FlashSim

End-to-end machine-learning simulation of CMS events at the analysis level

Prabhat Solanki (Università di Pisa and INFN Pisa), on behalf of the CMS Collaboration

ML4Jets 2026 · University of Vienna · 15 September 2026

1

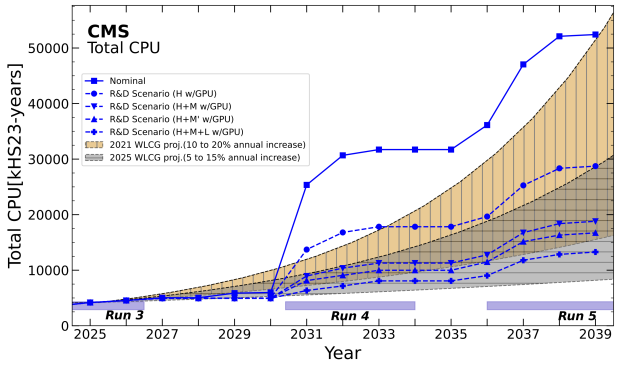
Introduction

Why CMS needs faster simulation, and where FlashSim stands next to FullSim and FastSim.

COMPUTING BUDGET · FULL, FAST AND FLASH

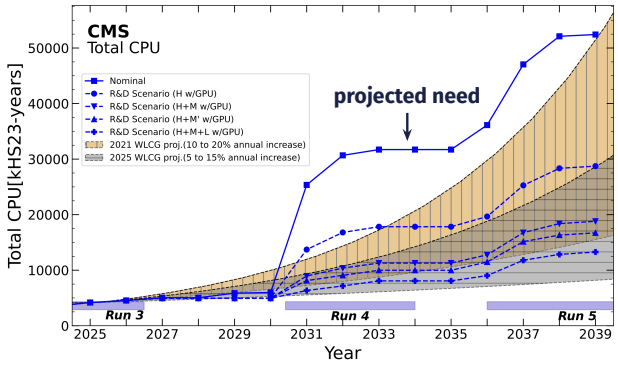
Why faster simulation?

- The CPU that CMS expects to need through the HL-LHC, compared with a flat budget.



Why faster simulation?

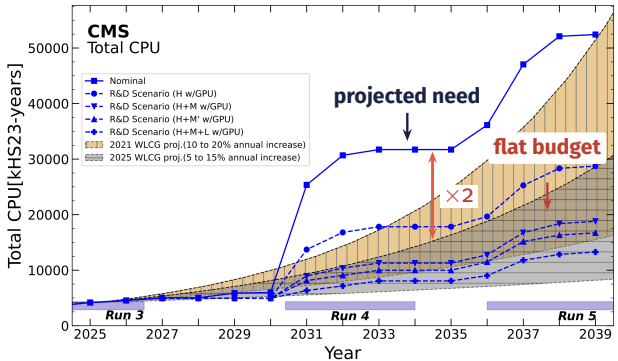
The CPU that CMS expects to need through the HL-LHC, compared with a flat budget.



Run 4 multiplies the need.
With HL-LHC pileup and trigger rates, the projected CPU need grows about five-fold from 2030 to 2033.

Why faster simulation?

The CPU that CMS expects to need through the HL-LHC, compared with a flat budget.

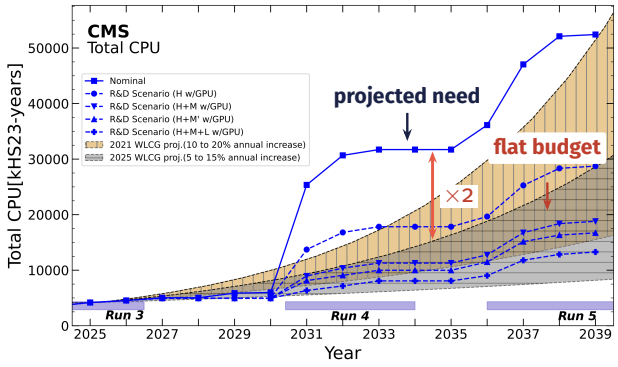


Run 4 multiplies the need.
With HL-LHC pileup and trigger rates, the projected CPU need grows about five-fold from 2030 to 2033.

Hardware alone will not close the gap.
Buying more CPU at a flat budget closes the gap only slowly, so the software itself has to get faster.

Why faster simulation?

The CPU that CMS expects to need through the HL-LHC, compared with a flat budget.



Run 4 multiplies the need.
With HL-LHC pileup and trigger rates, the projected CPU need grows about five-fold from 2030 to 2033.

Hardware alone will not close the gap.
Buying more CPU at a flat budget closes the gap only slowly, so the software itself has to get faster.

Simulation follows the data.
Analyses need 5 to 10 times more simulated events than recorded data.

CMS CPU needs at the HL-LHC grow faster than a flat budget can provide.

Full, fast and flash simulation

How a CMS event is simulated today, and which parts FastSim and FlashSim replace.

FullSim is the reference. Geant4, digitisation and the full reconstruction take about 30 s per event per core.

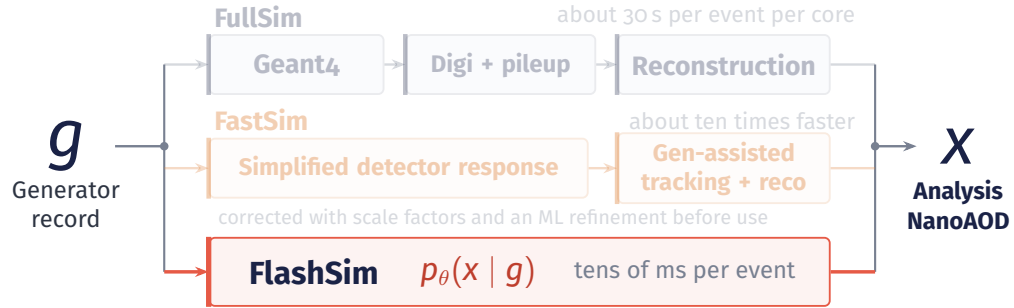
Full, fast and flash simulation

How a CMS event is simulated today, and which parts FastSim and FlashSim replace.

FastSim replaces the two expensive steps with parametrisations and keeps the rest. It is about ten times faster and is corrected before use.

Full, fast and flash simulation

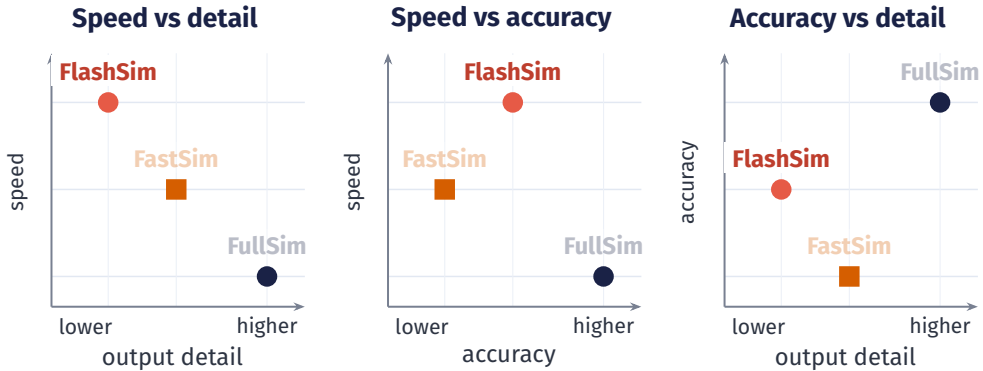
How a CMS event is simulated today, and which parts FastSim and FlashSim replace.



FlashSim is a machine-learning model trained on FullSim. It skips the chain and generates NanoAOD from the generator record in tens of milliseconds per event.

Full, fast and flash simulation

The three simulations compared in speed, output detail and accuracy.



FastSim makes the same chain cheaper. FlashSim replaces it with a machine-learning model trained on FullSim output.

2

Model

FlashSim was presented at ML4Jets 2025. A short recap of the model, with the parts that are new since then.

FACTORISATION · FLOW MATCHING · ONE MODULE · FAKE OBJECTS · DISCRETE BRANCHES · DISTILLATION · TRIGGER

Event factorisation

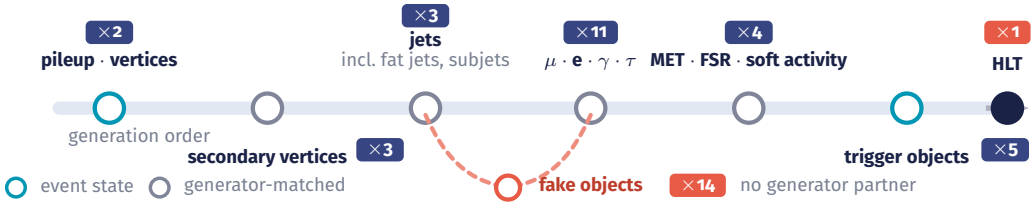
How FlashSim builds an event: one module per object collection, run in generation order.

$$p_\theta(x \mid g) \;=\; \prod_{m=1}^{43} p_{\theta_m}(x_m \mid c_m), \qquad c_m = C_m(g, x_{<m})$$

Event factorisation

How FlashSim builds an event: one module per object collection, run in generation order.

$$p_\theta(x \mid g) = \prod_{m=1}^{43} p_{\theta_m}(x_m \mid c_m), \qquad c_m = C_m(g, x_{<m})$$

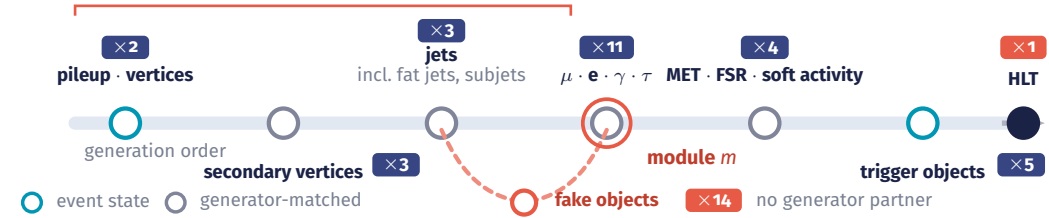


Event factorisation

How FlashSim builds an event: one module per object collection, run in generation order.

$$p_\theta(x \mid g) = \prod_{m=1}^{43} p_{\theta_m}(x_m \mid c_m), \qquad c_m = C_m(g, x_{<m})$$

c_m: the generator record g and everything generated before module m



Each module is conditioned on the generator record and on the collections generated before it. That is how correlations between collections enter.

