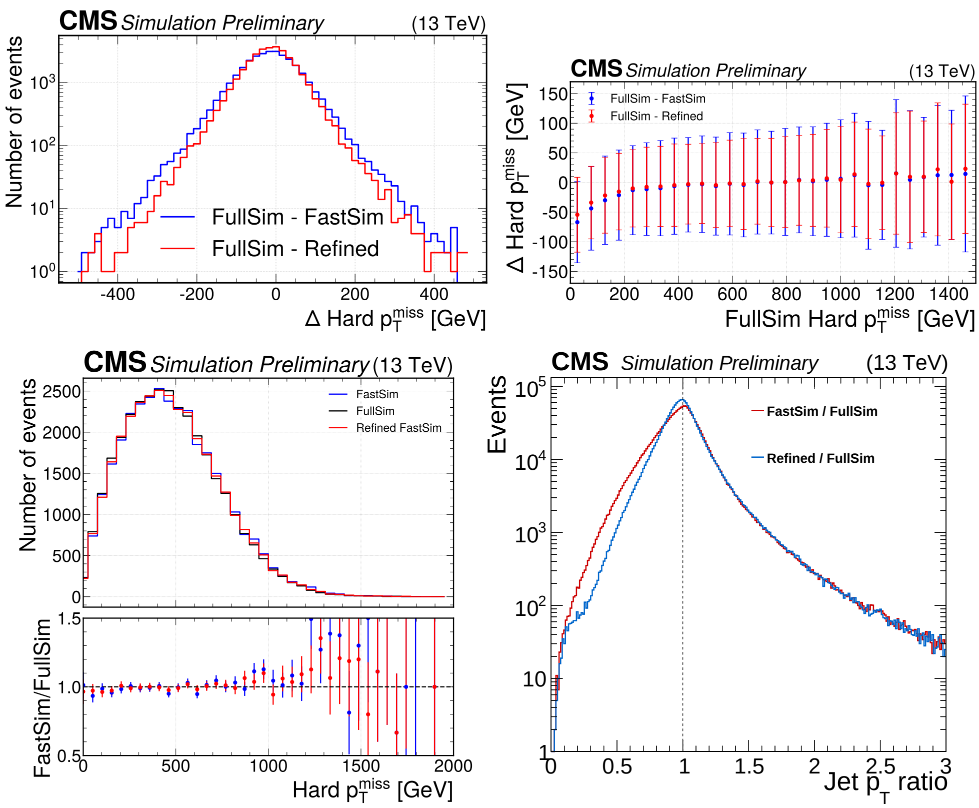

Figure 3: Upper: per-event difference in $p_{\mathrm{T}}^{\mathrm{miss}}$ between FastSim and FullSim, inclusive (left) and profiled in bins of $p_{\mathrm{T}}^{\mathrm{miss}}$ (right). Lower left: Distribution of $p_{\mathrm{T}}^{\mathrm{miss}}$ for FullSim, FastSim, and refined FastSim and the ratio of each FastSim version to FullSim. Lower right: per-jet difference in p_T between FastSim and FullSim jets, as well as those between refined FastSim and FullSim jets.

5 Training and monitoring framework

A framework for launching and monitoring the NN training has been developed. This framework facilitates scaling the refinement to more complex networks and larger numbers of input variables. A schematic of the monitoring system is shown in Fig. 4. The tabulation of the grid search results is illustrated in Fig. 4 (upper), where the progress is viewable as colored bars, and real-time information about the training status is provided through links. Models trained in the grid search are shown under a "Grouped Trainings" tab, where the training variations are organized based on the input variables and the object class. A comprehensive summary table, shown in Fig. 4 (middle), highlights the best-performing models based on predefined metrics, alongside detailed loss values for each model. Finally, real-time loss values of the training are monitored in Fig. 4 (lower).

This framework addresses the growing number of otherwise unconstrained choices in the network and training steps, such as input variable transformation functions, the depth and complexity of the network, and the learning rate.

6 Conclusion

An update to a machine learning-based approach to refine reconstructed jets from Fast Simulation in the CMS experiment is presented. A ResNet-like network, comprising preprocessing layers, a series of residual blocks, and a post-processing layer, estimates a residual correction to augment the FastSim output variables to render them more like FulLSim. The jet transverse momentum p_{T} has been incorporated in addition to jet flavor tagging discriminators, which were introduced in the first iteration of the study. The jet p_{T} correction substantially improves the modeling of the jet p_{T} and also improves the modeling of $p_{\mathrm{T}}^{\mathrm{miss}}$ when propagated to that variable. A grid search mechanism for hyperparameter tuning has been developed, along with a submission and monitoring framework that tracks training progress in real time, facilitating the scaling of the refinement methods to a large number of input parameters. Given the computational challenges posed by the upcoming High Luminosity LHC era, this work addresses the need for efficient yet accurate simulation methods.