Conditional flow matching

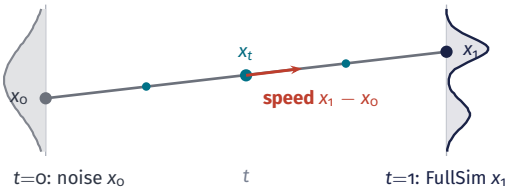
How the continuous features of one object are generated: a network is trained to transport Gaussian noise to the FullSim distribution, conditioned on the object's generator-level information.

THE TASK

Transform noise into the continuous features of one object, all features together, with the FullSim distribution for that c.

Conditional flow matching

How the continuous features of one object are generated: a network is trained to transport Gaussian noise to the FullSim distribution, conditioned on the object's generator-level information.



PAIR A NOISE SAMPLE WITH A FULLSIM SAMPLE

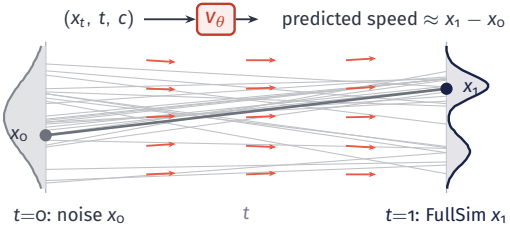
A noise value x_0 is paired with a FullSim value x_1 of the same object and joined by a straight line, $x_t = (1-t)\,x_0 + t\,x_1$, so the point moves at constant speed $x_1 - x_0$.

THE TASK

Transform noise into the continuous features of one object, all features together, with the FullSim distribution for that c.

Conditional flow matching

How the continuous features of one object are generated: a network is trained to transport Gaussian noise to the FullSim distribution, conditioned on the object's generator-level information.



PAIR A NOISE SAMPLE WITH A FULLSIM SAMPLE

A noise value x_0 is paired with a FullSim value x_1 of the same object and joined by a straight line, $x_t = (1-t)\,x_0 + t\,x_1$, so the point moves at constant speed $x_1 - x_0$.

THE TASK

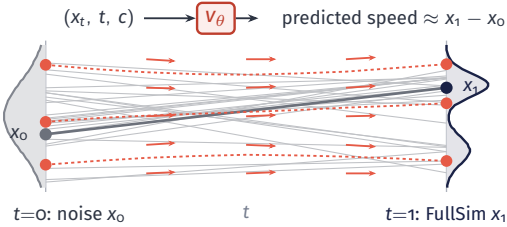
Transform noise into the continuous features of one object, all features together, with the FullSim distribution for that c.

A NETWORK LEARNS THAT SPEED

$v_\theta(x_t, t \mid c)$ takes the position, the time and c and is trained to return $x_1 - x_0$ with the loss $\mathcal{L} = \mathbb{E}\,\|v_\theta - (x_1 - x_0)\|^2$, an ordinary regression.

Conditional flow matching

How the continuous features of one object are generated: a network is trained to transport Gaussian noise to the FullSim distribution, conditioned on the object's generator-level information.



PAIR A NOISE SAMPLE WITH A FULLSIM SAMPLE

A noise value x_0 is paired with a FullSim value x_1 of the same object and joined by a straight line, $x_t = (1-t)\,x_0 + t\,x_1$, so the point moves at constant speed $x_1 - x_0$.

THE TASK

Transform noise into the continuous features of one object, all features together, with the FullSim distribution for that c.

A NETWORK LEARNS THAT SPEED

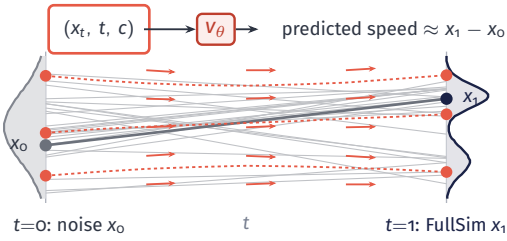
$v_\theta(x_t, t \mid c)$ takes the position, the time and c and is trained to return $x_1 - x_0$ with the loss $\mathcal{L} = \mathbb{E}\,\|v_\theta - (x_1 - x_0)\|^2$, an ordinary regression.

GENERATE

Draw new noise and follow the predicted speed for a few Euler steps. The endpoint is the generated object.

Conditional flow matching

How the continuous features of one object are generated: a network is trained to transport Gaussian noise to the FullSim distribution, conditioned on the object's generator-level information.



PAIR A NOISE SAMPLE WITH A FULLSIM SAMPLE

A noise value x_0 is paired with a FullSim value x_1 of the same object and joined by a straight line, $x_t = (1-t)\,x_0 + t\,x_1$, so the point moves at constant speed $x_1 - x_0$.

THE TASK

Transform noise into the continuous features of one object, all features together, with the FullSim distribution for that c.

A NETWORK LEARNS THAT SPEED

$v_\theta(x_t, t \mid c)$ takes the position, the time and c and is trained to return $x_1 - x_0$ with the loss $\mathcal{L} = \mathbb{E}\,\|v_\theta - (x_1 - x_0)\|^2$, an ordinary regression.

GENERATE

Draw new noise and follow the predicted speed for a few Euler steps. The endpoint is the generated object.

WHAT c CONTAINS

the object's generator-level quantities, its surroundings, the pileup and the collections generated before it. Which of these enter is chosen per module.

One flow per collection generates all the continuous features of an object at once.

Training one module

What one module learns from FullSim: three factors, trained separately on the same pairs of conditioning c_m and reconstructed object x_m.

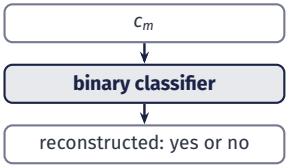
$$p_{\theta_m}(x_m \mid c_m) \;=\; p(\text{reco} \mid c_m) \cdot p(x^{\text{cont}} \mid c_m) \cdot p(x^{\text{disc}} \mid c_m,\, x^{\text{cont}})$$

Training one module

What one module learns from FullSim: three factors, trained separately on the same pairs of conditioning c_m and reconstructed object x_m.

$$p_{\theta_m}(x_m \mid c_m) = \boxed{p(\text{reco} \mid c_m)} \cdot p(x^{\text{cont}} \mid c_m) \cdot p(x^{\text{disc}} \mid c_m, x^{\text{cont}})$$

1 IS IT RECONSTRUCTED?



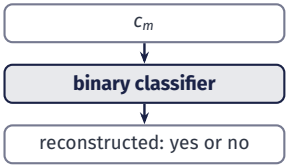
target: FullSim: matched or not
loss: cross-entropy

Training one module

What one module learns from FullSim: three factors, trained separately on the same pairs of conditioning c_m and reconstructed object x_m.

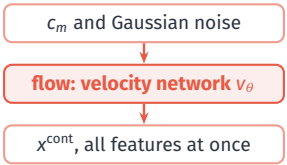
$$p_{\theta_m}(x_m \mid c_m) = p(\text{reco} \mid c_m) \cdot p(x^{\text{cont}} \mid c_m) \cdot p(x^{\text{disc}} \mid c_m,\, x^{\text{cont}})$$

1 IS IT RECONSTRUCTED?



target: FullSim: matched or not
loss: cross-entropy

2 CONTINUOUS FEATURES



target: matched FullSim x^{cont}
loss: $\|v_\theta - (x_1 - x_0)\|^2$

Training one module

What one module learns from FullSim: three factors, trained separately on the same pairs of conditioning c_m and reconstructed object x_m.

$$p_{\theta_m}(x_m \mid c_m) = p(\text{reco} \mid c_m) \cdot p(x^{\text{cont}} \mid c_m) \cdot p(x^{\text{disc}} \mid c_m,\, x^{\text{cont}})$$

1 IS IT RECONSTRUCTED?

c_m

↓

binary classifier

↓

reconstructed: yes or no

target: FullSim: matched or not
loss: cross-entropy

2 CONTINUOUS FEATURES

c_m and Gaussian noise

↓

flow: velocity network v_θ

↓

x^{cont}, all features at once

target: matched FullSim x^{cont}
loss: $\|v_\theta - (x_1 - x_0)\|^2$

3 INTEGER FEATURES

c_m and x^{cont}

↓

categorical heads

↓

x^{disc}: charge, flags, modes

target: matched FullSim x^{disc}
loss: cross-entropy, then calibrated

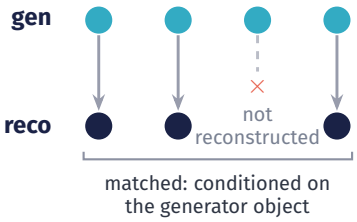
Efficiency, continuous features and integer features are trained separately for each collection, on the same FullSim pairs.

Fake objects

Reconstructed objects with no generator particle or jet behind them, and how they are generated.

NO SOURCE TO START FROM

The other modules generate each object from a source, a generator particle or a jet.

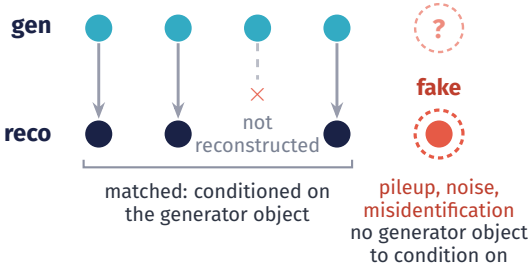


Fake objects

Reconstructed objects with no generator particle or jet behind them, and how they are generated.

NO SOURCE TO START FROM

The other modules generate each object from a source, a generator particle or a jet. Objects from pileup, noise or misidentification have neither.



Fake objects

Reconstructed objects with no generator particle or jet behind them, and how they are generated.

NO SOURCE TO START FROM

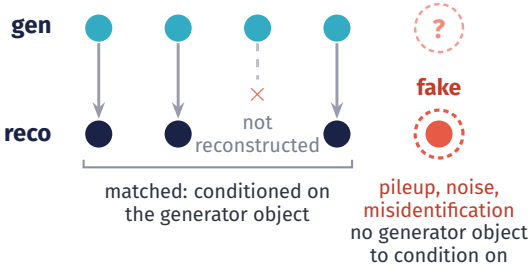
The other modules generate each object from a source, a generator particle or a jet. Objects from pileup, noise or misidentification have neither.

COUNT, KINEMATICS, FEATURES

FlashSim draws the number of fakes, their kinematics in one joint draw, and the features of each.

$$p(\text{fakes} \mid c) = \underbrace{p(N \mid c)}_{\text{how many}} \underbrace{p(\text{kin}_{1..N} \mid c)}_{\text{where, jointly}} \underbrace{\prod_i p(\text{feat}_i \mid \text{kin}_i, c)}_{\text{what, each}}$$

c: pileup, energy density, generator jets, leptons and MET.



Fake objects

Reconstructed objects with no generator particle or jet behind them, and how they are generated.

NO SOURCE TO START FROM

The other modules generate each object from a source, a generator particle or a jet. Objects from pileup, noise or misidentification have neither.

COUNT, KINEMATICS, FEATURES

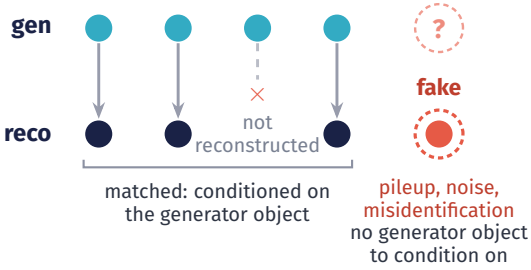
FlashSim draws the number of fakes, their kinematics in one joint draw, and the features of each.

$$p(\text{fakes} \mid c) = \underbrace{p(N \mid c)}_{\text{how many}} \underbrace{p(\text{kin}_{1..N} \mid c)}_{\text{where, jointly}} \underbrace{\prod_i p(\text{feat}_i \mid \text{kin}_i, c)}_{\text{what, each}}$$

c: pileup, energy density, generator jets, leptons and MET.

ONE CHAIN PER FAMILY

Jets, muons, electrons, photons and τ_h each have their own chain of modules, 14 in total.



Fakes are drawn from the event environment, all fakes of a family together.

Discrete branches: categorical heads

How the integer features are generated, with the τ_h decay mode as the example.

FLOW, THEN ROUNDING

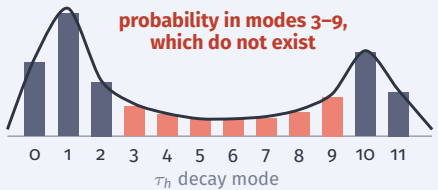
dequantise, transport, round to the nearest integer

Discrete branches: categorical heads

How the integer features are generated, with the τ_h decay mode as the example.

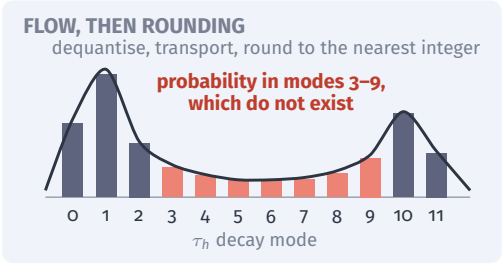
FLOW, THEN ROUNDING

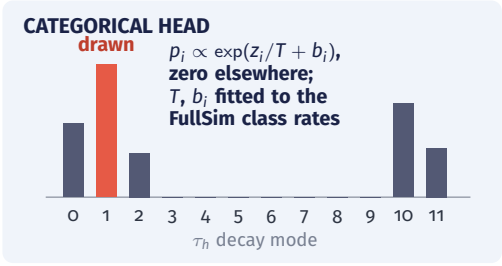
dequantise, transport, round to the nearest integer



Discrete branches: categorical heads

How the integer features are generated, with the τ_h decay mode as the example.





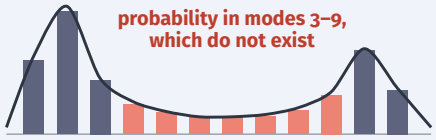
input: c and the object's continuous features $\rightarrow$ small network $\rightarrow$ categorical over the allowed values $\rightarrow$ one draw

Discrete branches: categorical heads

How the integer features are generated, with the τ_h decay mode as the example.

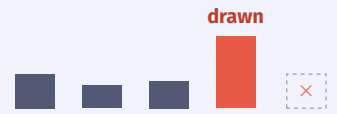
FLOW, THEN ROUNDING

dequantise, transport, round to the nearest integer



ONE HEAD FOR A CORRELATED FAMILY

Loose (L), medium (M), tight (T) muon ID: one draw sets all three, always a combination seen in FullSim



input: c and the object's continuous features → small network → categorical over the allowed values → one draw

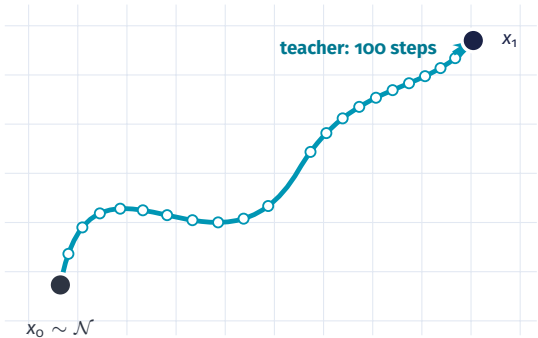
a correlated family (identification working points, trigger bits) is one head over the combinations seen in FullSim: one draw fills all its flags, and combinations never seen cannot be produced.

Integer branches are drawn from a calibrated categorical over the allowed values, and correlated flags are drawn together as one pattern.

Rectified-flow distillation

How the flow of each module, which needs about a hundred Euler steps per object, is distilled into a student that needs a few.

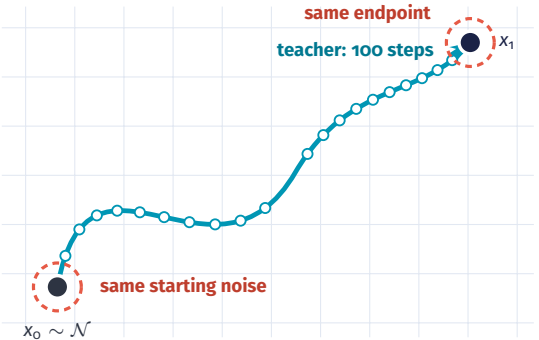
- The velocity field of the teacher is curved, since one noise point can end at many destinations, and needs about a hundred Euler steps per object.



Rectified-flow distillation

How the flow of each module, which needs about a hundred Euler steps per object, is distilled into a student that needs a few.

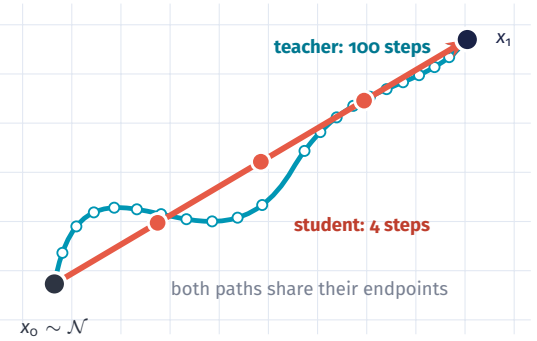
- The velocity field of the teacher is curved, since one noise point can end at many destinations, and needs about a hundred Euler steps per object.
- **Reflow** [Liu et al. 2022]: the teacher regenerates its training set, pairing each noise vector with its own endpoint. A student trained on these pairs learns an almost straight transport to the same endpoints in a few steps.



Rectified-flow distillation

How the flow of each module, which needs about a hundred Euler steps per object, is distilled into a student that needs a few.

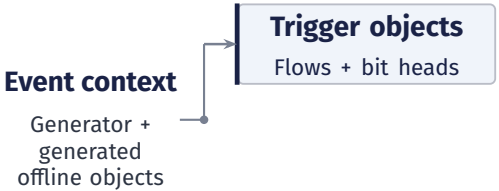
- The velocity field of the teacher is curved, since one noise point can end at many destinations, and needs about a hundred Euler steps per object.
- **Reflow** [Liu et al. 2022]: the teacher regenerates its training set, pairing each noise vector with its own endpoint. A student trained on these pairs learns an almost straight transport to the same endpoints in a few steps.



Reflow straightens the transport, so a few Euler steps reproduce the hundred-step teacher.

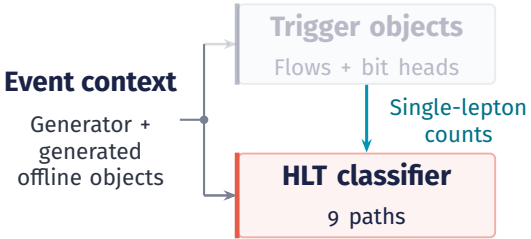
Trigger simulation

How the trigger objects, their filter bits and the HLT decisions are produced.



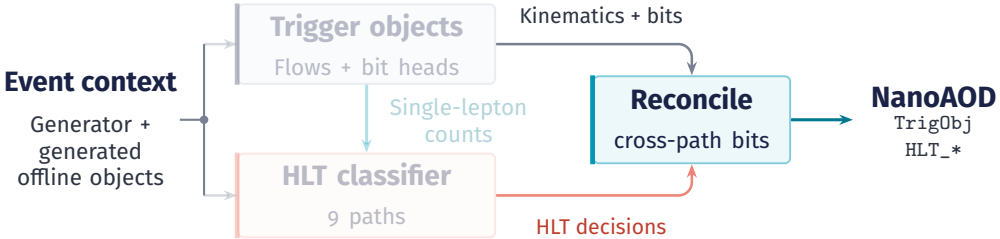
Trigger simulation

How the trigger objects, their filter bits and the HLT decisions are produced.