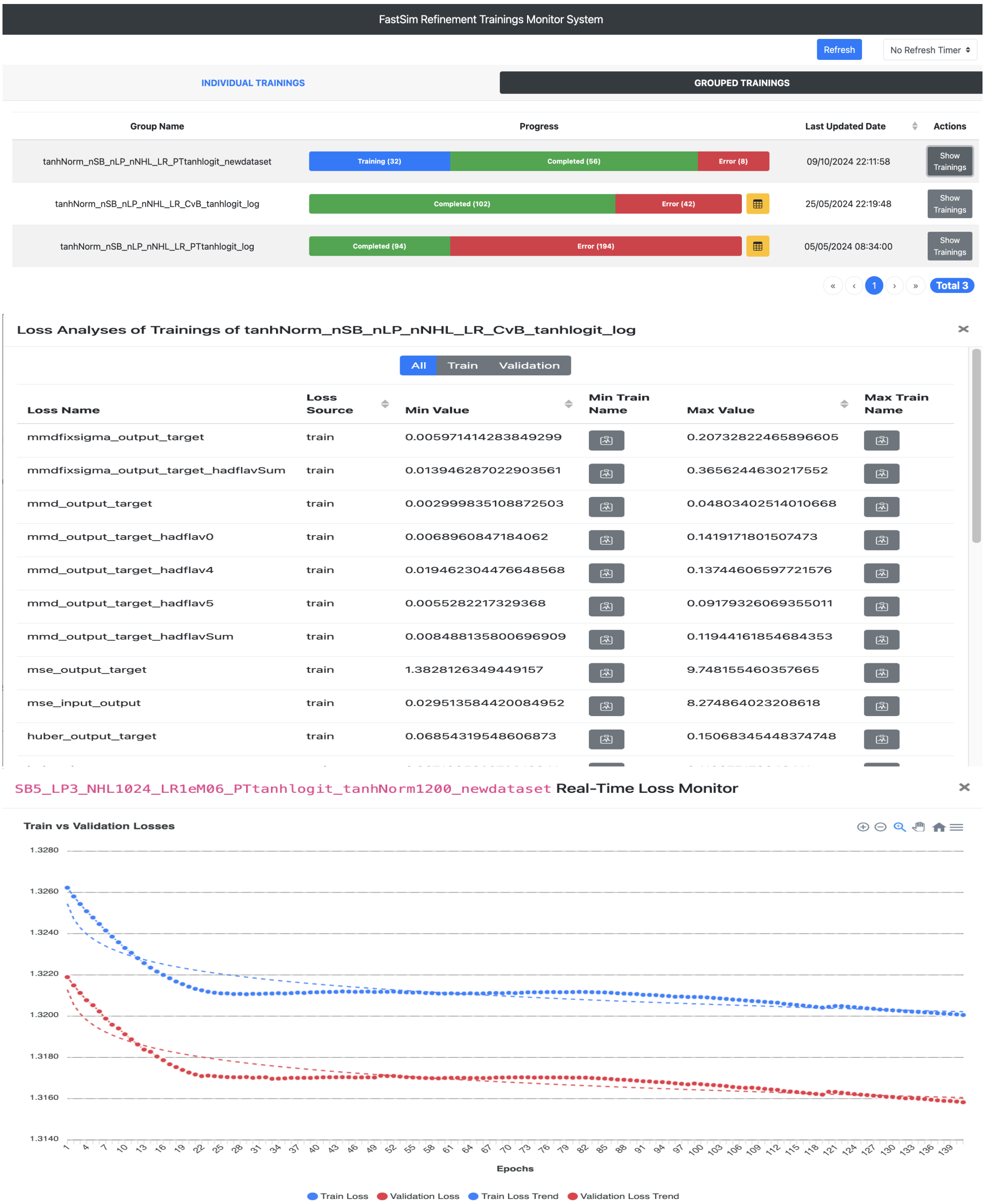

Figure 4: Various features of the training monitoring system: the grid search tabulation (upper), the summary table (middle), and the real-time loss monitoring (lower).