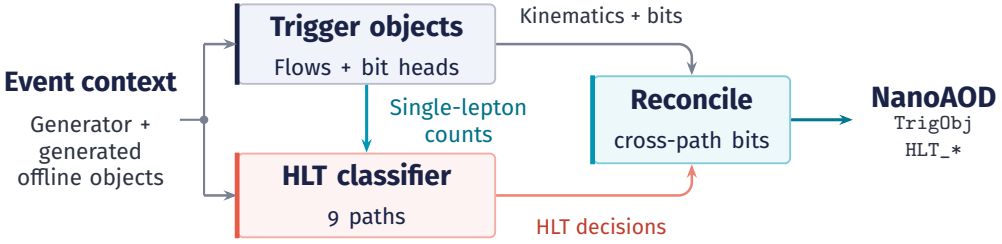
Trigger simulation

How the trigger objects, their filter bits and the HLT decisions are produced.



Trigger simulation

How the trigger objects, their filter bits and the HLT decisions are produced.



Lepton paths

$e,\ \mu,\ ee,\ \mu\mu$

Lepton + tau

$e\tau_h,\ \mu\tau_h$

Tau-based paths

$\tau_h\tau_h,\ \tau_h\tau_h + j,\ \mathrm{VBF} + \tau_h\tau_h$

Trigger objects, filter bits and HLT decisions are generated together and kept consistent with each other.

3

Validation

FullSim, FastSim and FlashSim compared on common Run 3 samples, from single objects to toy analyses. All figures are from CMS-DP-2026/132.

SETUP · **TRIGGER** · **FAKES** · **JETS** · $Z \to \mu\mu$ · $Z(\ell\ell)H(\mathrm{bb})$ · **ELECTROWEAK** Z**+JETS** · $\mathrm{HH} \to \mathrm{bb}\tau\tau$ · **THROUGHPUT**

Comparison setup

The samples and the protocol used in the following slides.

SAMPLES: 2024 DETECTOR CONDITIONS, NanoAOD v15, 13.6 TeV

- DY$\to \mu\mu$+2j ($m_{\ell\ell} > 50$ GeV): 7.1M events for FastSim and FlashSim, 7.3M for FullSim.
- $t\bar{t} \to 2\ell 2\nu$: 8.3M for FastSim and FlashSim, 7.3M for FullSim.

PROTOCOL

- FlashSim is generated from the generator record of the FastSim files. FullSim is a separate production.
- The same code extracts all three, and matching, flavour labels and fake definitions are recomputed in the same way.

REFERENCE SIMULATIONS

FullSim	central production
FastSim	central production

FLASHSIM VARIANTS, ONE TRAINED MODEL SET

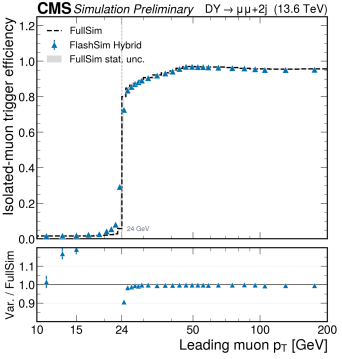
TS1	student, 1 Euler step
TS4	student, 4 steps
TS10	student, 10 steps
Hybrid	per-module step count
Teacher	undistilled, 100 steps

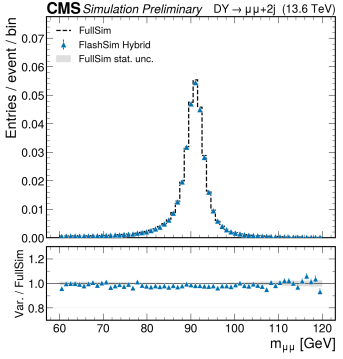
same colours and markers on every figure that follows

FlashSim and FastSim share generator events, FullSim does not. Histograms are per processed event, so ratios compare rates and shapes.

Trigger: turn-on and dimuon mass

The simulated single-muon trigger on Drell–Yan events, Hybrid variant, with tight isolated muons within $|\eta| < 2.4$. Left: the efficiency of the generated trigger decision versus the leading muon p_{T}. Right: the dimuon mass after the trigger.

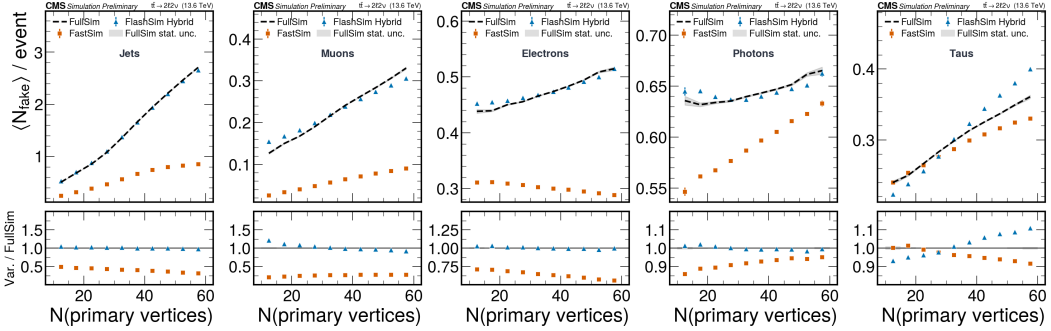




The generated HLT decision reproduces the FullSim turn-on and the dimuon mass after the trigger selection. FastSim has no trigger emulation.

Fake-object rates versus pileup

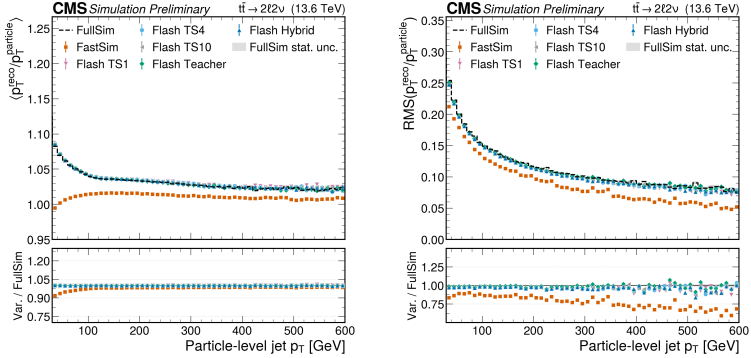
Fake rates versus the number of primary vertices for jets, muons, electrons, photons and hadronic taus in $t\bar{t}$ events, Hybrid variant. A fake is a reconstructed object with no generator partner within ΔR of 0.1 to 0.3, depending on the object, and the same definition is applied to all three simulations.



FlashSim reproduces the fake rates and their growth with pileup for all five object types. FastSim is low by a factor of two to four.

Jets: energy scale and resolution

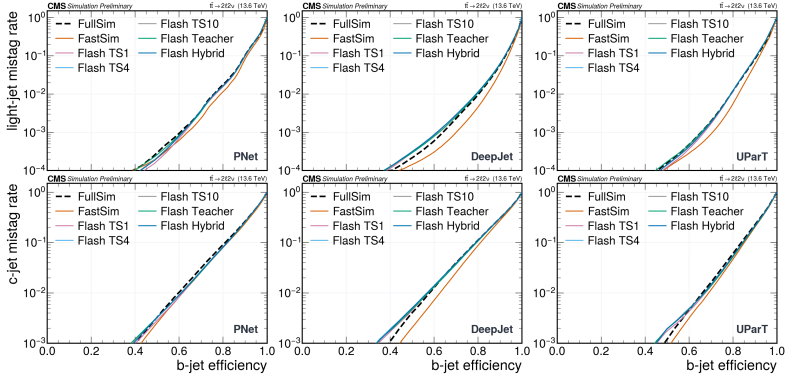
Jet energy scale and resolution in $t\bar{t}$ events: the mean and the RMS of the reconstructed over particle-level jet p_T, as a function of the particle-level p_T, for jets above 30 GeV.



Jet energy scale and resolution are reproduced by all five variants, down to the one-step student. FastSim has a low response at low p_T and a narrower resolution.

Jets: b tagging

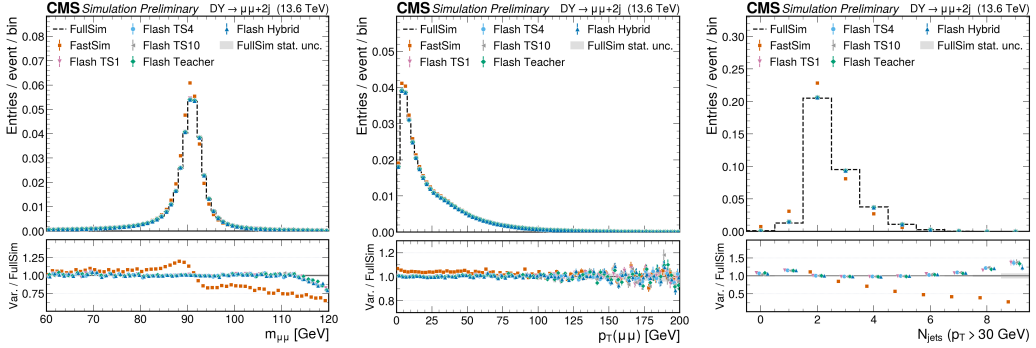
b-tagging performance in $t\bar{t}$ events for the PNet, DeepJet and UParT discriminants: b versus light on top, b versus c below. Jets above 30 GeV within $|\eta| < 2.4$, flavour from the generator-level hadrons.



All three b-tagging discriminants are reproduced by the five variants. FastSim has a lower mistag rate at the same efficiency.

$Z \to \mu\mu$

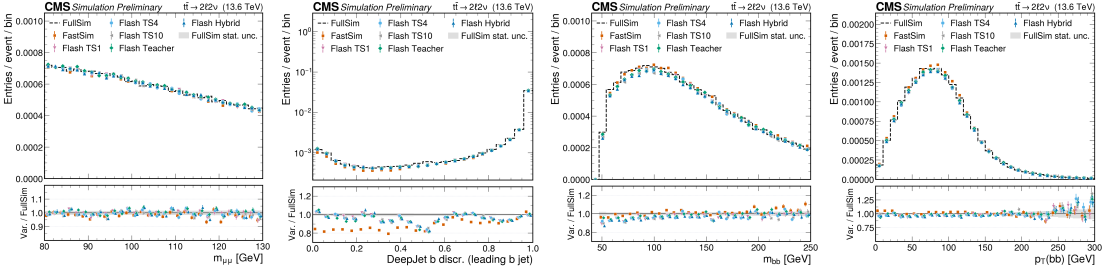
Drell–Yan events with a leading opposite-sign muon pair in the Z window: the dimuon mass, the Z p_{T} and the jet multiplicity above 30 GeV.



Muon scale and resolution, the Z p_{T} and the jet multiplicity follow FullSim. FastSim has a shifted peak and a deficit that grows with the jet multiplicity.

$Z(\ell\ell)H(\mathrm{bb})$: inputs and control regions

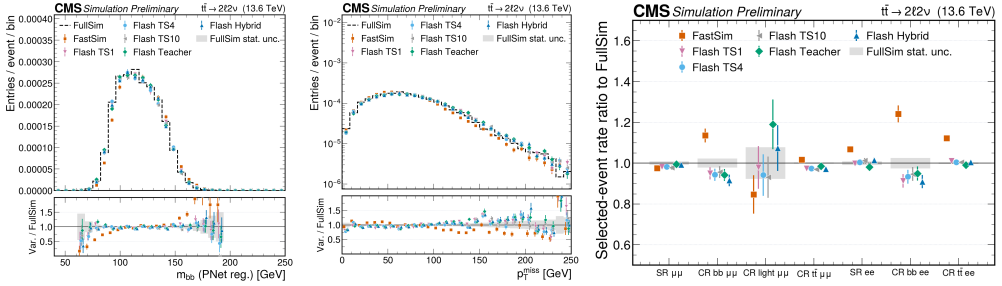
A $Z(\ell\ell)H(\mathrm{bb})$-like selection on $\mathrm{t\bar{t}}$ events, the main background of such a search: two opposite-sign leptons, at least two jets, Z p_T above 50 GeV, the Higgs candidate from the two highest-DeepJet jets. Left: the dilepton mass and the leading b-tag score, the two variables the region cuts use. Right: the b-jet-pair mass and p_T in the $\mathrm{t\bar{t}}$ control region (dilepton-mass sidebands, tight plus loose b tag).



The variables the region cuts use and the control-region spectra follow FullSim with flat ratios for all five variants.

$Z(\ell\ell)H(\mathrm{bb})$: signal region and rates

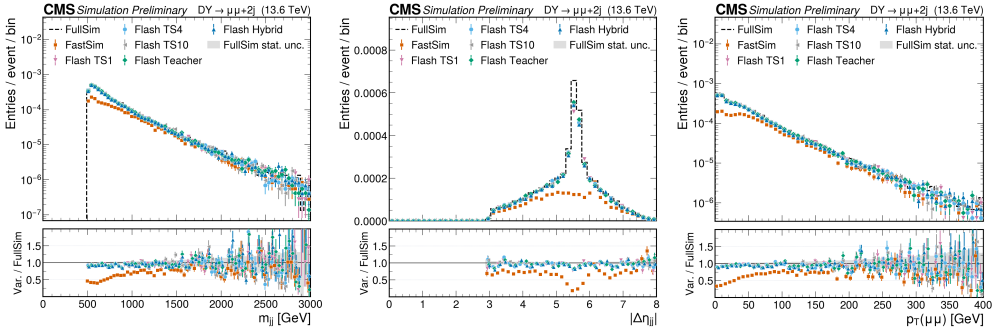
The signal region of that selection, $75 < m_{\ell\ell} < 105\,\mathrm{GeV}$, $90 < m_{bb} < 150\,\mathrm{GeV}$ and a medium b tag, and its three control regions: b-enriched, light-jet and $\mathrm{t\bar{t}}$, each in the $\mu\mu$ and ee channels. Left: the Higgs-candidate mass from PNet-regressed jets. Centre: the missing transverse momentum. Right: the selected-event rates relative to FullSim in every region and channel.



Signal and $\mathrm{t\bar{t}}$ control regions agree within a few percent. The b-enriched region is five to ten percent low, which comes from the missing-momentum tail.

Electroweak Z+jets: signal region

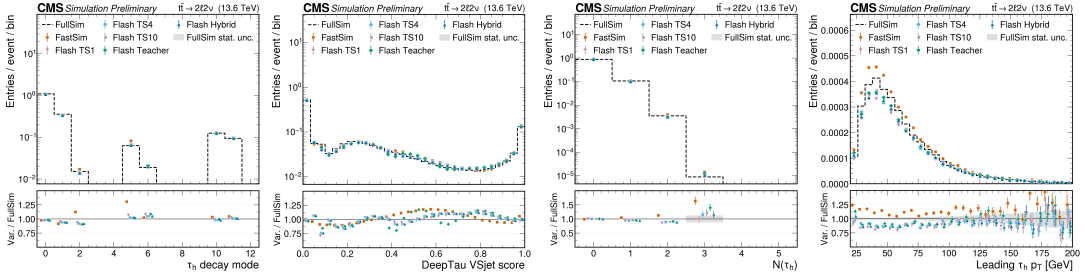
The electroweak Z+2 jets signal region on Drell–Yan events, with jets out to $|\eta| < 4.7$, $m_{jj} > 500\,\text{GeV}$, $|\Delta\eta_{jj}| > 3$, a b-tag veto and low missing momentum, no trigger requirement: the dijet mass, the pseudorapidity separation and the Z p_{T}.



Selected rates agree with FullSim within about five percent and the forward-jet shapes are reproduced. FastSim keeps about half of the rate.

HH → bbττ: hadronic tau objects

Hadronic taus in $t\bar{t}$ events, selected as in an $HH \to bb\tau\tau$ analysis with DeepTau working points: the decay mode, the DeepTau score, the number of selected candidates and the leading τ_h p_T in selected pairs.



Only the physical decay modes are populated and the discriminant shape is reproduced. The two-candidate rate was ten percent low, and twenty percent low in selected pairs.

Tracing a discrepancy: the τ_h charge

The τ_h pair deficit followed from symptom to cause, the sequence we use for any discrepancy.

1 SYMPTOM

The pair rate is twenty percent low after the opposite-sign cut. Single τ_h are fine.

Tracing a discrepancy: the τ_h charge

The τ_h pair deficit followed from symptom to cause, the sequence we use for any discrepancy.

1 SYMPTOM

The pair rate is twenty percent low after the opposite-sign cut. Single τ_h are fine.

2 HYPOTHESIS

If the charge of a fake τ_h does not follow its seed quark, the opposite-sign cut drops half of the pairs with one fake leg.

Tracing a discrepancy: the τ_h charge

The τ_h pair deficit followed from symptom to cause, the sequence we use for any discrepancy.

1 SYMPTOM

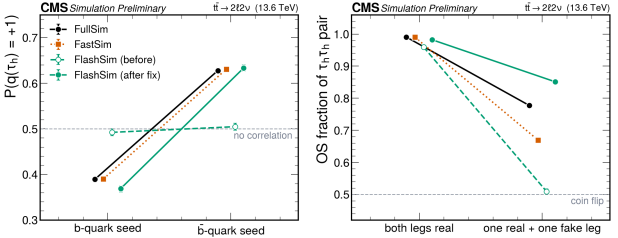
The pair rate is twenty percent low after the opposite-sign cut. Single τ_h are fine.

2 HYPOTHESIS

If the charge of a fake τ_h does not follow its seed quark, the opposite-sign cut drops half of the pairs with one fake leg.

3 TEST

Match each fake τ_h to its seed quark through the seeding jet, then plot its charge against the seed charge, and the opposite-sign fraction by pair composition.



Tracing a discrepancy: the τ_h charge

The τ_h pair deficit followed from symptom to cause, the sequence we use for any discrepancy.

1 SYMPTOM

The pair rate is twenty percent low after the opposite-sign cut. Single τ_h are fine.

2 HYPOTHESIS

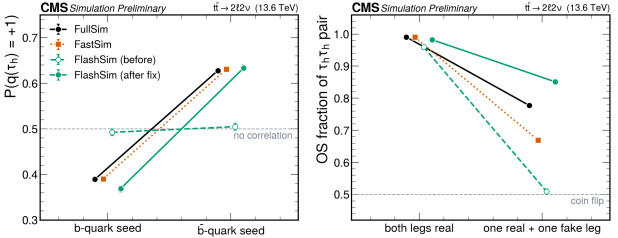
If the charge of a fake τ_h does not follow its seed quark, the opposite-sign cut drops half of the pairs with one fake leg.

3 TEST

Match each fake τ_h to its seed quark through the seeding jet, then plot its charge against the seed charge, and the opposite-sign fraction by pair composition.

4 CAUSE AND FIX

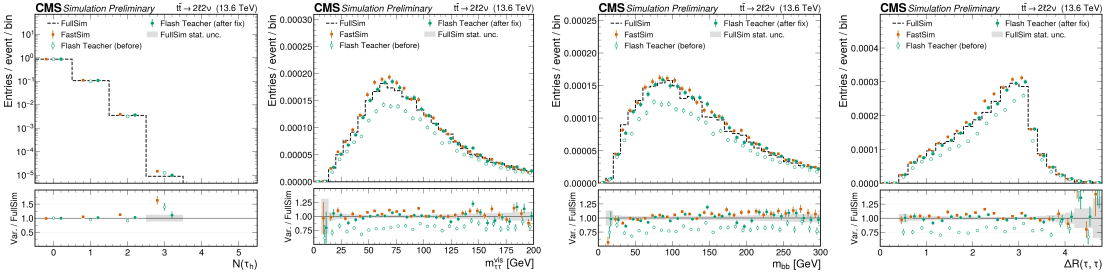
The parton flavour entered as an absolute value, so the sign was invisible. Retrain with the signed flavour.