References

[1] CMS Collaboration, The CMS experiment at the CERN LHC, JINST **3**, S08004 (2008). 10.1088/1748-0221/3/08/S08004

[2] CMS Collaboration, CMS Technical Design Report CMS-TDR-014, CERN (2017), `https://cds.cern.ch/record/2272264`

[3] CMS Offline Software and Computing, CMS Note CMS-NOTE-2022-008, CERN (2022), `https://cds.cern.ch/record/2815292`

[4] D.J. Lange, M. Hildreth, V.N. Ivantchenko, I. Osborne (CMS), Upgrades for the CMS simulation, J. Phys. Conf. Ser. **608**, 012056 (2015). 10.1088/1742-6596/608/1/012056

[5] M. Hildreth, V.N. Ivanchenko, D.J. Lange (CMS), Upgrades for the CMS simulation, J. Phys. Conf. Ser. **898**, 042040 (2017). 10.1088/1742-6596/898/4/042040

[6] S. Agostinelli et al., Geant4—a simulation toolkit, Nucl. Instrum. Meth. A **506**, 250 (2003). 10.1016/S0168-9002(03)01368-8

[7] J. Allison et al., Recent developments in Geant4, Nucl. Instrum. Meth. A **835**, 186 (2016). 10.1016/j.nima.2016.06.125

[8] S. Abdullin, P. Azzi, F. Beaudette, P. Janot, A. Perrotta, The fast simulation of the CMS detector at LHC, J. Phys. Conf. Ser. **331**, 032049 (2011). 10.1088/1742-6596/331/3/032049

[9] A. Giammanco, The fast simulation of the CMS experiment, J. Phys. Conf. Ser. **513**, 022012 (2014). 10.1088/1742-6596/513/2/022012

[10] S. Sekmen (CMS), Recent developments in CMS fast simulation, PoS **ICHEP2016**, 181 (2016), `1701.03850`. 10.22323/1.282.0181

[11] M. Wolf, L.O. Stietz, P.L.S. Connor, P. Schleper, S. Bein, Fast Perfekt: Regression-based refinement of fast simulation, SciPost Phys. Core **8**, 021 (2025), `2410.15992`. 10.21468/SciPostPhysCore.8.1.021

[12] S. Bein, P. Connor, K. Pedro, P. Schleper, M. Wolf (CMS), Refining fast simulation using machine learning, EPJ Web Conf. **295**, 09032 (2024), `2309.12919`. 10.1051/epjconf/202429509032

[13] D. Alves (LHC New Physics Working Group), Simplified Models for LHC New Physics Searches, J. Phys. G **39**, 105005 (2012), `1105.2838`. 10.1088/0954-3899/39/10/105005

[14] C. Bierlich et al., A comprehensive guide to the physics and usage of PYTHIA 8.3, SciPost Phys. Codeb. **2022**, 8 (2022), `2203.11601`. 10.21468/SciPostPhysCodeb.8

[15] M. Cacciari, G.P. Salam, G. Soyez, The anti-k_T jet clustering algorithm, JHEP **04**, 063 (2008), `0802.1189`. 10.1088/1126-6708/2008/04/063

[16] M. Cacciari, G.P. Salam, G. Soyez, FastJet user manual, Eur. Phys. J. C **72**, 1896 (2012), `1111.6097`. 10.1140/epjc/s10052-012-1896-2

[17] D. Bertolini, P. Harris, M. Low, N. Tran, Pileup per particle identification, JHEP **10**, 059 (2014), `1407.6013`. 10.1007/JHEP10(2014)059

[18] A.M. Sirunyan et al. (CMS), Pileup mitigation at CMS in 13 TeV data, JINST **15**, P09018 (2020), `2003.00503`. 10.1088/1748-0221/15/09/p09018

[19] E. Bols, J. Kieseler, M. Verzetti, M. Stoye, A. Stakia, Jet flavour classification using deepjet, JINST **15**, P12012 (2020). 10.1088/1748-0221/15/12/P12012

[20] K. He, X. Zhang, S. Ren, J. Sun, Deep Residual Learning for Image Recognition, in *IEEE CVPR* (2016), p. 770

[21] A.M. Sirunyan et al. (CMS), Performance of missing transverse momentum reconstruction in proton-proton collisions at $\sqrt{s}$ = 13 TeV using the CMS detector, JINST **14**, P07004 (2019), `1903.06078`. 10.1088/1748-0221/14/07/P07004