One missing input explained the deficit, and the retrained model recovers it.

HH → bbττ: after the retraining

The $\mathrm{bb}\tau\tau$ baseline before and after the retraining: opposite-sign τ_h pair, at least two jets, third-lepton veto, b pair from the two highest-DeepJet jets. Teacher variant against FullSim, FastSim for reference.



$\tau_h\tau_h$ PAIR RATE

twenty percent low → within a few percent

BASELINE YIELD

flat deficit → within one percent

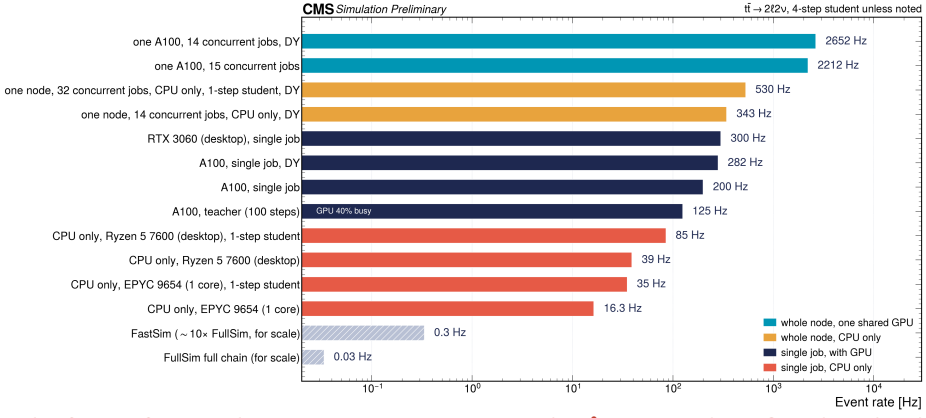
SHAPES

flat ratios before and after, so only the rate was off

After the retraining, ratios are flat and the yield agrees within one percent.

Throughput

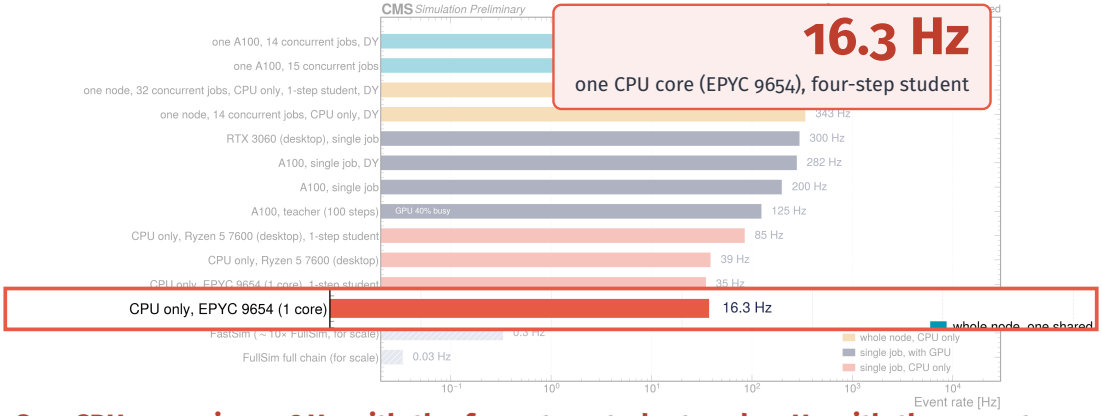
- Event rates per job and per node on different platforms, $t\bar{t}$ events, four-step student unless noted.



Every platform, from a laptop CPU to a GPU node, is more than five hundred times faster than the full chain.

Throughput

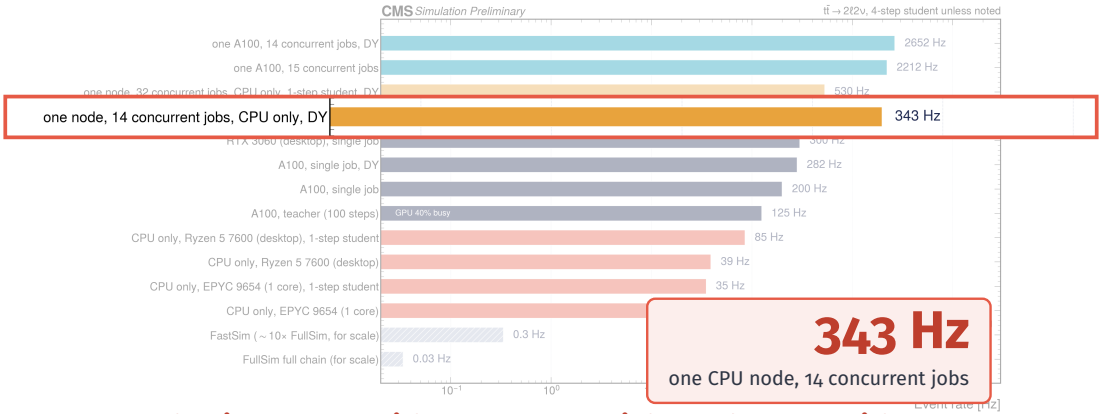
- Event rates per job and per node on different platforms, $t\bar{t}$ events, four-step student unless noted.



One CPU core gives 16 Hz with the four-step student and 35 Hz with the one-step student, with no GPU.

Throughput

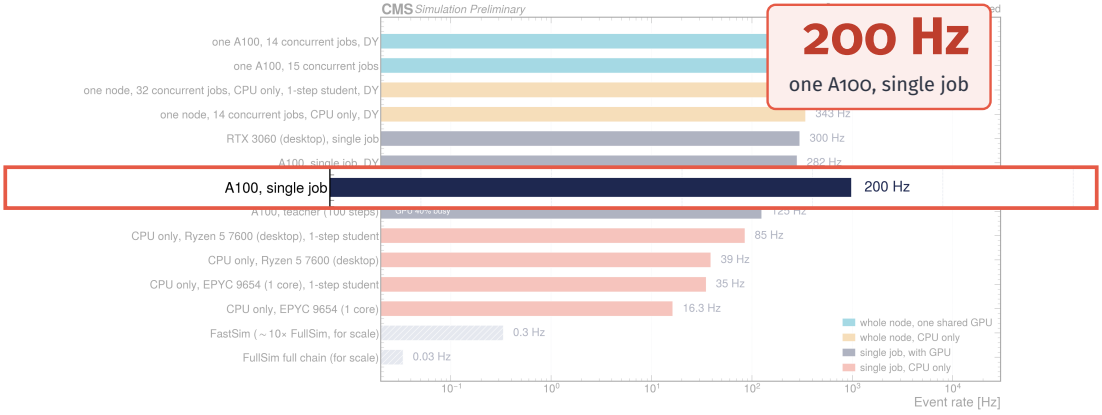
- Event rates per job and per node on different platforms, $t\bar{t}$ events, four-step student unless noted.



One CPU node gives 343 Hz with 14 concurrent jobs, and 530 Hz with 32 one-step jobs.

Throughput

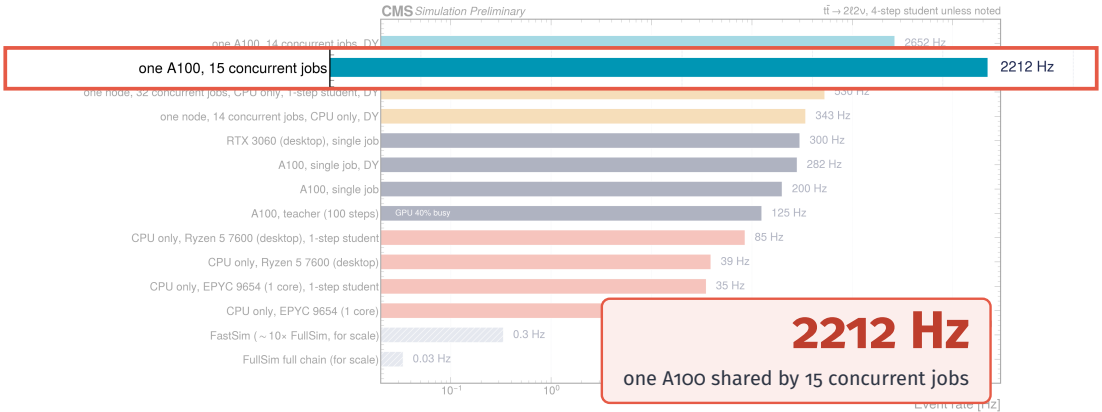
- Event rates per job and per node on different platforms, $t\bar{t}$ events, four-step student unless noted.



- **One GPU with a single job gives 200 Hz, and a single job does not saturate the GPU.**

Throughput

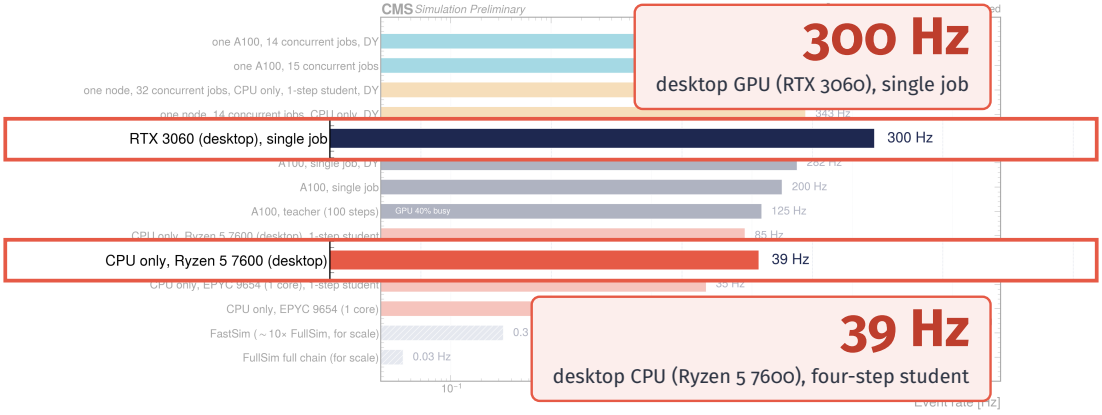
- Event rates per job and per node on different platforms, $t\bar{t}$ events, four-step student unless noted.



- **One A100 shared by 15 jobs gives 2.2 kHz, limited by the output I/O.**

Throughput

Event rates per job and per node on different platforms, $t\bar{t}$ events, four-step student unless noted.



A desktop generates events in a flash, 39 Hz on its CPU and 300 Hz on its GPU.

Summary and conclusions

Where FlashSim stands after this round of development and validation.

NEW SINCE ML4JETS 2025

- Trigger objects, filter bits and HLT decisions are simulated.
- Fake muons, electrons, photons and taus are modelled, where before only fake jets were.
- Larger models and many technical improvements in generation and training.
- Few-step students from reflow: 4 ms per event on a GPU, kHz per GPU node.

VALIDATION

- Trigger, fakes, jets and $Z \to \mu\mu$ on common Run 3 samples, against FullSim and FastSim.
- Three toy analyses run unchanged on every simulation, region yields within a few percent.
- Discrepancies are easy to trace and to fix, as the τ_h charge case showed.
- The first analyses are trying FlashSim samples next to their standard ones.

FlashSim is validated against FullSim from single objects up to analysis-level yields.

B

Backup

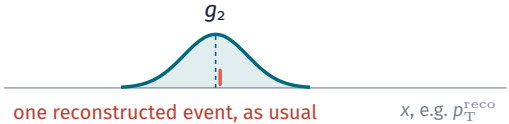
Oversampling, selections, additional validation figures and references.

OVERSAMPLING · **JET CONSTITUENTS** · $Z \to \mu\mu$ **KINEMATICS** · $Z(\ell\ell)H(\mathrm{bb})$ **OBJECTS AND REGIONS** · **TOY SELECTIONS** · **OBJECT DEFINITIONS** · **ELECTROWEAK** Z**+JETS BASELINE** · **TIME PER STAGE** · **REFERENCES**

Oversampling

The generator step runs once and FlashSim draws K reconstructed events from it.

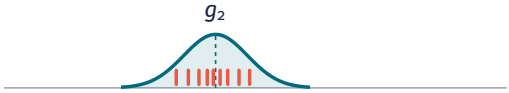
K RESPONSES PER GENERATOR EVENT: $x^{(k)} \sim p_\theta(x \mid g)$



Oversampling

The generator step runs once and FlashSim draws K reconstructed events from it.

K RESPONSES PER GENERATOR EVENT: $x^{(k)} \sim p_\theta(x \mid g)$



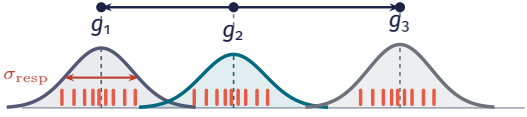
oversampling: K draws from the same g x, e.g. $p_{\mathrm{T}}^{\mathrm{reco}}$

Oversampling

- The generator step runs once and FlashSim draws K reconstructed events from it.

K RESPONSES PER GENERATOR EVENT: $x^{(k)} \sim p_\theta(x \mid g)$

σ_{gen}: the mean moves from one g to the next



oversampling: K draws from the same g x, e.g. $p_{\text{T}}^{\text{reco}}$

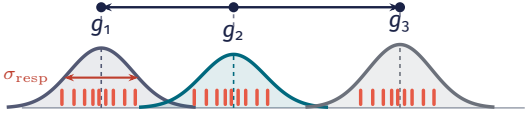
The generator runs once and each response adds milliseconds.

Oversampling

The statistical uncertainty of a mean as a function of K, compared with K independent events.

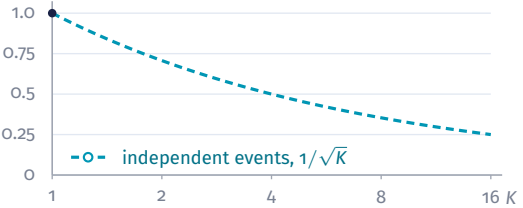
K RESPONSES PER GENERATOR EVENT: $x^{(k)} \sim p_\theta(x \mid g)$

σ_{gen}: the mean moves from one g to the next



oversampling: K draws from the same g x, e.g. $p_{\text{T}}^{\text{reco}}$

PRECISION OF THE MEAN, σ_K / σ_1



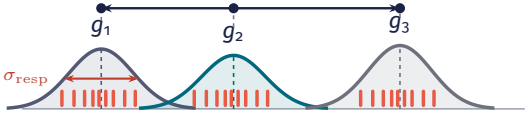
The generator runs once and each response adds milliseconds.

Oversampling

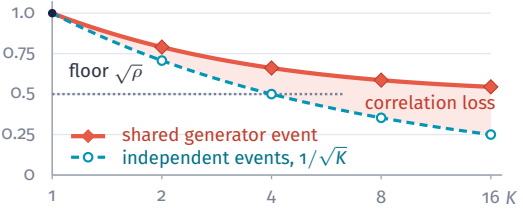
The statistical uncertainty of a mean as a function of K, compared with K independent events.

K RESPONSES PER GENERATOR EVENT: $x^{(k)} \sim p_\theta(x \mid g)$

σ_{gen}: the mean moves from one g to the next



PRECISION OF THE MEAN, σ_K / σ_1



The generator runs once and each response adds milliseconds.

$$\rho = \frac{\sigma_{\text{gen}}^2}{\sigma_{\text{gen}}^2 + \sigma_{\text{resp}}^2}$$

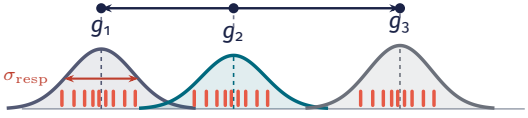
close to one for well-measured quantities, so no gain; small where resolution or efficiency dominate, so a large gain.

Oversampling

The statistical uncertainty of a mean as a function of K, compared with K independent events.

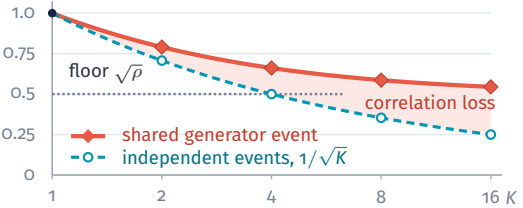
K RESPONSES PER GENERATOR EVENT: $x^{(k)} \sim p_\theta(x \mid g)$

σ_{gen}: the mean moves from one g to the next



oversampling: K draws from the same g $\quad x$, e.g. $p_{\text{T}}^{\text{reco}}$

PRECISION OF THE MEAN, σ_K/σ_1



The generator runs once and each response adds milliseconds.

$$\text{Var}(\widehat{\mu}_K) = \frac{1}{N_{\text{gen}}}\left[\sigma^2_{\text{gen}} + \sigma^2_{\text{resp}}/K\right]$$

unchanged — divided by K

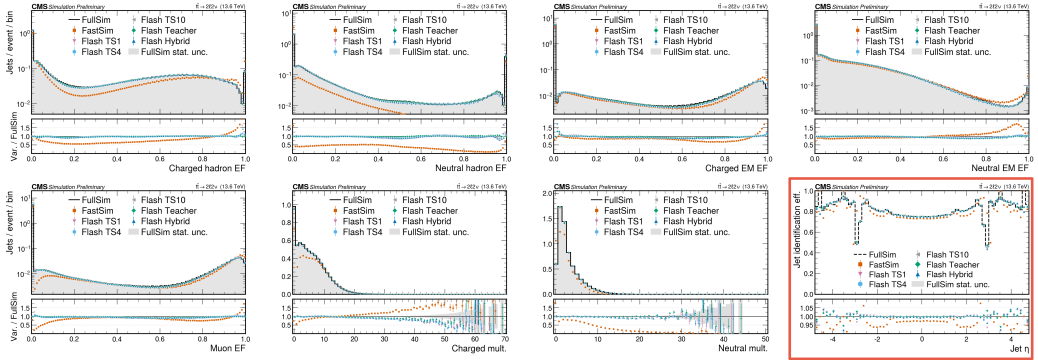
$$\rho = \frac{\sigma^2_{\text{gen}}}{\sigma^2_{\text{gen}} + \sigma^2_{\text{resp}}}$$

close to one for well-measured quantities, so no gain; small where resolution or efficiency dominate, so a large gain.

Oversampling reduces the response term of the uncertainty and leaves the generator term unchanged.

Jet constituents and jet identification

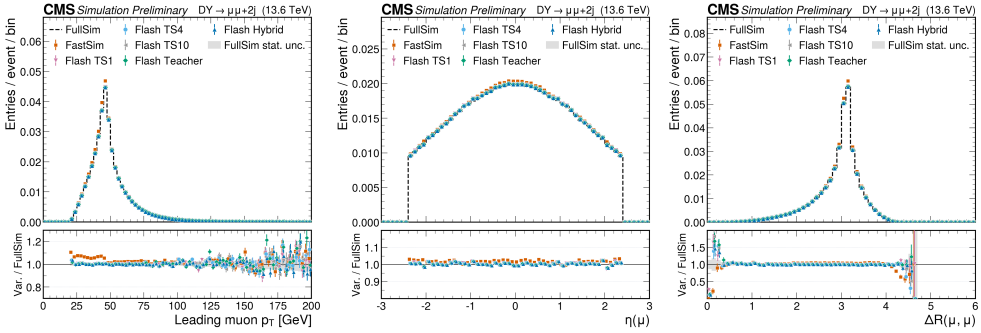
Constituent energy fractions and multiplicities of all reconstructed jets in $t\bar{t}$ events, and the jet identification efficiency versus η that follows from them.



The constituent energy fractions are reproduced, and the jet identification, which depends on their correlations, agrees with FullSim over the full η range.

$Z \to \mu\mu$: muon kinematics

Muon kinematics in the $Z \to \mu\mu$ selection: the leading-muon p_{T}, the muon η and the dimuon opening angle.



The Jacobian peak, the muon η across the detector and the back-to-back dimuon correlation are reproduced with flat ratios by all five variants.

$Z(\ell\ell)H(\mathrm{bb})$: objects and regions

The object and region definitions of a $Z(\ell\ell)H(\mathrm{bb})$-like selection, run on $\mathrm{t\bar{t}}$ events, the main background of such a search.

OBJECTS

- Muons: medium ID, $I_{\mathrm{rel}} < 0.25$, $p_{\mathrm{T}} > 20$ GeV (leading > 25), $|\eta| < 2.4$.
- Electrons: 90% efficiency working point of the multivariate ID, $I_{\mathrm{rel}} < 0.06$, $|d_{xy}| < 0.05$, $|d_z| < 0.2$ cm, $p_{\mathrm{T}} > 25/15$ GeV.
- Jets: $p_{\mathrm{T}} > 20$ GeV, $|\eta| < 2.5$, jet identification, $\Delta R(\ell, j) > 0.4$ cleaning; Higgs candidate from the two highest-DeepJet jets, working points L/M/T $=$ 0.049/0.278/0.710.
- Preselection: exactly two opposite-sign leptons, $\geq$2 jets, Z $p_{\mathrm{T}} > 50$ GeV, $m_{bb} > 50$ GeV; $\mu\mu$ and ee channels.

	$m_{\ell\ell}$	m_{bb}	b tag	other
SR	75–105	90–150	M	–
CR b-enriched	85–97	sideb.	M+L	$p_{\mathrm{T}}^{miss} < 60$, $\Delta\phi > 2.5$
CR light jets	75–105	90–150	veto L	$\Delta\phi > 2.5$
CR $\mathrm{t\bar{t}}$	sideb.	–	T+L	–

masses in GeV; $\Delta\phi \equiv \Delta\phi(Z, H)$; per dilepton channel

The region structure of a search, one signal region and three control regions per dilepton channel, the same in all three simulations.

Toy analyses: selections

Cuts of the electroweak Z+jets and $\mathrm{HH} \to \mathrm{bb}\tau\tau$ toys. The $Z(\ell\ell)H(\mathrm{bb})$ objects and regions are on the previous slide, and none of the three requires a trigger.

ELECTROWEAK Z+JETS, ON DRELL–YAN

- Muons: $p_\mathrm{T} > 20$ GeV, $|\eta| < 2.4$, medium ID, $I_\mathrm{PF} < 0.20$; opposite-sign pair with $70 < m_{\mu\mu} < 110$ GeV.
- Jets: $p_\mathrm{T} > 30$ GeV out to $|\eta| < 4.7$, cleaned against the selected muons ($\Delta R > 0.4$); at least two.
- $\mathrm{t\bar{t}}$ control region: at least one medium b tag and $p_\mathrm{T}^\mathrm{miss} > 40$ GeV.
- Signal region: b veto, $m_{jj} > 500$ GeV, $|\Delta\eta_{jj}| > 3$, $p_\mathrm{T}^\mathrm{miss} < 60$ GeV.

$\mathrm{HH} \to \mathrm{bb}\tau\tau$, ON $\mathrm{t\bar{t}}$

- τ_h: $p_\mathrm{T} > 20$ GeV, $|\eta| < 2.5$, $|d_z| < 0.2$ cm, decay modes 5 and 6 excluded; DeepTau v2p5 VSjet $\geq$ Medium, VSe $\geq$ VVLoose, VSmu $\geq$ Tight.
- Pair: the two leading τ_h, opposite sign, $\Delta R(\tau,\tau) > 0.5$; third-lepton veto (e: WP90, $p_\mathrm{T} > 10$ GeV; μ: medium or tight ID, $I_\mathrm{rel} < 0.3$, $p_\mathrm{T} > 10$ GeV).
- Jets: $p_\mathrm{T} > 20$ GeV, $|\eta| < 2.5$, tight lepton-veto ID, $\Delta R(\tau, j) > 0.5$ cleaning; Higgs candidate from the two highest-DeepJet jets.
- Baseline: pair, at least two jets, veto. b-tag region: leading jet medium, subleading loose.

The same reconstruction-level cuts are applied to the FullSim, FastSim and FlashSim output.

Object definitions

What is selected on the trigger, fake-rate, jet, b-tagging and $Z \to \mu\mu$ slides, identical in all three simulations.

TRIGGER

- Offline muons: tight ID, $I^{\text{rel}}_{\text{PF}} < 0.15$, $|\eta| < 2.4$, $p_{\text{T}} > 3$ GeV; single-muon path efficiency versus the leading muon p_{T}.

FAKES

- Geometric definition, no generator partner within ΔR: jets 0.2; μ, e, γ 0.1; τ_h 0.3. p_{T} floors 15 / 3 / 5 / 10 / 18 GeV for jets / μ / e / γ / τ_h.

JETS

- Scale and resolution: reconstructed jets matched to generator jets within $\Delta R < 0.2$, unique match, versus generator-jet p_{T}.
- b tagging: jets with $p_{\text{T}} > 30$ GeV, $|\eta| < 2.4$, cleaned against leptons; flavour from B and C hadron matching.

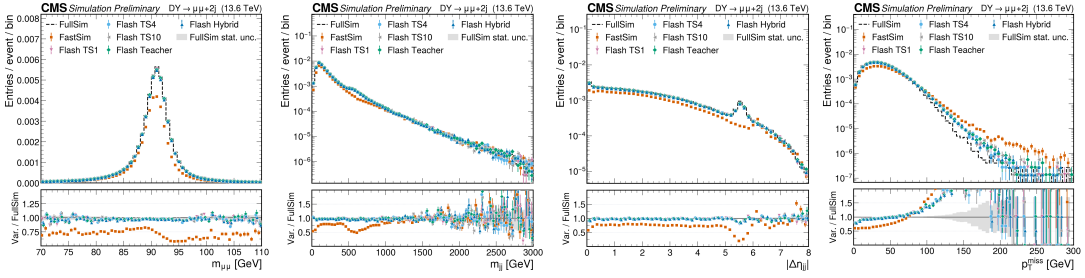
$Z \to \mu\mu$

- Leading opposite-sign muon pair, $p_{\text{T}} > 20$ GeV, $|\eta| < 2.4$, $60 < m_{\mu\mu} < 120$ GeV; jets $p_{\text{T}} > 30$ GeV, $|\eta| < 4.7$.

The same object definitions and matching are used for FullSim, FastSim and FlashSim.

Electroweak Z+jets: baseline

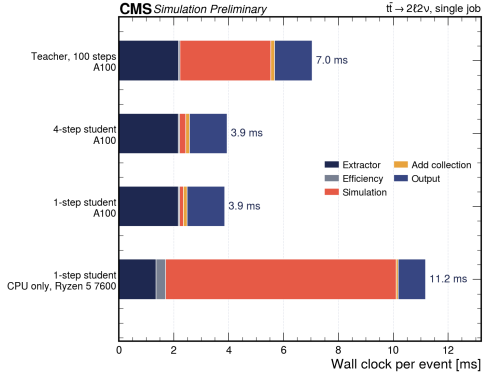
The electroweak Z+jets baseline before the VBF-style cuts: the dimuon mass, the dijet mass, the pseudorapidity separation and the missing transverse momentum.



Dimuon and dijet kinematics are reproduced over the full jet acceptance, forward jets included. FastSim is globally low and overshoots the missing-momentum tail.

Time per event by stage

Wall-clock time of a single job per event, split into input extraction, efficiency models, the simulation itself, adding collections and output writing.



On a GPU the student spends most of its time on input and output. On a CPU the simulation itself dominates.

References

- CMS Collaboration, *Developments and validation of the FlashSim end-to-end simulation on Run 3 samples*, CMS Detector Performance Note CMS-DP-2026/132 (2026), https://cds.cern.ch/record/2970438.
- F. Vaselli, F. Cattafesta, P. Asenov, A. Rizzi, *End-to-end simulation of particle physics events with flow matching and generator oversampling*, Mach. Learn.: Sci. Technol. **5** (2024) 035007, arXiv:2402.13684.
- F. Vaselli, A. Rizzi, F. Cattafesta, G. Cicconofri, *FlashSim prototype: an end-to-end fast simulation using normalizing flow*, CMS-NOTE-2023-003.
- Y. Lipman et al., *Flow matching for generative modeling*, arXiv:2210.02747.
- X. Liu, C. Gong, Q. Liu, *Flow straight and fast: learning to generate and transfer data with rectified flow*, arXiv:2209.03003.
- C. Guo et al., *On calibration of modern neural networks*, arXiv:1706.04599.
- A. Rizzi, G. Petrucciani, M. Peruzzi, *A further reduction in CMS event data for analysis: the NANOAOD format*, EPJ Web Conf. **214** (2019) 06021.
- S. Abdullin et al., *The fast simulation of the CMS detector at LHC*, J. Phys. Conf. Ser. **331** (2011) 032049; S. Bein et al., *Refining fast simulation using machine learning*, EPJ Web Conf. **295** (2024) 09032; M. Wolf et al., *Fast Perfekt: regression-based refinement of fast simulation*, SciPost Phys. Core **8** (2025) 021; A. D. Güngördü et al., *Refining jets for CMS Run 3 using fast simulation*, EPJ Web Conf. **337** (2025) 01313.
- CMS Collaboration, *CMS Phase-2 Computing Model: Update Document*, CERN-CMS-NOTE-2022-008.
- *CMS FlashSim: an end-to-end ML approach speeds up simulation in CMS*, talk at ML4Jets 2025, Caltech, 18 August 2025, https://indico.cern.ch/event/1526677/contributions/6530943/.